Molecular dynamics simulations of complex systems including HIV-1 protease

Owain Alfred Kenway

o.kenway@ucl.ac.uk

Department of Chemistry

University College London (UCL)

2010

A thesis submitted in fulfilment of the requirements for the degree of Doctor of Philosophy

Supervisors: Professor Peter V. Coveney
Dr. Lorna Smith
Dr. Stephen Booth

Signed declaration

I, Owain Alfred Kenway confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the thesis.

Signed:

Owain Alfred Kenway

Abstract

Advances in supercomputer architectures have resulted in a situation where many scientific codes are used on systems whose performance characteristics differ considerably from the platform they were developed and optimised for. This is particularly apparent in the realm of Grid computing, where new technologies such as MPIg allow researchers to connect geographically disparate resources together into virtual parallel machines. Finding ways to exploit these new resources efficiently is necessary both to extract the maximum benefit from them, and to provide the enticing possibility of enabling new science. In this thesis, an existing general purpose molecular dynamics code (LAMMPS) is extended to allow it to perform more efficiently in a geographically distributed Grid environment showing considerable performance gains as a result.

The technique of replica exchange molecular dynamics is discussed along with its applicability to the Grid model and its benefits with respect to increasing sampling of configurational space. The dynamics of two sub-structures of the HIV-1 protease (known as the flaps) are investigated using replica exchange molecular dynamics in LAMMPS showing considerable movement that would have been difficult to investigate by traditional methods.

To complement this, a study was carried out investigating the use of computational tools to calculate binding affinity between HIV-1 protease mutants and the drug lopinavir in comparison with results derived experimentally by other research groups. The results demonstrate some promise for computational methods in helping to determine the most effective course of treatment for patients in the future.

Acknowledgements

There are so many people without whom this thesis would not have been possible. First and foremost, my supervisors Professor Peter Coveney, Doctor Lorna Smith and Doctor Stephen Booth without whose tireless efforts and encouragement I would never have finished. I would also like to thank the other members of the HIV subgroup David Wright, Kashif Sadiq, Simon Watson and Ileana Stoica for their helpful suggestions and constructive criticism throughout this project.

I am also indebted to numerous people on Grid projects world-wide not least of all Doru Marcusiu at NCSA, Shantenu Jha at LSU and Jon MacLaren (then at LSU), Kenneth Yoshimoto at SDSC, Brian Toonen and Nicholas Karonis (the MPICH-G2/MPIg developers a UC/ANL) as well as too many people on working for the National Grid Service to name individually.

I would also like to thank Steven Manos and Simon Clifford for their tireless technical support and Nilufer Betik for helping navigate the many administrational complexities that are part and parcel of PhD life.

I would like to thank EPSRC for funding this PhD.

Last of all, I'd like to thank my friends and family for all their support and (most of all) my girlfriend, Heather Kelly, for putting up with all the stress.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

It is generally well understood that the computational power available to researchers is increasing all the time and that with this comes the ability to do new science. As this increase becomes more difficult to obtain through traditional methods, new system architectures require new techniques for efficiently using their available power. As the limits of single core performance are reached, there is a new emphasis on parallelisation. Individual chips now have multiple cores, and parallel machines have an ever greater number of processors requiring codes that scale better and better.

In addition to a rise in performance at the top-end, a trickle down effect (caused largely by the rise of the Linux cluster) has meant that there has been a democratisation of computer power. Small universities who cannot afford a large, traditional supercomputer are able to afford to build and maintain a Linux cluster which they can customise to suit their costs and needs, and the performance characteristics of such machines vary wildly depending upon the interconnects chosen and the nodes used.

In recent years, there has been a rise in interest in Grid computing - geographically distributed, cross-organisational networks of resources which need to be used in an effective way such that their utility is greater than the sum of that of their individual components. Grid computing has many aspects, from cycle-stealing (using unused resources at people's desks) to redundant multi-site data storage for datasets of unprecedented size to linking together supercomputers into a "meta-computer". Meta-computing brings more complexity to the table, from complications arising from heterogeneous architectures to varying performance depending on the performance of different networks for both intra-site and inter-site communications.

A number of tools have been created to enable meta-computing, including some implementations of MPI[1, 2] that use the Grid for communication. Some examples include MPICH-G2[3] and PACX[4]. The advantage of re-implementing MPI for a Grid is that many parallel applications are written in MPI and should be able to be run with little or no modification on top of such a solution.

All of these new architectures present new optimisations problems for developers. Parallel machines with many processors and fast interconnects present a very different problem from a Linux cluster which may have a relatively slow interconnect, or from a meta-computer on a Grid which may have a very non-uniform interconnect architecture. These all vary again from the architecture a legacy code may have been originally written for which (depending on its age) may be aimed at thirty-two or fewer processors, shared memory or vector machines.

Molecular dynamics codes are heavy users of compute resources. They are general purpose, legacy, (often) parallel codes which due to their nature are used for a long time. They are used to model a wide variety of physical systems of a wide variety of sizes and with force-fields that depend entirely on the problem at hand. A number of these codes were written for older architectures and may be running in a sub-optimal way on the modern generation of supercomputers and resources.

1.1 Motivation

Molecular Dynamics codes are heavily used by researchers world-wide (and more specifically here in the Centre for Computational Science) for investigating a variety of systems. Maximising the performance of these codes on the resources available would mean a more efficient use of available resources (in the form of CPU time) and this (as well as a move towards effective, reliable meta-computing) will allow new science.

1.2 Aims of PhD

1. To investigate the performance properties of existing Molecular Dynamics codes on current platforms.

2. Identify the key factors limiting the performance of these codes and modify the codes to avoid them.

3. Use the resulting improved codes to do new science.

Chapter 2

The Grid

2.1 Introduction

A "Grid" is a collection of geographically/organisationally distributed resources connected together in a uniform way. Those resources may be computational in nature but may also be storage or even scientific instruments. The original vision for the Grid was that computational power and other resources would be available to users much in the same way that electrical power is on the electrical grid (and hence the name). Rather than there being any one Grid project, there are numerous Grids both national and international, both general and project specific, for example QCDGrid[1][5], The National Grid Service in the UK[2], and TeraGrid[3] in the US.

The Grid is often proposed as a tool for allowing virtual organisations to be created, bringing together disparate people and resources and indeed this is the case for this project, drawing together resources at EPCC[4], the CCS at UCL and sites on the TeraGrid in the US and the National Grid Service in the UK. This is helped by the fact that UK e-Science certificates (used for authentication and issued by the Grid

[1] QCDGrid web-site: http://www.Gridpp.ac.uk/qcdGrid/
[2] NGS web-site: http://www.ngs.ac.uk/
[3] TeraGrid web-site: http://www.teraGrid.org/
[4] EPCC web-site: http://www.epcc.ed.ac.uk

Operations Support Centre[5]) are accepted by TeraGrid sites in the US.

Much of the drive for Grid projects has been in two main areas. The first of these is in providing general purpose capacity (and to some extent capability) computing at distinct sites to users at institutions without their own large computing resources. A good example of this is the UK National Grid Service which provides access for UK researchers to a number of small Linux clusters across the UK. Until relatively recently, users have been limited to running jobs on these clusters which fit inside a single cluster meaning that the majority of the use of these machines is in running large numbers of small (32 processor or less) jobs which could well be handled by a small local cluster and soon (with the rise in popularity of multi-core processors) by desktop machines. The second variety of Grid project is the bespoke Grid - a Grid constructed for a particular inter-institutional project, for example QCDGrid. The resources for this variety of Grid are those relevant to the problem, and often there will be a more user-friendly layer on top of the Grid middleware.

In this instance, the Grid (and its associated technologies) allows us to marshal resources across the US (and in future the UK) to deploy MD codes, both running on a single supercomputer and running across multiple sites. The Globus toolkit[6] is the standard set of tools used for the majority of Grid projects and is the set of tools used on the TeraGrid sites used in this work. The MPICH-G2[3] toolkit builds on top of this layer.

Grid projects face many challenges. The first challenge is organisational. Most Grid projects rely upon third party resources and cross institutional boundaries. Different organisations have different policies for machine access and different support practises. Many of the requirements for good interoperability between different resources may require changes in policy at one or more sites, and support staff at those sites are often unwilling to surrender autonomy to support the larger organisation. This can be seen in the organisational differences between the TeraGrid and the UK National

[5]GOSC web-site: http://www.Grid-support.ac.uk/

Grid Service. On the TeraGrid, sites retain a good level of autonomy, with each site being responsible for managing accounts in their own way, and many sites issuing their own user certificates. This system is politically easier, but causes significant problems for users, who have to interact with multiple support teams, completely different user environments and a system that layers GSI (the Grid Security Infrastructure) over traditional UNIX accounts. It is possible to use the TeraGrid resources without ever interacting with Grid software, as every user is issued a separate username and password for every resource (or at least site) they have access to. If they choose to use GSI, then they need to maintain the link between their certificate and their UNIX user account across all sites, and make sure that their client software always has up to date certificates for multiple certificate authorities. This autonomy has, however, allowed a large variety of resources to be added to the TeraGrid as the operational requirements of providing a TeraGrid resource are much lower.

In contrast, the National Grid Service is more centrally managed. Certificates are issued from a single authority, this is the only authority which sites and users need to trust. Users may only log in via GSI and when they do for the first time, UNIX accounts are created transparently for them. Access to resources (time) is shared across all sites, and once one has NGS access, they have access to all the resources (with the exception of HPCx/HeCToR which are separate projects). This approach has issues too. Many of the resources on the NGS are relatively underpowered when compared with those available elsewhere (primarily due to funding problems) and the system suffers from at least one single point of failure (the UK e-Science Certificate Authority) an issue that was demonstrated twice in 2008, firstly when an issue with the Debian OpenSSL package caused a number of e-Science related certificates to be generated with weak random numbers, and the second when some of the Certificate Authority certificates were revoked with only a few days notice causing havoc for a number of users and sites.

The second challenge is technical. Much of the software available for managing Grid resources are unreliable and unfriendly. Early in the development of the Grid, this was excusable, but Globus has been in development since 1995 and is still a significant usability barrier. The certificate system is excellent from a security point of view, but this comes at the expense of a very high barrier of usability for new users, and obtuse/difficult to solve errors for experienced ones. Many Grid projects have attempted to hide or remove this usability problem by designing a bespoke web portal for their users, but this is not really practical for general purpose use. A significant refinement of this idea is the RealityGrid Application Hosting Environment (AHE)[7] which provides the user with a user-friendly, general purpose, extensible Java GUI for interacting with Grid resources. Other key software components, like a system for managing resource heterogeneity are unavailable and this task is passed entirely onto the user in the form of the Resource Specification Language (RSL).

It is important to understand however, that often problems attributed by support teams to technical problems are really political issues. Most certificate problems for example, are caused by the interactions between various CAs/users/sites and the "hidden node IP" problem discussed later is a problem created entirely by site-specific security policies. Attempts to correct these problems by technical means are sometimes made (in the case of the latter, complicated port forwarding tools, for example) but these will always be less preferable to a change in policy at the respective institutions.

2.2 Globus

The Globus[6] Toolkit (see [6]) is the de-facto standard Grid toolkit. It provides a security framework based on certificates, a communications library and a job submission system that provides a common, standardised interface to the native batch schedulers on the machines it is running on and many other tools. These tools are intended to be middleware which developers can build upon in order to build tools for users to use on a computational Grid but the toolkit also contains some user tools which can be used for tasks such as job submission and creating proxy certificates. Because they are

[6]Globus project web-site: http://www.globus.org/

intended for developers, these tools are not particularly user-friendly. In addition, the Globus Toolkit itself is (because it is a substantial library of software) rather large and can be quite complicated to install and configure. Many of the issues that applications developers face in using it are discussed in more detail in [8].

All Globus resources and jobs are specified in a language called RSL (or "Resource Specification Language"). Unfortunately, because this language pre-dates the selection of web services, and therefore XML as a primary component of Grid software, this language is not XML-based or even XML-like and is unique to Globus, requiring all new users to learn RSL and its idiosyncrasies which is a usability nightmare.

2.3 MPICH-G2

MPICH is an Open Source implementation of the MPI message-passing standard[1] developed by Argonne National Lab and freely available from the MPICH web-site[7]. It is designed in a modular way so that the actual message-passing system can be selected or even re-written for the platform in use. As an example, the MPI implementation on the IBM eServer Blue Gene machines is a version of MPICH with a communications layer that is specifically written for the network hardware inside an eServer Blue Gene. The communications layer is referred to as a "device", and the default distribution comes with a small number of devices. One of these is the "globus" device which allows communication over a Grid using the Globus Toolkit. The name "MPICH-G2"[3] refers to a recent version of MPICH with a much-improved version of the "globus" device, referred to as the "globus2" device.

If MPICH-G2 is installed correctly, it can use the vendor's MPI library for communication inside a site (which should allow for the best possible performance inside a site) and MPICH-G2 communications between the sites. MPICH-G2 is designed with the intention that coding for it should be no different from coding for any other MPI library,

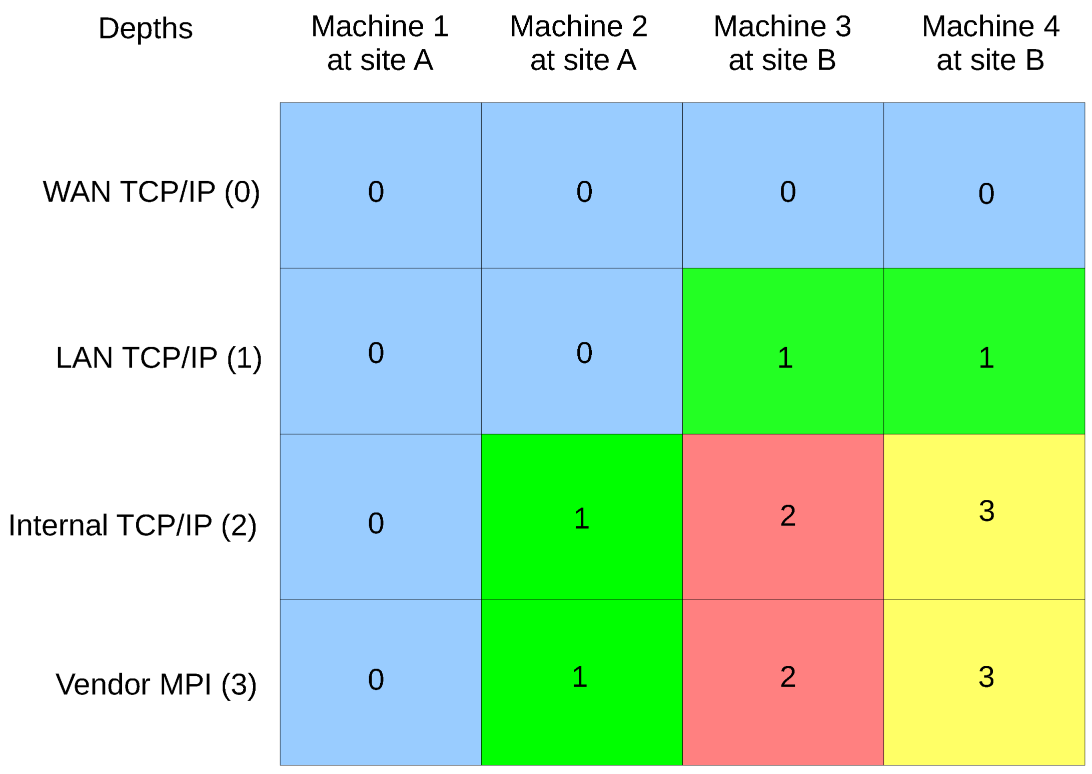
[7]MPICH web-site: http://www-unix.mcs.anl.gov/mpi/mpich/index.htm

and this provides a familiar interface for developers, as well as providing a convenient level of abstraction. It has also been designed so that it is possible to use additional functions and variables provided to discover the configuration of the systems the code is running on and use this information to optimise communications patterns.

One of the major improvements over MPICH-G (the previous implementation) is that the Globus communications library is no longer used for inter-site communications. Instead, MPICH-G2 has its own communications library. This increases the available bandwidth, and also improves the latency when accessing whichever MPI implementation is used of the intra-site communications[3], and allows the use of alternative protocols for inter-site communication such as UDT[9].

The communications topology is described in MPICH-G2 by the use of the terms "depth" and "color". These terms are explained in [3] and that information is summarised here. Depths represent a particular level of communication and at each depth, all the processes of the same colour can communicate with each other at that layer. So for example, given two processes, one at SDSC and one at NCSA, the processes will have the same colour at the TCP/IP depth (since they can talk to each other), but different colours at the vendor MPI layer (since they cannot communicate at this level). This information is stored in arrays that may be obtained by any process and used to construct a list of processes it can communicate with at the vendor MPI layer. It can therefore work out which processors are at different sites from it and from each other. Figure 2.1 shows the colours and depths for four different machines located at two hypothetical sites. All processes can communicate at the WAN (wide area network) layer but at the LAN (local area network) layer only intra-site communication is possible. For both internal TCP/IP and vendor MPI layers, communication is only possible inside a particular machine and so all for machines have different colours at these depths.

When selecting to use MPICH-G2 for a code, it is important to be aware of the "Hidden IP" problem. Specifically, MPICH-G2 (and indeed its predecessor MPICH-G) perform



Depths	Machine 1 at site A	Machine 2 at site A	Machine 3 at site B	Machine 4 at site B
WAN TCP/IP (0)	0	0	0	0
LAN TCP/IP (1)	0	0	1	1
Internal TCP/IP (2)	0	1	2	3
Vendor MPI (3)	0	1	2	3

Figure 2.1: Illustration of colors at different depths for communication of four machines at two sites. If MPIg/MPICH-G2 has been built without vendor MPI support, there is no depth=3 for that machine.

communication by allowing any one node to talk directly to any other. This causes problems in a large number of clustered systems (for example HPCx) where the IP addresses of the nodes are on private networks and are not routable from the other machine. This is not the case for some other implementations of MPI over a Grid (most noticeably PACX-MPI[4]) which route the inter-site communications through a particular node at each site and therefore only that node needs to be routable from the other machine. A work-around used at a number of sites (the latest being HPCx, with the work solution described in [10]) is to use a port-forwarder to forward traffic to the nodes in the private network, essentially replicating the system used by PACX-MPI. The disadvantage of this solution is that it adds another bottle-neck to the inter-site communications.

Because it implements most of the MPI-1 standard, it is not only possible to use MPICH-G2 to develop new applications for deployment on a computational Grid, but also to deploy existing applications across multiple sites on a computational Grid. The possible advances this gives depend on the application at hand, but in general can be summarised as being that the user has access to "more resource", whether that is that the physical memory required for a simulation is greater than they have access to on a single resource, or that they require more processor power than is available in a single system. It is likely that some code modification will have to be made in order to best exploit the available resources (and work around the more complicated communications hierarchy). Other studies[11] have shown that in certain codes with somewhat less intensive communications patterns than molecular dynamics codes, the performance of cross-site runs is considerably better than might be expected given the increased latency and comparatively poor bandwidth of the inter-site links.

Jobs may be run in the traditional MPICH way, or by giving the `mpirun` command an RSL file describing the job managers that the job needs to be submitted to, the number of processors on each, locations of binaries on different systems etc. This builds on top of existing multi-job support in Globus bringing with it a number of usability challenges.

Figure 2.2 shows an example of a three-site RSL file which runs a test job between the Manchester (red), Oxford (blue) and Leeds (green) UK National Grid Service nodes. This sample file highlights a number of usability problems with using RSL to launch jobs. Firstly, nearly every possible variable needs to be specified at each site and this leads to poor readability. Each site has to have the TCP/IP port range specified separately and each has to have the correct site-specific path to the executable and input/output files. Secondly, although there are standard variables, sites often require site specific variables as a side-effect of software or hardware configuration, for example the number of nodes as well as processors, to force the job onto a specific set of nodes, as per Oxford in the example[8] or the `NGSMODULES` variable at Oxford to force Globus libraries into the environment the job runs in which is not the default. Configuration changes at the sites mean that these extra variables change over time, common example being the UNIX environment variable `LD_LIBRARY_PATH` on numerous TeraGrid machines, where common libraries drop in and out of the default library paths as the system is updated.

The over-verbose structure of RSL also hinders code re-use, as each RSL file needs to be tailored to a specific run and cannot be re-used, particularly when reservation-specific variables are added, and no tool exists for automatically generating them. Early work on this project involved developing such a tool, but keeping the site-specific variables up to date generated too much work, making little saving over doing so by hand.

The MPICH-G2 implementation of `mpirun` wraps around the `globusrun` job submission tool. Unfortunately, this tool does not support non-interactive submission of multi-job RSL files for reasons that remain beyond the understanding of the author, meaning that each multi-job run must be left bound to a terminal and closing the terminal cancels the job. This may be mitigated by using the common UNIX tool `screen` but still causes problems when (for example) the login node that a user is using crashes, taking their job with it, despite there being no good reason for this to be the case.

[8]Since this test job was run, the configuration at Oxford has changed and now this is no longer necessary. It is still necessary on some other sites on other general purpose Grids.

```
+
( &(resourceManagerContact="vidar.ngs.manchester.ac.uk/jobmanager-pbs")
   (count=32)
   (jobtype=mpi)
   (maxwalltime=20)
   (label="subjob 0")
   (environment=(GLOBUS_DUROC_SUBJOB_INDEX 0)
                (GLOBUS_ERROR_VERBOSE 1)
                (GLOBUS_TCP_PORT_RANGE 64000,65256))
   (directory="/nfs/users/ngs0027/Experiments/ahm_bench")
   (executable="/nfs/users/ngs0027/src/mpig_lammps/src/lmp_NGS_man_MPIg")
   (arguments="-in" "/nfs/users/ngs0027/Experiments/ahm_bench/ahm.in" "-partition"
"6x16")
   (stdout="/nfs/users/ngs0027/Experiments/ahm_bench/ahm_6x16.out")
   (stderr="/nfs/users/ngs0027/Experiments/ahm_bench/ahm_6x16.err")
)
( &(resourceManagerContact="ngs.oerc.ox.ac.uk/jobmanager-pbs")
   (count=32)
   (host_count=4)
   (jobtype=mpi)
   (maxwalltime=20)
   (label="subjob 1")
   (environment=(GLOBUS_DUROC_SUBJOB_INDEX 1)
                (GLOBUS_ERROR_VERBOSE 1)
                (GLOBUS_TCP_PORT_RANGE 64000,65256)
                (NGSMODULES globus))
   (directory="/home/ngs0039/Experiments/ahm_bench")
   (executable="/home/ngs0039/src/mpig_lammps/src/lmp_NGS_ox_MPIg")
   (arguments="-in" "/home/ngs0039/Experiments/ahm_bench/ahm.in" "-partition" "6x16")
   (stdout="/home/ngs0039/Experiments/ahm_bench/ahm_6x16.out")
   (stderr="/home/ngs0039/Experiments/ahm_bench/ahm_6x16.err")
)
( &(resourceManagerContact="ngs.leeds.ac.uk/jobmanager-pbs")
   (count=32)
   (jobtype=mpi)
   (maxwalltime=20)
   (label="subjob 2")
   (environment=(GLOBUS_DUROC_SUBJOB_INDEX 2)
                (GLOBUS_ERROR_VERBOSE 1)
                (GLOBUS_TCP_PORT_RANGE 64000,65256))
   (directory="/home/ngs0083/Experiments/ahm_bench")
   (executable="/home/ngs0083/src/mpig_lammps/src/lmp_leeds_MPIg")
   (arguments="-in" "/home/ngs0083/Experiments/ahm_bench/ahm.in" "-partition" "6x16")
   (stdout="/home/ngs0083/Experiments/ahm_bench/ahm_6x16.out")
   (stderr="/home/ngs0083/Experiments/ahm_bench/ahm_6x16.err")
)
```

Figure 2.2: Three site RSL file to launch an MPIg/MPICH-G2 job across three sites on the UK National Grid Service. The section of the code for controlling the job at Manchester is shown in red, the section which launches the the Oxford job is shown in blue and the Leeds section is shown in green. Note the large amount of duplication of variables across the three sites, as well as the necessity of site-specific variables (for example, the `NGSMODULES` variable at Oxford) both of which hinder comprehension, code re-use and portability.

2.4 MPIg

MPIg[9] is the successor to MPICH-G2. Like MPICH-G2, it is based on the open-source MPICH MPI implementation, providing a communication layer which uses Globus for Grid-based inter-machine communication. Rather than being an incremental improvement over the old library, MPIg is a complete redesign and re-implementation. It does however, share many concepts with MPICH G2.

Like MPICH-G2, MPIg can use the vendor-specific MPI implementation at each site for intra-site communication. This allows for maximum performance to be extracted from the interconnects. Its inter-site communication also suffers from the “hidden-IP” problem as this is a problem that exists at a level below the inter-site MPI implementation. Its `mpiexec` command takes an RSL file in the same way as MPICH-G2’s `mpirun` and in the same format. The topology discovery mechanisms are the same, although the variable names used have changed, making some trivial code modification necessary when moving between the two MPI implementations.

Both MPIg and MPICH-G2 provide similar topology discovery mechanisms, representing the available virtual machine in terms of “depths” and “colours” as described in 2.3.

In addition it needs to be made clear that it is possible to run MPIg/MPICH-G2 jobs as two distinct sub-jobs at a particular site. In this instance, two processes in different sub-jobs will not be able to communicate with each other at the vendor MPI layer, even though there is no hardware reason for this being the case (there is of course a software issue with the necessary security model for the vendor MPI layer meaning that different MPI jobs cannot and should not be able to talk to each other).

The two main architectural changes are that MPIg uses the newer Globus-XIO library for communications (which allows for greater performance and provides an abstraction layer for different communications protocols) and that MPIg has full support for

[9]MPIg web-site: http://www.mcs.anl.gov/˜toonen/mpig/

threads.

MPIg is fully thread safe (as long as the installation of Globus it runs on is sufficiently up to date) and supports multi-threading. This means that computation and communication may be overlapped, something that is not possible with MPICH-G2, and indeed some vendor implementations of MPI but has always been theoretically possible with the MPI standard. Over-lapping computation and communication may be relatively unimportant on large parallel machines where the interconnect is fast, but on a computational Grid with high latency inter-site communications, it allows a developer to lessen the impact of inter-site communication. This optimisation is only suitable for some algorithms, as for it to be useful, the communication must not be necessary to be complete for the computation it is overlapped with. Considerable success has been shown using the HemeLB lattice-Boltzmann code using this technique[12].

2.5 Co-scheduling

One of the problems with using Grid technologies like MPIg/MPICH-G2 is that they rely on jobs running at the same time across multiple hosts and multiple sites simultaneously. Current methods for achieving this either involve using Globus to submit the jobs to the different sites when there are enough free processors that both jobs start at almost the same time, or contacting the administrators of the sites and requesting that they manually reserve a number of processors for a user to use at a particular time. This is inconvenient both for the user and for the system administration teams at the sites involved, meaning that it is a method of last resort for very large production runs and demos.

2.5.1 The ideal co-scheduler

In order that it is possible to perform cross-site runs in a reliable way on a more regular basis, it is necessary for a tool to be available that performs the reservations automatically. The ideal solution to this problem is analogous to a meta example of the batch scheduler where a user submits a job file with a number of binaries and the total number of processors required and the scheduler finds that number of processors across the available sites transparently. There are a number of challenges associated with achieving this.

The tool itself needs needs uniform access to the Grid resources. This can either be achieved by having the tool installed at all locations where it needs to be used, or else by defining a common standard so that any user front-end can communicate with any combination of co-scheduler back-ends. Unfortunately, both these approaches face problems. In the case of a universal tool, this brings up issues with co-ordination and co-operation of the systems teams at local sites, all of whom need to agree upon a single tool. With standards, these standards need to be agreed and implemented, avoiding the current issues (for example) with the Globus toolkit where constantly changing standards for Grid communication have forced complete re-implementation of many parts of the toolkit. Currently, there are no agreed standards for co-scheduling.

The ideal co-scheduler needs to have the ability to match resources properly with the job requirements automatically without the user specifying where the job will run, and manage the transfer of I/O (and importantly errors) to the proper locations. Some of these aspects are already built into local tools like Condor[10], and existing batch systems.

The tool needs to be reliable. This implies some level of redundancy since the system itself has multiple potential points of failure. It needs to be able to handle the unavailability of processors at a specific site and either fail gracefully (without making any reservations at all) or find appropriate resources elsewhere.

[10]Condor project web page: http://www.cs.wisc.edu/condor/

2.5.2 Co-reservation

Co-reservation side-steps some of the implementation and political problems with the ideal co-scheduler, by replacing it with the ability to automatically reserve a specific amount of resource at specific locations at a specific time. This avoids some of the more difficult challenges (like resource matching) whilst still making the act of performing regular cross-site runs possible for every day research. Two such tools are currently deployed on the large, general purpose Grids used in this project (NGS and TeraGrid), GUR and HARC.

2.5.3 HARC

HARC (Highly-Available Robust Co-scheduler)[13] is a tool for performing co-reservation that is available on both the NCSA and SDSC TeraGrid sites as well as the Manchester and Oxford NGS nodes. It is designed with robustness in mind, and uses an algorithm known as the "Paxos Consensus algorithm"[14] in order to ensure that at the end of the allocation attempt, the system has either made all the necessary reservations or it has made none.

It is designed to be able to schedule resources other than just CPU time. It can (for example) schedule dedicated network links. This is a feature that might become more useful in the future with dedicated fibre-optic networks like JANET light-paths which promise scheduled fast fibre optic links between UK supercomputing sites. It is easy to envisage (for example) scheduling an optic link between two sites which are running the two halves of a pair-wise cross-site MPIg run in order to decrease the performance hit of a cross-site run, or scheduling a link to handle the live transfer of visualisation data to a user's desktop while their simulation is running. Supporting a new resource only requires the development of a new Resource Manager, not a modification of the HARC tools.

HARC is designed in a modular way, with three main components. The Resource Manager (or "RM") is the component that runs on a particular resource, and interacts with that resource's native scheduling system (in the case of a computational resource, a batch scheduler such as LoadLeveller or PBS). This acts as an abstraction layer, providing a standardised interface to perform scheduling operations upon that resource. The second component is the HARC acceptor. A network of these act as intermediaries for performing the reservation, providing robustness by requiring that multiple acceptors agree on the status of the reservation before it is completed.Not all the acceptors need to be available for reservations to be possible. The final component is the HARC client toolkit, consisting of commands for making and deleting reservations, and written in Java for portability. The client talks to the acceptors, and the acceptors talk to the resource managers. Adding a new resource simply requires the appropriate RM to be written, with no modification of the rest of the toolkit, or the other deployed software necessary.

Creating reservations is as simple as issuing a single command:

```
harc-reserve -c testmachine1/4 -c testmachine2/4 -s 15:20 -d 1:00
```

The above command creates a pair of reservations, each of four processors, lasting one hour on two machines, testmachine1 and testmachine2. The reservations start at 15:20 in the time-zone which the workstation the harc-client is running on is in. This time is automatically translated into the appropriate local times on each of the resources. HARC will then attempt to make the reservations, and if the reservations are made successfully, the reservation IDs of the two reservations will be reported to the user. These can then be incorporated into the RSL files used by MPICH-G2/MPIg. HARC also provides commands for cancelling and checking the status of reservations. Because the client tool-kit is written in Java, it is easy to incorporate it into other tools. As part of the GENIUS project, for example, HARC reservation has been integrated both into the general-purpose Application Hosting Environment, and into a bespoke GUI for the project.

2.5.4 GUR

GUR[15] (or "Generic Universal Remote") is a co-reservation tool, consisting of a number of Python scripts which layer on top of the SSH tools to allow a user to perform reservations across multiple sites and supports the GSI-enabled versions of SSH. It is designed to interact with the Catalina scheduler, a job scheduling system designed to plug into existing batch systems such as LoadLeveller or PBS, but can work with any scheduling system that allows users to create reservations.

Users interact with GUR by writing a configuration script called a "jobfile". This jobfile specifies all the requirements of the job, such as the number of processors/RAM required, the length of the job, the sites (including account information) and a time window during which the user wishes the reservation to be made. This configuration file allows the user to request resources in a fuzzy way, for example, they can request n processors over two or three given clusters, but not need to specify how those processors are split over the available resources. This flexibility comes at the expense of usability, much as with RSL, and from the documentation it does not appear to have a simple, HARC-like user interface.

Currently, GUR is supported by a number of sites on the TeraGrid, with official support on the NCSA and SDSC Itanium clusters.

2.5.5 Co-reservation tool selection

Due to the portability of the two systems, a user can use either. The primary requirement for both is that user-settable reservations are available on the resources. The infrastructure cost of GUR is less than HARC, as it does not require the acceptors to act as go-betweens between the user and the resources. To some extent, for simple reservation use, HARC appears over-engineered. GUR is however less easy to use than HARC and HARC is fully deployed across the TeraGrid and NGS resources used for this project. Therefore, work with HARC is the focus here. In the "real world" users need to be prepared to use whatever tools are available on a given resource.

Chapter 3

Molecular Dynamics

3.1 Introduction

Molecular dynamics (MD) codes simulate the interaction between molecules at an atomic level. They do this by calculating the forces between the atoms (and therefore the acceleration) at each time-step and from that calculate the new positions and velocities of all the atoms. Many MD codes are general purpose, implementing a number of different algorithms to allow them to model different problems, and the path that a particular simulation takes may depend a lot on what system is being modeled. They are used in a wide variety of fields, from materials science to chemistry and biology.

Traditionally, the algorithm used updates the forces on each atom and from that calculates the acceleration. It then integrates the acceleration to find the new velocity, and again to find the new position of each atom. There are number of different integration methods that a code might use and they have varying scaling of accuracy with the size of the time-step. Once these have been updated, the algorithm loops around again. In some time-steps it may make a calculation of a physical property like the temperature or the pressure, or it may dump the status of the simulation to disk for later analysis. Some algorithms may adjust the simulation to keep a physical factor constant (like the pressure or temperature of the system) in an effort to remove problems like rounding errors. MD codes are heavy users of CPU time, and may be parallelised in order to run efficiently on parallel machines.

The decomposition strategy and the memory model of the code is an important factor in the performance characteristics of a code. All codes generally use either a replicated data or distributed data model. In the former, each processor has a complete copy of all the data about the entire simulation. This means that when performing the computation, each processor has access to all the data it needs from local memory, but means that each processor has to communicate its data to every other processor (and receive theirs) at every time-step which vastly increases the communications cost. In a distributed data model, each processor owns a sub-set of the data and when it requires data owned by another processor it needs to retrieve the data from the other processor. Typically, thanks to the use of neighbour lists (described in more detail in 3.5.2), this means that overall there is less communication required. In addition, it is easier to fit the simulation in memory as each processor only needs to hold a fraction of the total data.

The decomposition strategy is the way in which the work done by the processors is shared out. In a distributed data model code like LAMMPS, there are a number of ways this could be done. The volume that the simulation takes place in could be broken down into equal-sized blocks with each processor getting a block. The force computations could be split up so that each processor performs an equal portion of the force-calculations, or the atoms in the simulation could be divided up equally between the processors. This matter is investigated in great depth as part of [16].

There are a number of computational optimisations that may be made. Because at large separation the force between a pair of atoms generally tends to zero, a technique called neighbour lists decreases the amount of computation required by assuming that for a large enough radius between a pair of atoms, the force is zero. A technique called "rESPA" [17] allows the developer to make use of the fact that the different forces on an atom can be calculated to different levels of accuracy before they become detrimental to the accuracy of the model, and a technique called “PPPM” (or “Particle-Particle Particle-Mesh”)[17] allows the electrostatic forces to be calculated more efficiently than the naive approach. These methods are all used in the LAMMPS code which is used in this project.

3.2 Application of molecular dynamics codes

There are many uses of molecular dynamics codes in a wide variety of areas of study in the chemical, biological and materials sciences. One of these uses is to study materials science, an active research area here at the CCS. Many of the materials studied here require large and computationally intensive simulations and it is this that drives the need for large high performance computing and Grid resources.

One of the areas of research at the CCS is the study of the behaviour of clay minerals. These are a species of materials that have a structure built up in layers or sheets. C. Greenwell *et. al.* [18] contains a thorough discussion of the applications of computational methods to modelling these materials and the accuracy of various methods when compared with real-world experiment, along with a detailed discussion of the structure of these materials. In particular they mention that the primary advantage of using Molecular dynamics to model these materials is that this allows the scientist to view the evolution of the system over time (using visualisation tools) which can lead to a better understanding of why the simulated material behaves the way that it does.

These simulations can be very large, with hundreds of thousands or even millions of atoms. They select large systems to decrease the chance of accidentally introducing inaccurate behaviour due to the periodic boundary conditions. Due to the fact that long-range electrostatic forces have to be modelled, PPPM techniques are used to improve the performance and accuracy of the simulations. They also make use of rESPA. They use the LAMMPS code (and indeed the benchmarks used in [29] are based on versions of their models for an older version of LAMMPS) because of its scalability and ability to handle very large systems.

The increase in computational power available has allowed larger and larger simulations to be run, and as the simulations increase in size, the accuracy of the model tends to increase. This in turn leads to the the correlation between the behaviours that are observed in simulation and those which are observed in experiment increasing. In the very largest of the simulations run by researchers at the CCS ([18] suggests 1 million atoms but recently, simulations of up to 10 million atoms have been run[1]), physical behaviours that are seen in real-world experiments (such as undulations in the layers) have been observed[18]. This implies that the simulations are, at this scale, becoming sufficiently accurate that it may shortly become possible to use them to predict the results of real-world experiment. Running these codes on the Grid potentially allows access to larger amounts of hardware than is available at just one site.

3.3 Force fields

In molecular dynamics, a force-field is a method of parametrising the forces between atoms within a system. There are a number of commonly used force-fields available, but the most commonly used are variations of the AMBER[19, 20, 21] force-field and the CHARMM force-field, both named after the molecular dynamics packages they were originally developed for. Because these particular methods have become so popular (due the popularity of those codes leading to availability pre-parametrised systems), they are commonly implemented in new molecular dynamics codes. Both of the molecular dynamics codes used here for this thesis implement these force-fields.

In the studies presented here, when the AMBER force fields are used, the General AMBER Force Field[22] is used to parametrise the ligands the standard AMBER forcefield for biorganic systems[23] is used to parametrise the protease. For work with LAMMPS

[1]Source: Professor P. V. Coveney, private communication.

presented here, the CHARMM force-field[24] is used due to issues converting systems using the AMBER force-field for use in LAMMPS.

An alternative to using a full force-field approach is to use Brownian dynamics. In [25], Chia-En Chang et. al. represent individual amino acids as beads with simplified interactions between them designed to roughly approximate the bonds between amino-acids.

3.4 NPT vs. NVT constraints

In order for the researcher to calculate bulk properties of the system they are modelling it is necessary to constrain some of the physical parameters of the system. There are three physical parameters to consider - pressure, volume and temperature. Assuming the researcher wishes to model a physical system at a particular temperature (common for biological systems) then they have a choice of restricting either the pressure of the system, or the volume of the system. In an NPT system the pressure is kept constant and in an NVT system the volume is kept constant. Both keep the temperature constant. For the work presented here, the biological systems are modelled using an NPT system as constant pressure/temperature conditions allow the calculation of the minimum Gibbs free energy and therefore change in the Gibbs free energy when two molecules are bound together, as given in equation 3.1 below, where H is the enthalpic and $-T\Delta S$ is the entropic contribution to the change.

$$\Delta G_b = \Delta H - T\Delta S \tag{3.1}$$

3.5 LAMMPS

LAMMPS[16, 17] (Large-scale Atomic/Molecular Massively Parallel Simulator) is a molecular dynamics code developed and maintained by Steve Plimpton at the Parallel

Computational Sciences Department at Sandia National Lab. It is a distributed data model code and the parallel portions of the code are written using MPI. It is a general purpose code, with the characteristics of the simulation defined by an input file.

There are two version of LAMMPS. The first one is an older, legacy version written in Fortran with MPI which has been used previously for benchmarking studies and is referred to here as LAMMPS 2001. The newer version (referred to as LAMMPS 2005) is the version which will be investigated here and is a complete re-write in C++, again with MPI communications. The formats of the input files for the two versions of the code are slightly incompatible meaning that in order to move to the new version the input files may have to be re-written. Both versions of the code are available for download and use under the GNU Public License from the LAMMPS web-site[2]. Along with the software, a number of examples and benchmarks are provided. LAMMPS is currently used in a number of areas of research at the Centre for Computational Science at University College London and as a result the performance of this code on current and future supercomputers is important. This is the motivation behind this work. This code has also been the subject of previous work[29].

LAMMPS implements a number of algorithms, which are thoroughly discussed in [16] and [17]. The information therein relevant to this report is summarised in sections 3.5.1 to 3.5.7.

3.5.1 Decomposition

LAMMPS decomposes the simulation using "Spatial Decomposition"[16, 17] which is a method whereby the volume which the atoms are being modelled in is decomposed over the processors into a three-dimensional mesh of three-dimensional boxes. These boxes contain all the information about both the atoms that a particular processor owns, and a sub-set of the information about some of the atoms in neighbouring boxes that is required to calculate the forces.

[2]LAMMPS web-site: http://www.cs.sandia.gov/~sjplimp/lammps.html

This decomposition has a couple of down-sides. Firstly, it is possible because of the nature of the simulation for load-imbalance to be introduced if the atoms are not evenly dispersed over the space of the simulation and secondly, it is not the most optimal decomposition for performing Fast Fourier Transforms (FFTs) which are (for example) necessary for the PPPM algorithm (described in section 2.3.3). The former of these is only relevant for a sub-set of possible simulations. The latter is solved by re-decomposing the space so that instead of owning a box, each processor owns a whole column of the three-dimensional space. This allows the three-dimensional FFT to be performed as a series of one-dimensional FFTs on a single processor (which saves in what might otherwise be extremely expensive communications costs). After the calculations which require FFTs have been completed, the space is re-decomposed back to the original, mesh configuration.

3.5.2 Neighbour lists

Neighbour lists[16] are a method by which a molecular dynamics code can decrease the communication cost between processors in a parallel molecular dynamics code. Typically, a cut-off radius is defined r_c such that beyond that radius the force between the atoms is close enough to zero that it can be treated as being zero with an acceptable loss of accuracy. A list is then maintained for each atom of every other atom which is inside r_c, and the force terms are evaluated for those atoms. In the case of long distance forces, communication of possitions and velocities of atoms can be passed from neighbouring processor to neighbouring processors repeatedly until all processors have the information they need.

Initially it might seem that calculating the neighbour lists is very expensive, but it need not be done every time-step if r_c is chosen so that it is less than the distance that an atom can move in t time-steps then the neighbour lists only need to be update every t time-steps.

3.5.3 PPPM

PPPM or "Particle-Particle Particle-Mesh"[17] methods calculate the coulomb terms of the force equation by replacing it with a term which calculates the force based on charge density rather than on point charges. This algorithm is used rather than calculating the coulomb forces in the traditional way because it scales as $Nlog(N)$ rather than N^2. The charge density is decomposed spatially into a three-dimensional Grid of equally sized boxes. Each box contains the charge density from the atoms in the box as well as "ghost point" charge densities from the atoms in neighbouring boxes.

The algorithm is described in [17] and works as follows. The first step in the algorithm is to interpolate the charge density based on the point charges of the atoms inside the box. The next step is to exchange the charges of the ghost points with the neighbouring processors. Then the simulation is re-decomposed into a decomposition that is more optimal for FFTs and the entire system is transposed into Fourier space. The algorithm then computes the electric field and transposes back to real space. Finally, the first three steps are reversed with the decomposition being put back to the three-dimensional mesh, each processor updates its ghost points from its neighbours and then finally the forces on the atoms are interpolated from the electric field.

Because this method requires the data decomposition to be transposed multiple times (so that processors have all the appropriate data to perform the one-dimensional FFTs required to perform the three-dimensional FFTs, it has a relatively high communication cost but this overwhelmed by the computational cost of performing the FFTS (see [17] for more details).

3.5.4 rESPA

The rESPA algorithm[17] in LAMMPS allows the different terms in the force equation to be integrated at different levels of granularity, allowing the computational cost to be decreased for the terms where it will not have an unacceptable affect on accuracy while keeping it at the appropriate level for those where it will. This is achieved by having the terms which have to be evaluated a greater number of times calculated repeatedly in "sub-cycles" inside one cycle of the main algorithm.

3.5.5 File I/O

From looking at the freely available source code, it is clear that when LAMMPS performs file I/O, it makes no use of the parallel routines available in the MPI 2 standard[2]. This means that it is more portable (and also avoids problems to do with the file formats with MPI-IO like its reliance on binary output), but potentially adds a bottle-neck in simulations that perform a lot of heavy I/O, and particularly those that write restart files (files that contain a complete dump of the simulation so that if the code is terminated abnormally, the simulation can be restarted from the restart file saving time).

The algorithm used by both versions of LAMMPS works as follows. Firstly, processor 0 writes the file header information. It creates a buffer in memory with enough space to hold the largest number of atoms on any one processor, and copies its data into this buffer, before writing it out to disk. Then it polls each of the other processors in turn, receiving their data into the buffer and then writing out to disk before finally closing the file. This means that the most memory that this operation uses on processor 0 is two times the amount required for the largest number of atoms on any one processor, which is helpful when running on systems like the eServer Blue Gene system where the memory available per processor may be relatively small (in this case either 512 megabytes or 256 megabytes depending on the mode that the machine is running in). However, this code does not scale with the number of processors being used because the amount of time taken for the actual writing/reading of the data will always be the same (the other processors are doing nothing while this is going on) and there will be an additional communications cost.

3.5.6 SHAKE

SHAKE is a method used to improve the accuracy of molecular dynamics by constraining bonds to a particular rigid length. This allows the user to set up simulations with a larger time-step without loss of accuracy. This is possible with protein systems because changing bond length is not thought to be part of biological processes.

3.5.7 Langevin temperature control

Langevin temperature control [30] maintains the temperature of a simulation by coupling it to a (virtual) external heat bath. Molecules at the edges of the simulation have their velocities adjusted as if they are interacting with an infinite bath of some solution surrounding the simulation.

3.6 NAMD

The NAMD[33][3,4] molecular dynamics code is also widely used in the CCS and worldwide to study a variety of systems. It differs from LAMMPS in a number of ways.

Like LAMMPS, NAMD is available for free (although under a more restrictive license than the GPL) from the NAMD web-site, in both source and pre-built binary forms.

Unlike LAMMPS (which is written in C++/MPI making it relatively portable), NAMD is written in a little-used language called "Charm++" (also developed at the University of Illinois). This language is a rival to MPI and was an attempt to build an object orientated parallel extension to C++ with a number of advanced features such as automatic load balancing. It failed to achieve large-scale adoption and therefore is not generally available on any resource where a user wishes to use NAMD. It is therefore

[3]NAMD web-site: http://www.ks.uiuc.edu/Research/namd/

[4]NAMD was developed by the Theoretical and Computational Biophysics Group in the Beckman Institute for Advanced Science and Technology at the University of Illinois at Urbana-Champaign.

necessary (as part of the installation process for NAMD) to port Charm++ to that platform. A version of Charm++ is provided with source releases of NAMD and this version is guaranteed to work with that version of NAMD. Like LAMMPS, NAMD uses FFTW to perform Fourier transforms. Installation of NAMD from source is generally non-trivial making it more difficult to use than LAMMPS.

NAMD generally seems to perform better than LAMMPS, ranging between about 8 and 1.5 times faster depending on the machine the code is running on and how well Charm++ works on that system as well as other unknown factors. It also has support for more features than LAMMPS (although the list of features supported by LAMMPS is growing all the time) and the combination of performance and features make it popular with computational chemists.

3.7 Tools for calculating the Binding Free Energy (MM-PBSA and NMODE)

A potential metric for determining resistance to a particular drug is the binding affinity[34, 35, 36] (i.e. the change in the free energy of the system caused by the binding of the two molecules) between the drug and the protein it binds to (in the case of protease inhibitors, HIV-1 protease). The larger the negative change in free energy caused by binding, the less resistant the mutant is to the drug (i.e. the more negative ΔG_b, the more strongly attracted the two molecules are, and the less resistant the protease mutant). Equation 3.1 in section 3.4 shows the change in Gibbs free energy when two molecules are bound together. The MM-PBSA (Molecular Mechanics Poisson-Boltzmann Solvent Accessible Surface Area) and NMODE (Normal Mode) modules from the AMBER molecular dynamics package[19, 20, 21] make it possible to approximate ΔG_b as in equation 3.2. For this study, the General AMBER Force Field[22] is used to parametrise the ligands the standard AMBER forcefield for biorganic systems[23] is used to parametrise the protease.

$$\Delta G_b = \Delta G^{MM}_{vdW} + \Delta G^{MM}_{ele} + \Delta G^{sol}_{pol} + \Delta G^{sol}_{nonpol} - T\Delta S \quad (3.2)$$

ΔG^{MM}_{vdW} is the van der Waals component and ΔG^{MM}_{ele} is the electrostatic component of the free energy difference. ΔG^{sol}_{pol} is the polar component of the solvation free energy and G^{sol}_{nonpol} is the non-polar component. Finally, $-T\Delta S$ is the component from the change in entropy.

The MM-PBSA module approximates the average free energy difference of binding in a solution, providing the $\Delta G^{MM}_{vdW} + \Delta G^{MM}_{ele} + \Delta G^{sol}_{pol} + \Delta G^{sol}_{nonpol}$ contributions in equation 3.2. The NMODE module is used to calculate the change in entropy, $-T\Delta S$. Together, these tools allow an approximation of ΔG_b.

NMODE calculates the entropic contributions by relaxing the system and then calculating the normal modes. It makes the assumption that the motions in the system are built up out of harmonic motions and attempts to break down the quite complex motions into contributions from each of these harmonic motions. This is a reasonable assumption to make in most circumstances when the system is in a fairly stable state. However, when the system is undergoing a major conformational change (for example a flap opening event in HIV-1 protease) this assumption may not be valid.

3.8 FFTW

FFTW[37] is an Open Source Fast Fourier transform library that is required to run both LAMMPS and NAMD (specifically, the older, version 2.1.5 of FFTW is required). Like LAMMPS, it is available under the GPL from the project web-site[5]. It not targeted at a particular platform, but is instead self-optimising. It works by constructing the routines used at run-time from small, pre-compiled sections of code according to a "plan" file which is generated based on benchmarks run on the system at run-time.

[5]FFTW web-site: http://www.fftw.org/

The user can choose how optimal a plan should to be generated (the more optimal the plan, the longer it takes to generate). The algorithms and implementation is described in [37]. Although this library is required for LAMMPS, it is not the focus of work here.

3.9 Replica exchange

Replica Exchange[38, 39] (sometimes referred to as "Parallel Tempering"[40]) is a form of coupled model where instead of linking together different models, multiple copies of the same model are run with a particular physical parameter (usually temperature) varied across the replicas. These simulations are coupled by performing Monte Carlo swaps of the configurations (see figure 3.1) every so often (on a much larger time-scale than individual time-steps). In theory, this allows the simulation at the target value of the physical parameter varied to explore a larger proportion of configuration space that it would normally. It also helps to mitigate the problem of simulations becoming stuck in local energy minima.

The way the selection is done depends on the code and even (in the case of LAMMPS) on the algorithm selected by the user. LAMMPS offers two methods. In the first, pairs are selected as odd and even pairs and in the seconds, random pairs are selected. When a pair of prospective sites are selected, they are subjected to a fitness test to determine whether a swap is valid. The complete test is in two parts. First a "Boltzmann factor" is calculated as below (equation 3.3) for two trajectories with potential energies E_1, E_2, temperatures T_1 and T_2 where k_B is the Boltzmann constant:

$$F_B = (E_1 - E_2) \times (\frac{1}{k_B \times T_1} - \frac{1}{k_B \times T_2}) \tag{3.3}$$

If this factor, F_B is greater than or equal to zero, the swap happens, otherwise it has a chance to swap if a random number selected from the Boltzmann distribution is less than e^{F_B}. This process is illustrated in figure 3.1.

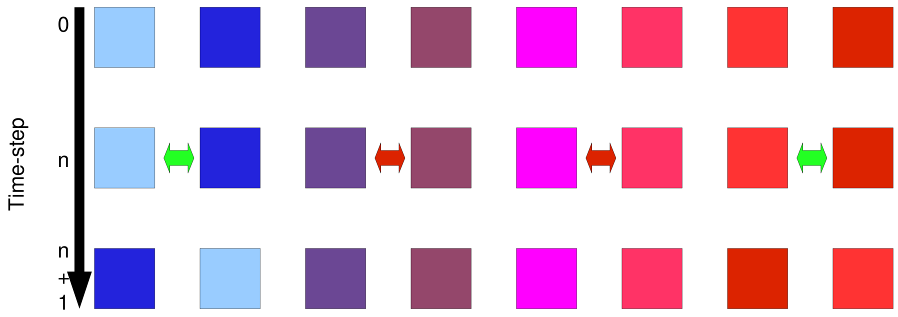


Figure 3.1: This figure shows the process of replica exchange. In the above diagram replicas are coloured based on starting temperature. Every n steps, possible pairs for exchange are selected and a test for compatibility is performed. If this test is passed (as in the two outer pairs) then the configurations are exchanged. If it fails (as in the middle two pairs) those replicas carry on as before. The selection criteria may be defined by the user.

In the realm of meta-computing, Replica Exchange has the added advantage that the amount of communication between replicas is relatively small compared to the amount of communication that occurs inside a replica. This allows the programmer to make use of systems where the communications network is heterogeneous as they can arrange the replicas so that the intra-replica communication only happens on the faster bit of the network while the inter-replica communication (which happens less often) can operate on the slower links. Of course, it is not necessary for the inter-replica communication to take place only on the slower links, only that the intra-replica communication only happens on the fast links. This is the situation that we have with a Grid-based meta-computer, where the communication inside a site is relatively fast (both in terms of bandwidth and latency) and inter-site links (connections which may be travelling across the internet, or else on dedicated links) are considerably slower.

Replica Exchange may be implemented as part of the core code (for example in the case of LAMMPS) but it may also be implemented separately with custom code performing the exchanges and farming out the chunks between exchanges to the main code. This second approach is the one taken by Essex *et. al.*[41] In this case, the performance is significantly reduced as at every exchange the MD code has to write out an entire checkpoint of the simulation (may be many megabytes), the exchange code has to read it in, do the exchange and write out the changed system and the MD code then needs to start up and read in the entire system before continuing. This has to be done for all the replicas. With the approach taken in LAMMPS, the exchange communication happens in memory/interconnect rather than via disk and therefore the cost of swapping is considerably lessened. Instead of writing everything out, replicas which are candidates for swapping only need to exchange potential and kinetic energies (for the closeness calculation) which is considerably less information every at swap.

Chapter 4

HIV-1 Protease

4.1 Introduction

Karplus et. al. [42] gives a thorough overview of the structure of HIV-1 protease. HIV-1 protease is one of three primary proteins (the other two are reverse transcriptase and integrase) that form the replication mechanism of the most common (HIV-1) form of the HIV virus. The importance of this enzyme in viral replication has led to it being the target of nine commercial, Food and Drug Administration[1] (FDA) approved HIV drugs which work by inhibiting its function (referred to as "protease inhibitors"). HIV-1 protease is subject to a large number of mutations, some of which are related to resistance to commonly used protease inhibitors. The interactions between the enzyme, the drug and the protein strands it operates upon need to be well understood in order to aid inhibitor design and to improve patient care. This understanding is hindered by the flexible nature of the protease, and the difficulty in directly observing its behaviour experimentally. As a result computational modeling must be used to provide key insights into the proteases interactions.

Patients are routinely genotyped to determine the mutations present in the various

[1]FDA Web-site: http://www.fda.gov/

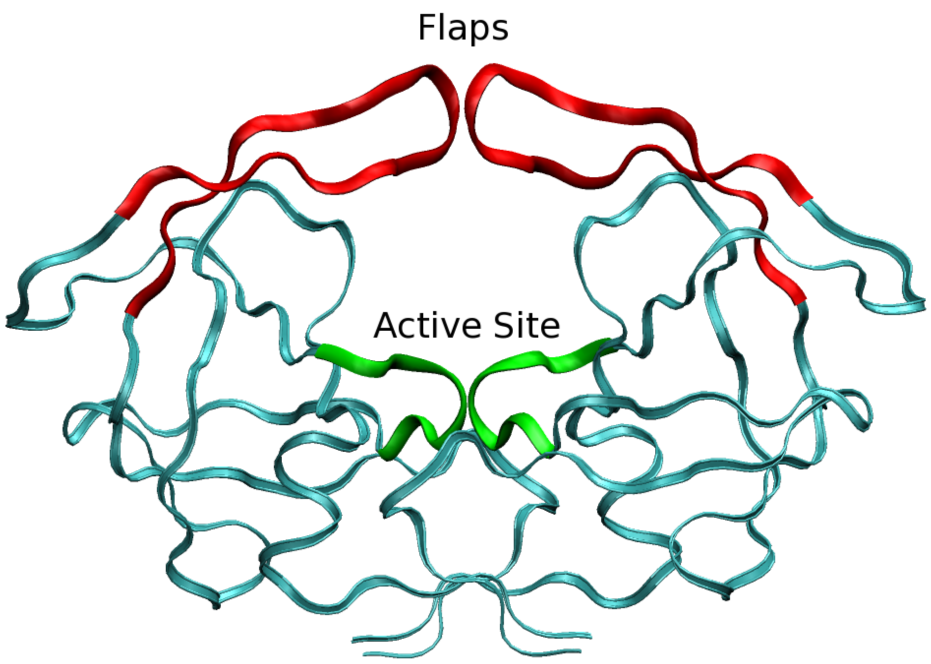


Figure 4.1: HIV-1 Protease with flaps marked in red and active site marked in green. This structure is structure “1HHP” from the RCSB protein data bank, rendered in VMD and annotated later in an image editing package.

strains of the protease present in their bloodstream[43] but this information needs to be interpreted by clinicians in selecting the appropriate drug to prescribe. Clinicians rely on a number of sources to help make this determination including tables of which mutations are generally related to resistance to particular inhibitors and expert-system-style decision support software.

4.2 Structure

HIV-1 protease (figure 4.1) is a homodimer with c_2 symmetry, formed of two identical monomers, each ninety-nine amino acids long. The active site (marked in green in figure 4.1) is gated by a pair of flexible structures (residues 43 to 58 in each monomer, marked in red in figure 4.1) referred to as the “flaps”. These flaps open and close and their configuration at any given time can be categorised as being in one of three loosely

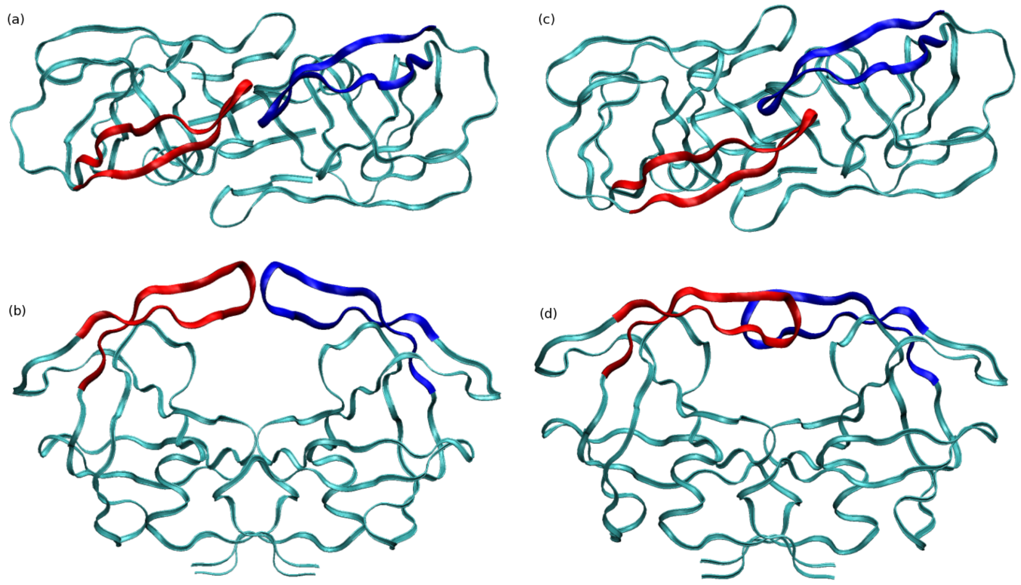


Figure 4.2: Two different structures of HIV-1 protease to indicate the difference in "handedness" between "semi-open" (a,b) and "closed"(c,d) conformations, shown from both "above"(a,c) (showing "handedness") and "in front"(b,d) ("showing openness"). The semi-open structure is 1HHP (from the RCSB Protein Databank) and has a semi-open conformation. The closed structure is 1HVR (from the RCSB Protein Databank), and has a closed conformation. This image was rendered in VMD and composited in an image editing package and is a recreation of Figure 1 in Hornak et. al. [44]

defined states; “open”, “semi-open” and “closed” [44]. Two structures illustrating this from the Research Collaboratory for Structural Bioinformatics (“RCSB”) Protein Data Bank (“PDB”) are shown in figure 4.2. The flaps in figure 4.2 are coloured red and blue to aid in distinguishing them from the rest of the protease. The structure on the left, 1HHP exhibits a semi-open conformation and the structure on the right exhibits a closed conformation. An important feature, is the change in conformational “handedness”[44]. To understand what is meant by handedness, one must imagine the two flaps from above as being a pair of arms with the hands curled slightly and one arm twisted 180 degrees with respect to the other. In the semi-open state, the hands are positioned with the backs of the hands facing each other, while in the closed state the palms face each other. This change in handedness is a good way of characterising the change in flap conformation. The “open” state is not easily observable experimentally and at the time of writing no such crystal structure is available. Some computational effort has been made to investigate flap opening, but since these simulations have not demonstrated the flaps returning to a closed conformation, it is not clear that the structures produced are physical[44].

HIV-1 protease is subject to a number of mutations which are associated with drug resistance, including not only mutations which occur in the flap region (such as M46I/I54V) and active site (such as V82A/I84V) but also in the dimerisation region (such as L10I/L90M). The wide variation in the mutants in the protease means that there is no single “wild-type”, even in a single patient, and most studies select a common mutant to be their wild type for that study. Drug resistant mutations occur in the protease and (often as a side-effect of failure to adhere to the very strict treatment schedule) resistant mutants can become the dominant variant within a particular patient, hindering treatment.

The process by which this occurs is a process of evolution and natural selection. Due to its very nature, the HIV-1 virus mutates relatively often and completely randomly. Some mutations in the protease improve resistance to an inhibitor the patient is being treated with and so the mutants with those mutations manage to reproduce more often than the others and pass on those mutations (along, eventually, with gaining new ones). Some of the mutations that hamper successful protease inhibition also hamper the binding of the protease to the substrate so that while the resistant mutants are resistant to the inhibitor, they are also less successful in reproducing. Further random mutations may decrease this effect and so they will be selected for. It is therefore not only interesting to investigate mutations which directly affect drug resistance but also those which make some resistant strains viable.

4.3 Function

During the replication process, the reverse transcriptase transcribes the virus RNA into DNA and this DNA is integrated into the DNA of the host by the integrase. This combined DNA is then replicated inside the cell and re-transcribed into RNA. The protease cleaves long polyprotein chains (referred to as the "substrate") into much shorter protein chains so that they may be re-assembled into new HIV virii. Without protease function, the replication process does not result in a viable virus halting the spread of infection. Inhibition of this function therefore has become a common method of HIV treatment used by a class of drugs known as "protease inhibitors". These typically function by binding the the HIV-1 protease structure in such a way that the active site is no longer accessible, blocking function. Mutations in the structure of the protease therefore have an effect on the clinical efficacy of numerous common HIV treatments.

Much of the work done on drug design involves performing so-called "docking studies" where the physical matching of potential drugs to a docking site on the protease structure is investigated to see how strongly it binds to the static structure. These studies do not take into account the dynamic nature of the system.

4.4 Flap dynamics

Understanding the dynamics of the flaps is important in fully understanding their function. Unfortunately, the motion is not easily experimentally observable (due to the length scales, and time scales (milliseconds) involved), although it is understood from NMR studies that the structure is flexible[44]. X-ray crystallography studies have provided researchers with a large number (approximately 255 in total at time of writing) of crystal structures of the HIV-1 protease, both un-liganded and bound to the substrate and numerous drugs. Many of these are freely available from the RCSB Protein Data Bank[2] making computational investigation possible.

Mutations occurring in the flap region will have an affect on the behaviour of the flaps, but the flap dynamics may also be affected by mutations elsewhere in the protease, in particular around the active site. Some of these mutations may directly impact drug resistance but others may compensate for a decreased binding affinity to the substrate (and thus the function of the protease) caused by drug resistance-causing mutations.

4.5 Computational limitations

Unfortunately, computational investigation of the behaviour of the protease is not easy either. Investigation with traditional molecular dynamics methods has been hampered by the fact that flap motion is understood to happen at millisecond time-scales while even the longest time-scale molecular dynamics simulations are only run to around a hundred nanoseconds, limited by processor power, wall-clock time and the accuracy of the method (which as with all iterative simulations loses accuracy with time). In addition, molecular dynamics suffers from considerable sampling problems; the results of a single run may not accurately represent the general behaviour of the structure due to the possibility of the simulation exploring an unphysical region of configurational space and becoming trapped in a local energy minima. These limitations have led to a

[2]RCSB Protein Data Bank: http://www.rcsb.org/pdb

number of papers (for example [44]) where the results of a single, long run are published but the usefulness of these results in determining the general behaviour of the protease is not entirely clear.

Other researchers have attempted various methods of speeding up the simulation process. For example, Chia-En Chang et. al.[25] simplify the problem in two ways. Firstly, they course-grain the model by representing each amino acid in the protein chain as a bead linked to each other by simplified bonds. Secondly, they use of Brownian dynamics (i.e. assume that the system is operating in an overdamped regime where overall acceleration is zero) to evaluate the motion of the beads rather than a full force-field molecular dynamics method. Their work shows that some combinations of mutations have a considerable effect (approximately seven-fold) on the time the protease spends with its flaps in the open conformation, confirming that mutations have a considerable affect on flap dynamics.

An alternative approach is to attempt to make it easier for the simulation to explore configurational space. One technique using this idea is called “hyperdynamics”[26] where the energy landscape of the simulation is modified such that energy minima are easier to escape, allowing the simulation to explore configurational space more easily. Because the scope of the modification is known, it is then possible to recover the correct characteristics of the simulation by performing the correct inverse transformation. Replica Exchange (discussed in chapter 3) similarly has the potential to improve the sampling of a simulation, as, like hyperdynamics, it allows the simulation to more easily navigate the complex energy surface of such a flexible molecule. It is even possible to combine these two techniques, with the change in the energy landscape for the hyperdynamics being the variable that is varied across the replicas in replica exchange. Both these techniques avoid problems with long-term accuracy of MD by allowing the simulation to explore greater conformational space in less simulated time. Replica exchange suffers from requiring a very large amount of computational resource, while at the time of the initiation of this study code for hyperdynamics simulations was not generally available. The selected variable for replica exchange will be temperature, with values varying around the target simulation temperature of 300K.

A number of other methods exist. One such approach is "metadynamics"[27] where the energy surface is explored by performing a number of non-equilibrium molecular dynamics runs with a number of different variables constrained harmonically to try and form a function for the free energy in terms of these variables. Another option is "milestoning"[28] where the process of a reaction (and the resulting large movement across the energy surface) is broken down into a series of smaller reactions (and therefore smaller movements) and these smaller movements are modelled and then the results combined to form the path taken by the full reaction.

4.6 Drug interaction

Central to drug design issues is how well a given protease inhibitor binds to a particular mutant of HIV-1 protease. There are six protease inhibitors currently in clinical use; saquinavir, indinavir, ritonavir, lopinavir, amprenavir and nelfinavir. Experimental studies [46] have given some indication of how various mutations affect the binding between these drugs and the protease and intriguingly, that multiple mutations significantly increase the drug resistance of a given mutant.

The Centre for Computational Science has already developed a set of tools called the "Binding Affinity Calculator"[47] (or "BAC") which wrap around existing tools such as Amber, NAMD and VMD allowing the easy construction of equilibrated systems of HIV-1 protease in water bound to either the substrate or a number of drugs from structures in the Protein Data Bank. Amongst its functions is the ability to insert mutations into the structure and investigation of the accuracy of this technique has shown that it produces structures whose properties are very close to those of experimentally derived structures even for large numbers of mutations. Characterising the binding between HIV-1 protease and a drug or substrate is more fully discussed in Chapter 8, along with an in depth computational study of the binding affinity between various mutants of HIV-1 protease with lopinavir.

4.7 Lopinavir

Lopinavir (ABT-378, $C_{37}H_{48}N_4O_5$, DrugBank[3] ID: DB01601) is a protease inhibitor[49], prescribed together with ritonavir (ABT-538, $C_{37}H_{48}N_6O_5S_2$, DrugBank[1] ID: DB00503), marketed under the single brand-name “Kaletra”. The two drugs are prescribed together because the presence of ritonavir helps to increase the level of lopinavir in the blood stream reducing the chance of resistant mutations succeeding. This is important because, without the addition of ritonavir, the levels between doses of lopinavir fall below the level at which protease inhibition is effective[50]. The structure of lopinavir is shown in figure 4.3.

The Ohtaka[46] experimental study suggests that lopinavir is adversely affected by the presence of multiple mutations in the protease.

4.8 Conclusions

HIV-1 protease is an important enzyme in the replication and treatment of HIV. Understanding of its interactions is important both for drug design and improving patient-specific drug selection. Particularly important is understanding the causes of the effects of mutations related to drug resistance and the interaction between mutants and clinically available protease inhibitors.

Computational modelling of the protease should give insight into the behaviour of the flaps although care will need to be taken in finding an approach which gives informative and useful results on feasible time-scales. Replica exchange has been selected as a technique for overcoming these issues due to its availability at the time of project

[3]The DrugBank is a large database of drug information at http://www.drugbank.ca/

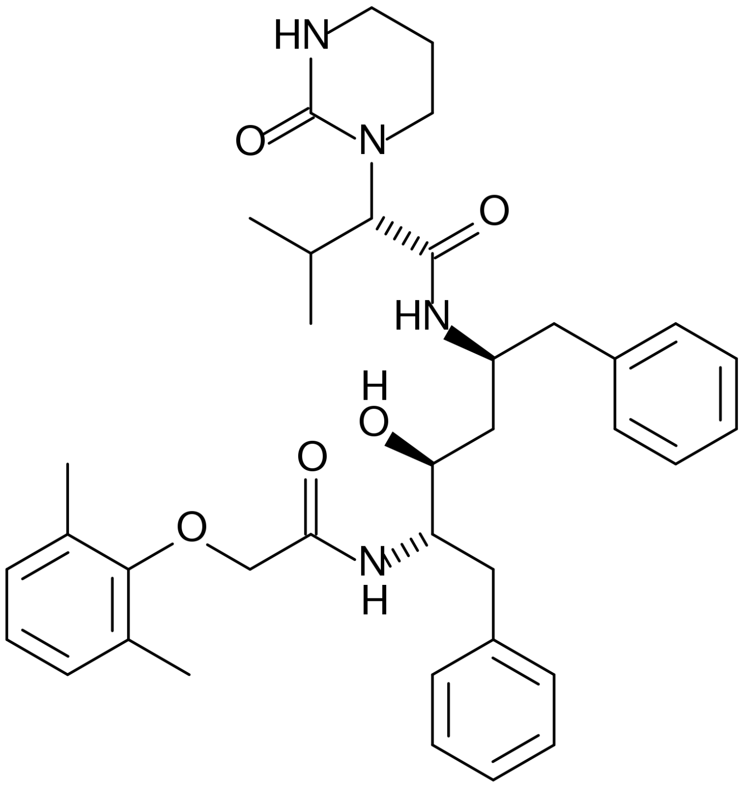


Figure 4.3: Chemical structure of lopinavir.

inception. In addition, it also maps well onto a Grid infrastructure allowing the use of easily available, geographically distributed resources.

Previous experimental work has indicated that lopinavir will be a suitable inhibitor to study in conjunction with the protease.

Chapter 5

Benchmarking Un-modified LAMMPS on the TeraGrid

5.1 Overview

Clearly, there are a number of issues to be faced when running a code across multiple sites on a Grid. The most obvious one is the problem of the "weak link" between the sites. Even if the communication inside a site is normal TCP/IP (and in this case it is not - it is using a version optimised for the cluster's interconnect), then it will still have a lower latency (simply because the boxes are closer together, both in a physical sense and in that there is less network hardware between them) than the communication between sites. It is also likely that there will be a much larger amount of available bandwidth between nodes at the same site than there is between nodes at different sites. Inter-site communications have been shown in other studies[11] on other types of codes to be a bottleneck, and it is likely that they are even more so on molecular dynamics codes which are very communications-heavy. Other studies[10] performed at EPCC between two machines much closer (HPCx which is housed at Daresbury Laboratory in Cheshire and an unnamed machine at Edinburgh, sites which are approximately 200 miles apart) together report the bandwidth to be around 4 megabytes/second and the latency to be around 650 times as large as the latency inside the HPCx system itself. The two TeraGrid sites selected (SDSC and UC/ANL) are an order of magnitude further apart (around 2000 miles). Although the backbone TeraGrid network backbone which links the two sites is reportedly 40 Gb/s, more important (due to communication pattern and frequency) is the latency which is limited by distance.

The two sites selected for this preliminary benchmarking were chosen because they are geographically distant, which represents a worst case scenario for latency and therefore performance.

5.2 High performance computing resources

Both the SDSC and ANL/UC clusters are Linux clusters with Intel IA64 nodes connected to the TeraGrid in the US. The SDSC cluster consists of 262 dual-processor nodes, each with four gigabytes of memory. The interconnect is Myrinet. The ANL/UC cluster has similar hardware but only 62 compute nodes. The SDSC cluster has 1.5 GHz processors while the ANL/UC one has 1.3 GHz processors. On both systems, the “vendor” MPI is a version of MPICH optimised for the Myrinet interconnect.

In addition to the problems due to the increased latency/decreased bandwidth between sites mentioned in the previous section, there were a number of problems due to the fact that the two machines at the two different sites are within different administrational domains. Even though the sites selected have similar hardware and software configurations, libraries are installed in different places, and in particular the libraries and compilers for the improved version of MPICH-G2 that is used here are installed in different locations. If the sites had completely different systems, then it is not clear how to decide what the performance of “one processor” in the resulting heterogeneous meta-computer would be and since the purpose here is benchmarking, the selection of two similar supercomputer systems makes this task easier.

Of the two pre-release versions of the compiler available (a GCC based version and an Intel-based version), the GCC compiler was chosen because initially there were configuration issues with the Intel compiler at one of the sites. This is likely to adversely affect the performance of the codes in the actual computational portion of the code, but not in the communications portions (which will be more affected by latency and bandwidth).

5.3 The benchmarks

LAMMPS comes supplied with a number of benchmarks. One of the benchmarks which has been used previously[29] is the rhodospin benchmark. The base version of this benchmark is a 32,000 atom model of the rhodospin protein. The entire system can be replicated in all three dimensions (creating a system with n proteins within one simulation) allowing the size of the problem to be scaled. It does not perform any file I/O (although it could be easily added) and so this removes one complication.

In all the benchmarks in this chapter, a single trajectory system was used rather than an ensemble. Even cross-site benchmarks use a simulation of a single protein spread across multiple resources. To the LAMMPS code (and indeed any code) MPIg makes these multiple resources appear to be a single resource with only a set of MPIg-specific environment variables distinguishing this configuration from a single resource, to enable a programmer to attempt to mitigate performance problems.

This benchmark was selected because it uses long-range forces and therefore is representative of the other biological systems we wish to study later on. As a result, it makes use of PPPM as described earlier in chapter 2.

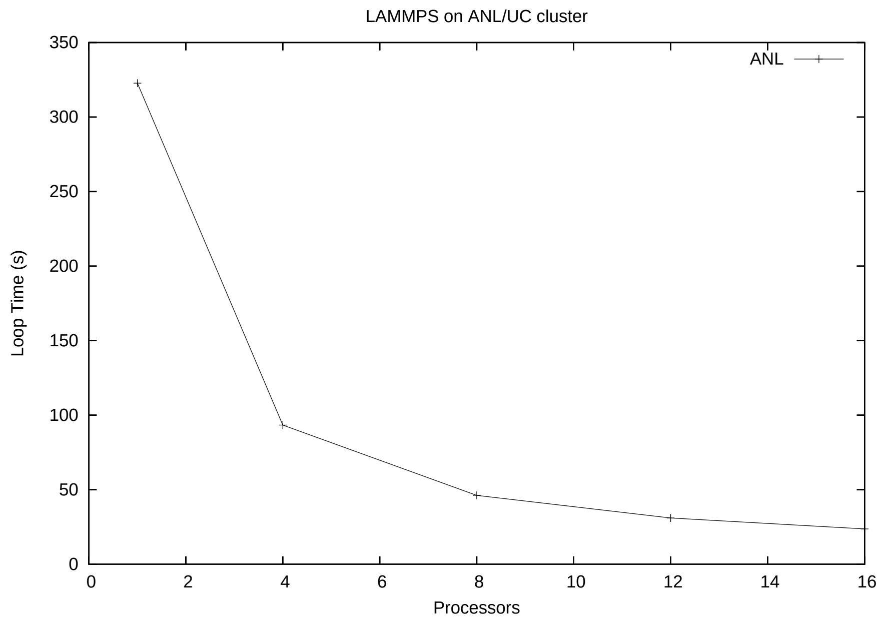


Figure 5.1: Loop time of the 32,000 atom Rhodospin LAMMPS benchmark on ANL/UC cluster.

5.4 Results and analysis

5.4.1 Single site performance

Before looking at how well LAMMPS performs in a cross-site environment, it is necessary to consider how well the code performs on the two systems separately. If the code does not perform well there (unlikely, because LAMMPS has a good track record of scaling well on a variety of systems) then there is little point attempting to get it running across multiple sites.

Figures 5.1 and 5.2 show the loop time of LAMMPS running on the ANL/UC and SDSC systems, and figures 5.3 and 5.4 show the scaling of the same. As can been seen from these graphs, the code scales fairly well overall. There is not a great deal of difference in performance between the two systems, despite the SDSC cluster having slightly faster processors.

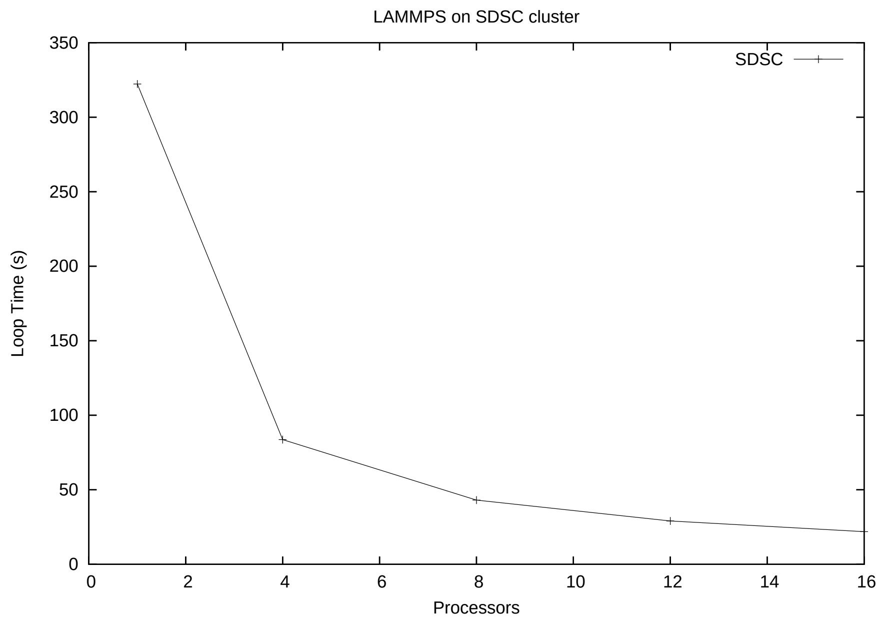


Figure 5.2: Loop time of the 32,000 atom Rhodospin LAMMPS benchmark on SDSC cluster.

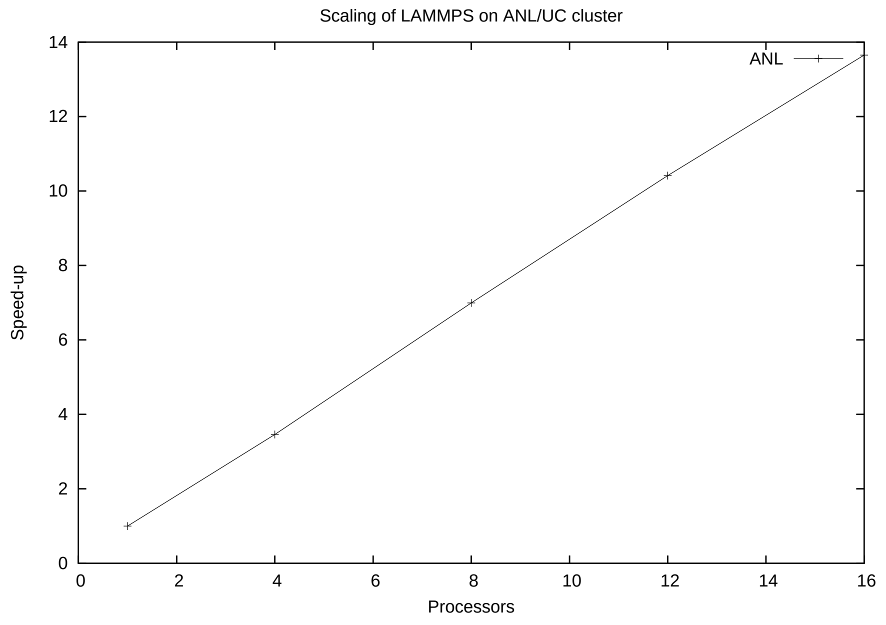


Figure 5.3: Scaling of the 32,000 atom Rhodospin LAMMPS benchmark on ANL/UC cluster.

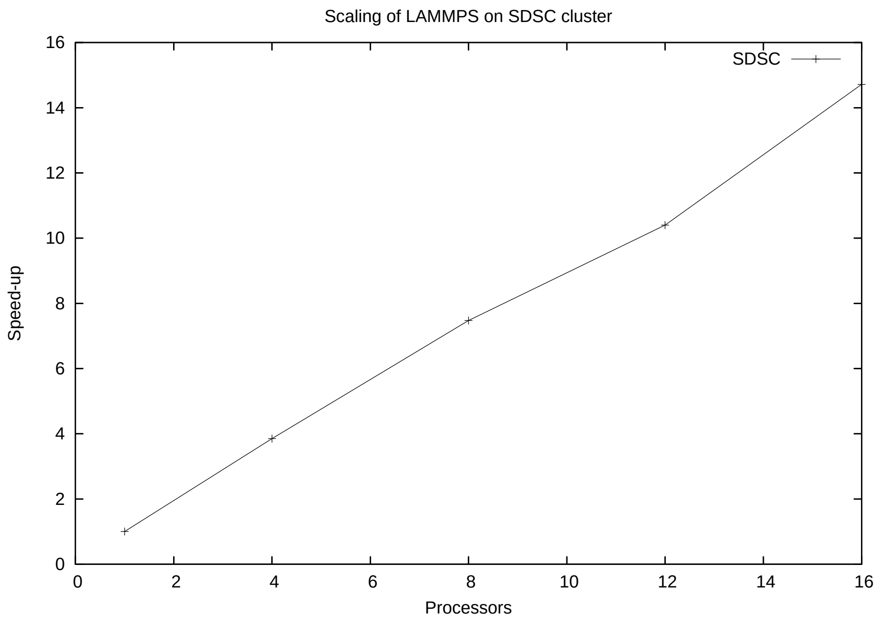


Figure 5.4: Scaling of the 32,000 atom Rhodospin LAMMPS benchmark on SDSC cluster.

The scaling curves in particular look very promising with the ANL/UC system being almost linear and with the SDSC system coming close to that. These results show that the code both performs and scales well on these systems and so it may be worth attempting to get it running in a cross-site environment.

5.4.2 Cross-site performance

Figure 5.5 shows a graph of performance of cross-site runs using MPICH-G2 between ANL/UC and SDSC compared with runs on those machines. The cross-site runs are split evenly between the two machines, so (for example) the sixteen processor run has eight processors on the SDSC system and eight on the ANL/UC system. The two job runs are the same except that both jobs are on one system. This gives an indication of the overhead from using the Globus device rather than the normal MPI. The single job runs are just using the vendor MPI library for the system. The performance for

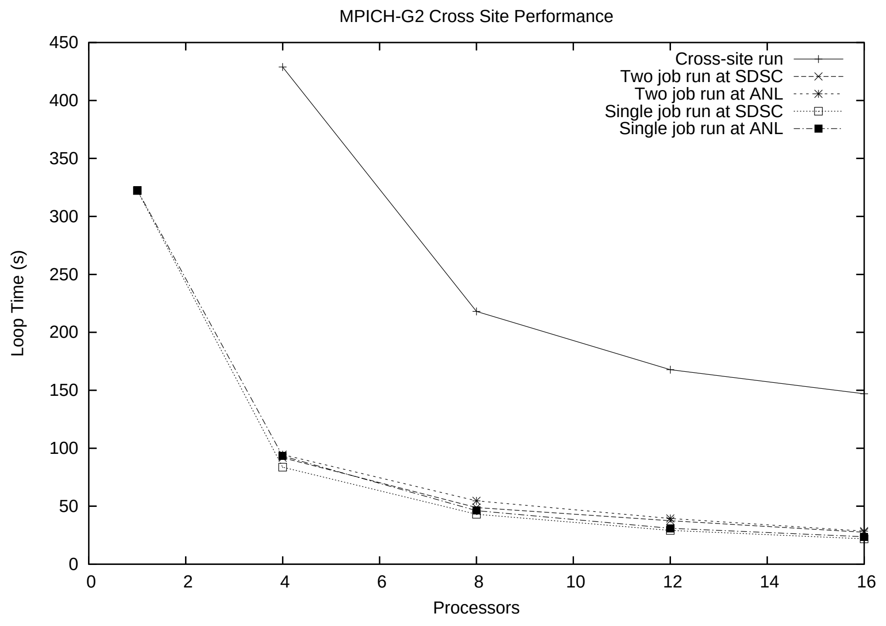


Figure 5.5: Loop time of Rhodospin LAMMPS benchmark run with a single simulation spread across the SDSC and ANL/UC Teragrid clusters, compared with the single-site performance at the two sites and the single-site, two-job performance at the two sites. The difference between the latter shows the performance loss of splitting the simulation into two Globus sub-jobs in comparison to the single job, which is negligible compared to the difference between the sites.

the cross-site runs seems to be comparatively poor with runs taking approximately five times as long as the comparable runs on one system.

Figure 5.6 shows the scaling of the various different types of run, as calculated from the run-time of a single processor job on the relevant machine. In the case of the cross-site run, deciding what the performance of "one processor" is is problematic. There are two obvious options. The first is to make the assumption that the code scales perfectly to the smallest number of processors that was actually run (in this case four). This might be an acceptable assumption on a single machine, but given the nature of this system, it is entirely possible that it will give a vastly inflated estimate of the time taken by one processor. The second option is to take an average of single processor runs on the two different systems. This is in reality more accurate as what we are interested in here is

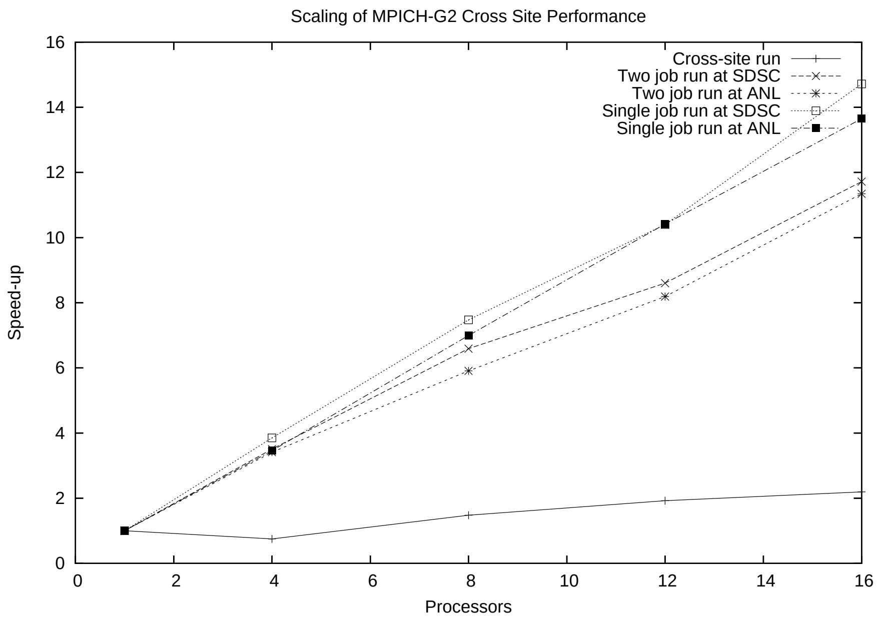


Figure 5.6: Scaling of Rhodospin LAMMPS benchmark run with a single simulation spread across the SDSC and ANL/UC Teragrid clusters, compared with the single-site performance at the two sites and the single-site, two-job performance at the two sites. The difference between the latter shows the performance loss of splitting the simulation into two Globus sub-jobs in comparison to the single job, which is negligible compared to the difference between the sites.

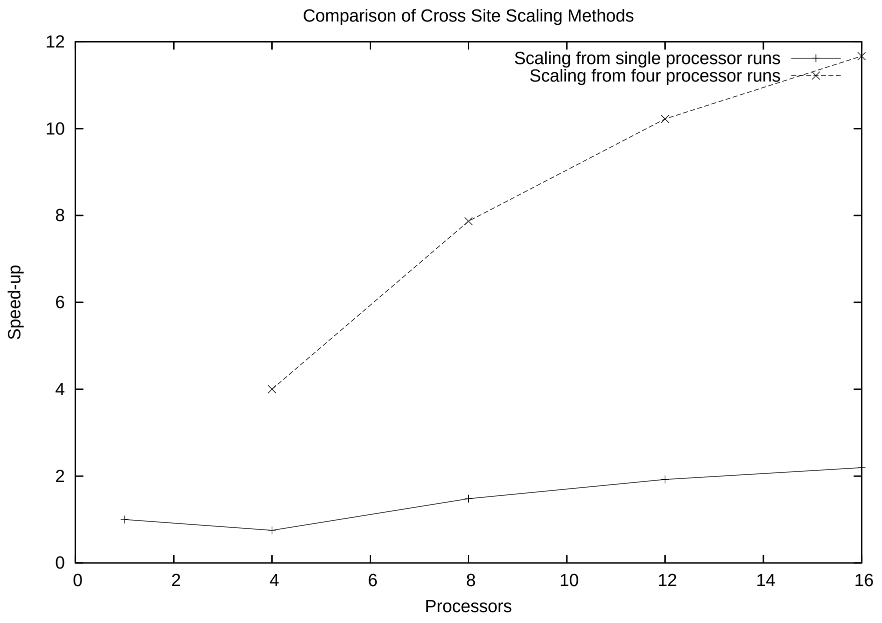


Figure 5.7: Comparison of the scaling curves obtained using two different methods of estimating the speed of one processor for cross-site runs. As can be seen, estimating the performance by dividing the smallest possible cross-site run by the number of processors gives a false scaling curve because the loop time of a cross-site run on four processors spread between two sites is considerably larger than a single-processor run at a single site. This means that the most effective way of calculating single processor performance is a mean of the single-processor performances at each site.

how the performance of a cross-site run compares to that of a single machine. It also gives a worst possible result for the cross-site run's performance. The latter of the two methods has been chosen for figure 5.6.

In order to show just how much these results differ, figure 5.7 contrasts the two. From this we can see that while the first method gives a rather good scaling curve (almost comparable with the single machine runs), the second shows that running things in a cross-site mode is actually worse on four processors than on only one processor on a single machine.

Looking at figure 5.6, the scaling is poor in comparison to single machine runs, and it is rather unlikely that there is a point at which MPICH-G2 runs over-take runs on a single machine. What is also interesting, looking at these results, is that the single machine two-job runs are indeed scaling somewhat slower than the single job ones. This is to be expected given that some of the communication is going over the "slower" globus device, but it does reinforce how much better the dedicated MPI back-end is.

Because the Rhodospin benchmark is designed to be easily scaled, it is possible to generate benchmark figures for larger systems from it. Replicating the benchmark by a factor of two in the x direction creates a benchmark with 64,000 atoms, twice the previous number. Figure 5.8 shows a comparison between a cross-site run on SDSC and ANL/UC and a normal run on SDSC with such a system.

Figure 5.9 shows the scaling curve of the larger benchmark. Although at first glance it looks to be as poor as the one for the original system, the scaling is very slightly better. This is most noticeable on these graphs when comparing the speed-up from sixteen processors. It seems likely from this that as the system gets larger, the scaling will get better. While it seems possible that for a suitably large system the scaling will start to approach that of the conventional MPI runs it seems likely that the system would have to be extremely large. It is also likely that the cross-site runs will never over-take the conventional MPI runs because it is also clear from these figures that the scaling of those is improving too and they are faster to start with.

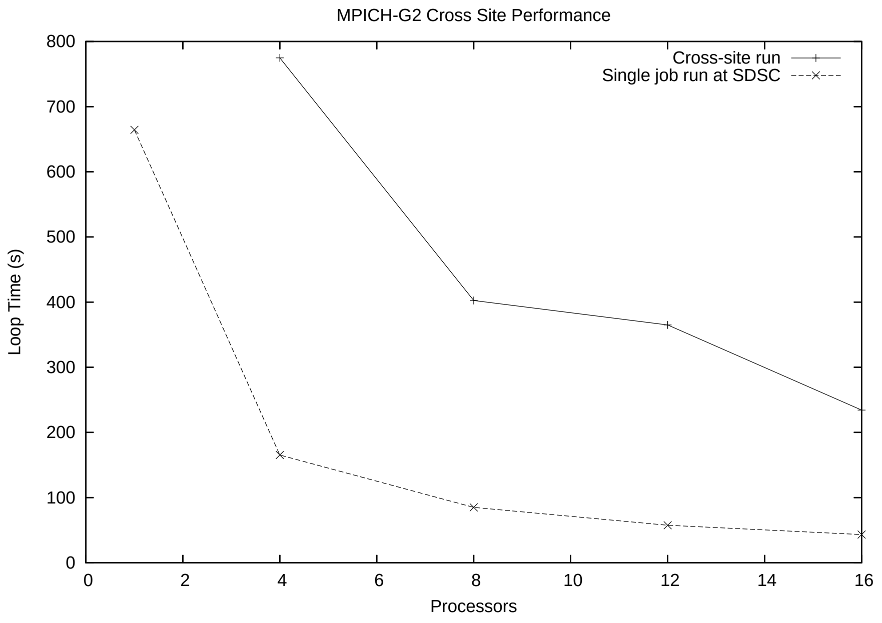


Figure 5.8: Loop time of replicated (64,000 atom) Rhodospin LAMMPS benchmark run with a single simulation spread across the SDSC and ANL/UC Teragrid clusters, compared with the single-site performance at the two sites and the single-site, two-job performance at the two sites. The difference between the latter shows the performance loss of splitting the simulation into two Globus sub-jobs in comparison to the single job, which is negligible compared to the difference between the sites.

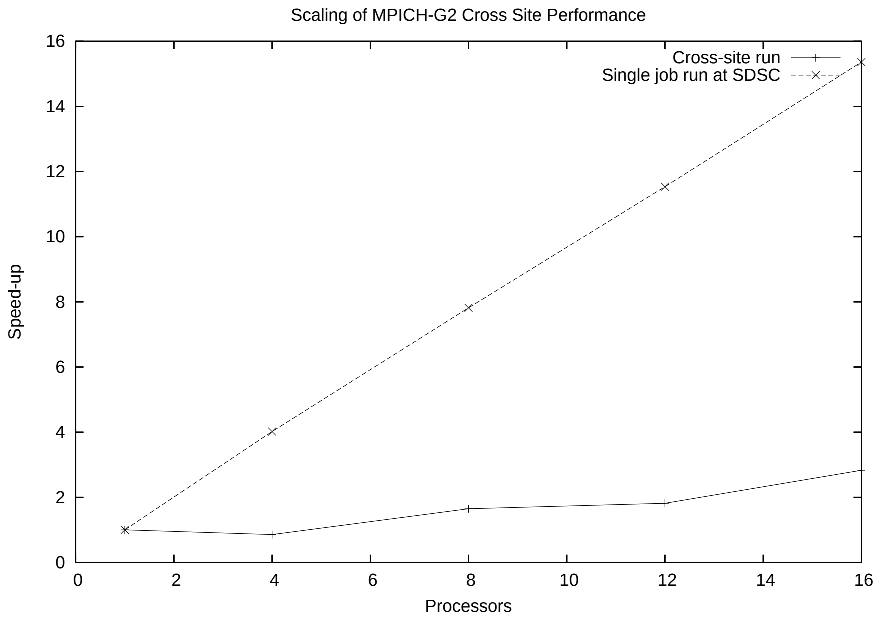


Figure 5.9: Scaling of replicated (64,000 atom) Rhodospin LAMMPS benchmark run with a single simulation spread across the SDSC and ANL/UC Teragrid clusters, compared with the single-site performance at the two sites and the single-site, two-job performance at the two sites. The difference between the latter shows the performance loss of splitting the simulation into two Globus sub-jobs in comparison to the single job, which is negligible compared to the difference between the sites.

There are a number possible explanations for the poor performance. This particular benchmark might, in this case be suffering from being bandwidth-limited (there might be an issue with the total network bandwidth between the two sites). It might also be suffering with problems due to the vastly increased latency between the two sites. It is likely that with two sites which aren't as far apart geographically that it will perform better, somewhere between the performance shown here and the performance on one machine but with two separate Globus jobs. The selection of the two sites was deliberate, in order to produce a set of results that were close to the worst possible, and while it is possible that a user might want to run between two sites this far apart, doing so is never going to be optimal.

Chapter 6

Modifications to LAMMPS for Replica Exchange on a Grid

6.1 Introduction

As described in chapter 3, section 3.9, replica exchange explores configuration space by running multiple replicas of a system with changes to a particular environmental variable (usually temperature) and then randomly exchanging configurations regularly. The results in chapter 5 indicate that running LAMMPS across a Grid as a single monolithic molecular dynamics simulation is unlikely to ever be an efficient use of the available resources. Therefore, it is necessary to explore other methods which decrease the performance loss due to slow inter-site communication, either by hiding it (overlapping inter-site communication with some other process) or by limiting inter-site communication. Replica exchange has the potential to be a successful example of the latter option, because it is possible to set up the simulation in such a way that each replica has all its cores at one site. Then the only communication between sites will be the replica exchange itself which happens infrequently relative to the intra-replica (and therefore intra-site) communication which happens every time step. Figure 6.1 gives

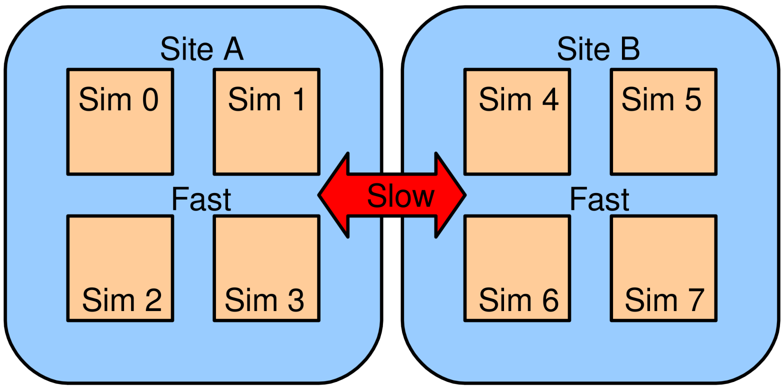


Figure 6.1: Division of cores across two sites (A and B) with an arbitrary replica exchange molecular dynamics code (for our purposes this will be LAMMPS), modified to be aware of the topology discovery mechanisms available to the communications library (in this instance MPIg).

an example of how eight simulations may be allocated over two sites so that they only communicate over the slow inter-site link at the replica exchange steps.

6.2 Algorithm

The algorithm chosen (shown in figure 6.2) is relatively basic, splitting the available cores into pools based on their colour at the vendor MPI layer (see chapter 2, section 2.3 and figure 2.2 for definitions and examples) and then simply allocating the available cores to simulations in chunks on a first come/first served basis from each of those pools in turn as they run out of cores. Table 6.1 shows an example allocation of sixteen cores over four simulations at two sites based on MPI rank.

This algorithm's implementation is simple in that it does not try to cope with systems where replicas have different numbers of cores (as supported by LAMMPS) but this is

Core Rank	Site	Simulation	Rank in Simulation
0	A	0	0
1	A	0	1
2	A	0	2
3	A	0	3
4	A	1	0
5	A	1	1
6	A	1	2
7	A	1	3
8	B	2	0
9	B	2	1
10	B	2	2
11	B	2	3
12	B	3	0
13	B	3	1
14	B	3	2
15	B	3	3

Table 6.1: Core allocation for a small demonstrative test case as allocated by the topology aware version of LAMMPS. Note change in rank for each core within simulation. This means that depending on the MPI communicator used, cores have different rank, and within a simulation, the molecular dynamics communication code is exactly the same as a normal, single simulation version of LAMMPS.

```
1) Make sure that system is suitable for re-allocation.  If it is not, print an error.
2) Set up arrays:
     Create a mask array (an array which shows whether or not a core has been
     allocated).
     Create an array of colours of cores at vendor MPI level.
     Create an array of processor mapping where mapping[i] is the simulation core[i] is
     part of.
     Create an array with the number of processors at each site.
3) Convert array of colours into an array of processors at each site.
4) Allocate processors to simulations:
     Select the first site.  When cores at current site = 0, move on to next site.
     For every simulation (s):
       Scan through the cores.
       Until a simulation has all the necessary cores, if a core is available at the
       current site, allocate it to current simulation:
         Set mapping to simulation number.
         Set mask of core to 1 (true).
         Decrement number of cores available at site.
5) Transfer the mapping into the LAMMPS "iworld".
```

Figure 6.2: Pseudo-code algorithm for processor re-allocation in modified, MPIg topology-aware version of LAMMPS.

not a situation that would normally be encountered when performing replica exchange. It may however become the case if the number of processors available changes, either during a run, or between a run being terminated and being restarted. It is also unable to deal with a situation where the number of cores in each simulation does not exactly divide the number of available cores, but this situation is caught by error handling in LAMMPS anyway (giving an error about an inconsistent number of cores/worlds). This code could be improved in future to use uneven numbers of cores in replicas in order to avoid cross-site communications in more distributions of cores between sites.

6.3 Implementation

LAMMPS already supports the concept of running multiple replicas of a system by partitioning the cores into separate MPI worlds as specified by a command-line option, as well as support for replica exchange. For the purposes of this study, it is necessary to modify the code such that when these cores are allocated to replicas, they are allocated so that each replica wholly resides at a particular site. In this way, the communications that occur between sites over the relatively "slow" inter-site links are limited to the replica exchange steps. This was done by modifying the file "universe.cpp" which handles the creation of all the worlds when the code starts up. No other modifications were required.

At this point in the project, the successor to MPICH-G2, named "MPIg" became available. Both MPICH-G2 and MPIg share the same topology discovery mechanism (with a slight change in variable names) which allow a code running on them to determine which cores are at a particular site (the cores at a particular site should all be able to communicate by the vendor MPI layer). Since MPIg seemed likely to replace MPICH-G2 across the TeraGrid and was to be deployed on the National Grid Service resources, developing the modified version of LAMMPS to run on MPIg, rather than MPICH-G2 seemed a sensible course of action.

LAMMPS adds partitions as it parses the command-line in a sequential fashion, as opposed to splitting up the cores all at once. As a result, it is not trivial to allocate partitions to sites as partitions are added. Instead, the code has been modified so that at the end of the allocation phase, it checks to see if the configuration is suitable to be re-allocated to specific sites, and then if possible does so.

6.4 Benchmarking

The aim of this series of benchmarks is to show that performing replica exchange simulations with LAMMPS scales nearly as well cross-site as it does at a single site. Unfortunately, this does not allow a perfect comparison with earlier work as that is running LAMMPS differently and also uses a different system as the previous was performed before the decision to investigate HIV-1 protease was made.

Benchmarks were run on two different general purpose Grids: the National Grid Service (NGS) in the UK, and the TeraGrid in the US. In order to guarantee proper execution, HARC reservations were used. In the US, the SDSC and NCSA IA-64 clusters were selected for their similar architectures. In the UK, the Leeds, Manchester and Oxford x86-64 clusters were selected. The modified version of LAMMPS was deployed at all five sites. Due to the number of sites selected, three site runs were performed on the NGS with two site and single site runs performed on both Grids. Attempts were made at cross-Atlantic runs using pairs of machines on either side, but these were unsuccessful due to configuration problems.

The two TeraGrid sites (NCSA and SDSC) are connected by the shared 10Gb network infrastructure. The internal interconnects are Myrinet. On the National Grid Service, the Manchester and Oxford sites are connected by a lightpath. Other intersite communication occurs over the normal JANET[1] network.

For benchmarking purposes on the NGS, a system of six replicas was selected (a number

[1]http://www.ja.net/

that can be divided by three and two, but few enough that low core counts can finish in a timely fashion) over a temperature range of 295K to 320K with an unliganded HIV-1 protease in water. In an attempt to increase the negative effects of inter-site communication, the swapping frequency was increased ten-fold to every 200 femtoseconds. On the TeraGrid, a system with four replicas was used.

In both sets of benchmarks, HARC was used to perform user-defined co-reservations for all cross-site runs (but not single-site runs), ensuring that all of the sites started the sub-jobs at the appropriate time. All benchmarks were run with the MPIg modified version of the code.

A network benchmark was also performed on the National Grid service machines between Manchester and Leeds to investigate the performance characteristics of this network. The latency between two sites can be measured in two ways. The first is to use the UNIX `ping` command which sends packets to a particular IP and times the response. This method is a useful first step but may not give a sufficiently accurate measure for cases where the inter-site traffic is routed down a lightpath, or where internet traffic to or from compute nodes communications are routed through additional hardware which traffic to or from the login nodes is not, distorting the times measured. It was therefore decided to develop a short code in C with MPI which passes messages between two cores and measures the time (using `MPI_Wtime`) that these operations take to complete. This code can also measure bandwidth by using it to pass a single, large message between two cores. This same code can be run locally to measure the interconnect performance between the two sites and a comparison made. When measuring latency, the code passes 1 million integers between the two cores as individual messages, and when measuring bandwidth 1 million integers are sent as a single message. Knowing the size of an integer on a particular platform it is then possible to calculate the bandwidth from the times measured. This code was run both between nodes and inside multi-processor nodes at a given site to give an overview of the hierarchical nature of the modern cluster.

6.5 Results and analysis

Site(s)	Ping	Latency	Bandwidth
Leeds (intra-node)	n/a	3.1 μs	790 megabytes/s
Leeds (intra-site)	n/a	26 μs	201 megabytes/s
Manchester (intra-node)	n/a	2.9 μs	745 megabytes/s
Manchester (intra-site)	n/a	29 μs	190 megabytes/s
Cross-site	2.7ms	2.6ms	45 megabytes/s

Table 6.2: Network characteristics of NGS. Shown are average (over ten) `ping` times between login nodes, latency inside and between compute nodes as measured by the interconnect test code and bandwidth inside and between compute nodes as measured by the interconnect test code.

The network benchmarking code was run locally at the two sites on the NGS (Leeds and Manchester) and between those two sites. The results are shown in table 6.2. For comparison, the average timings from ten `ping` packets between the login nodes are also shown. These results show that the Manchester and Leeds clusters have similar performance characteristics. For the more geographically displaced TeraGrid sites, the disparity in latencies and bandwidth between inter-site and intra-site communications will be far greater.

The cross-site network performance is considerably lower than the intra-site network performance, with latencies approximately two orders of magnitude larger and approximately one quarter of the bandwidth. This latency overhead will have a significant impact on any code which heavy communications patterns (such as molecular dynamics code like LAMMPS), particularly if the code's developer makes little or no effort to hide the latency of communications. Percentage-wise, the bandwidth cost is lower .

Figure 6.3 shows the loop time of the modified version of LAMMPS run cross-site between SDSC and NCSA, compared with the loop time at SDSC. As can be seen, from this graph, the performance of the cross-site run closely tracks that of the single-site run (although there is still a very small decrease in performance) and the performance in real time from running LAMMPS like this over a Grid is very much better than the original method. Figure 6.4 compares the speed-up from a particular number of cores at SDSC obtained by doubling that number of cores at SDSC and the speed-up obtained by adding the same number of cores at NCSA. The ideal shape of this graph is of course a horizontal line at 2x speed-up. As can be seen in this graph, the speedup obtained by adding a second site (NCSA) closely tracks that of adding more cores at SDSC.

Figure 6.5 shows the loop time vs. core count for a similar system with six replicas when run on the National Grid Service at a single site (Manchester), across two sites (Manchester and Leeds) and across three sites (Manchester, Leeds and Oxford). As with the TeraGrid, the cross-site performance closely tracks the performance of a single site, even when extended to three sites.

In fact, this performance should not be surprising. In essence, what has been achieved by replica exchange is an avoidance of cross-site communication by making it so infrequent that it has barely any effect on the run-time and so the performance of a replica exchange code is close to that of an individual simulation, i.e. in the case where we have 128 cores and eight 16 core simulations and we double the number of cores, the code should scale close to the move from 16 to 32 cores rather than 128 to 256 which means it should scale better than normal. In addition, accuracy may be improved by adding extra replicas (and therefore enhancing sampling) with very little performance loss if a particular simulation only scales to a small number of cores.

The cross-site performance of this modified code is very good compared to the original method of performing cross-site runs in LAMMPS. This is due to the replica exchange method decreasing the cross-site communication performed to a level where it has relatively little effect on the performance of the code. This implies that this method of running the code is likely to be more useful to researchers when running large simulations on geographically disparate resources on a Grid.

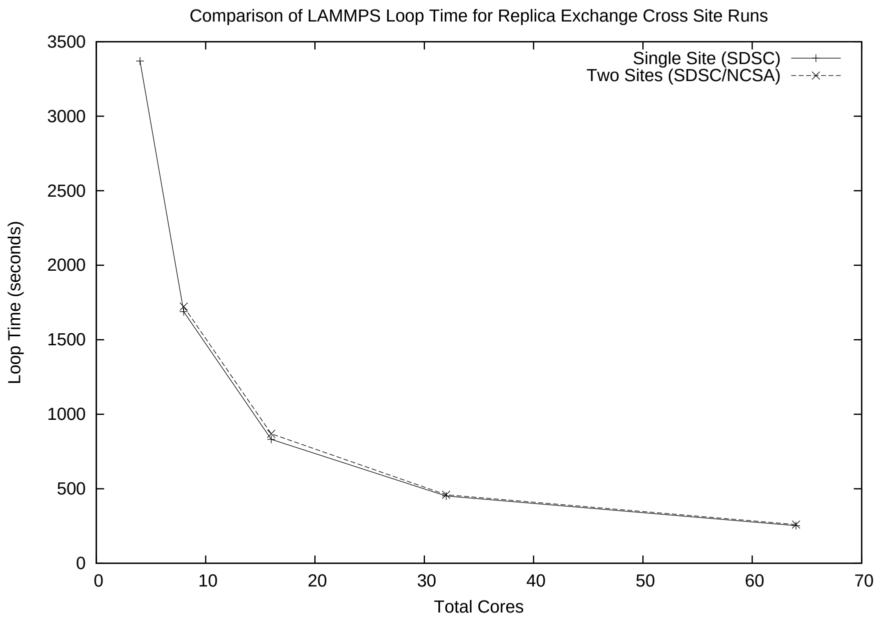


Figure 6.3: Comparison of the loop time vs. core count of the LAMMPS(MPIg-modified) replica exchange test system run cross-site between SDSC and NCSA with the same system run only at SDSC. Cross-site runs were performed using reservations made by the HARC co-scheduler.

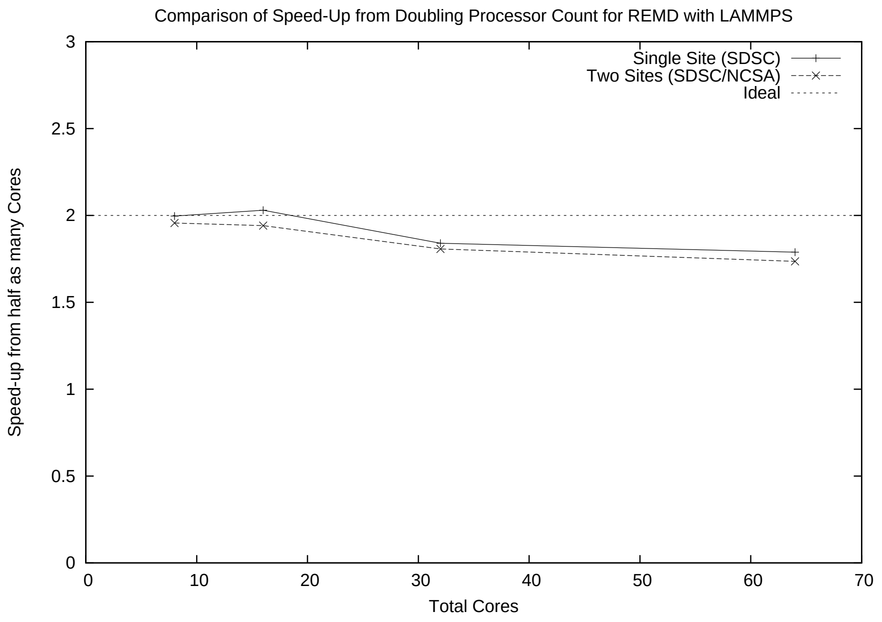


Figure 6.4: Speed-up observed on TeraGrid when doubling the number of cores of a job at SDSC by adding more cores at SDSC and by adding the same number of cores at NCSA. The speed-up is given as Loop Time(N)/Loop Time(N/2).

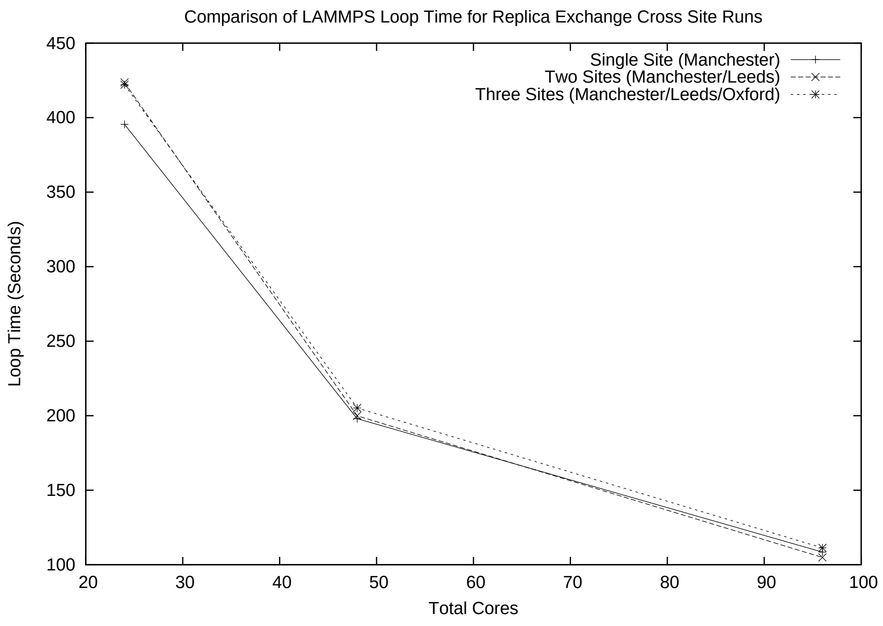


Figure 6.5: Comparison of the loop time of the LAMMPS(MPIg-modified) replica exchange test system with total core count when run cross-site between Manchester, Leeds and Oxford (three sites) and Manchester and Leeds (two sites) with the same system run only at Manchester. Cross-site runs were performed using reservations made by the HARC co-scheduler.

Chapter 7

Replica Exchange Molecular Dynamics with HIV-1 Protease

7.1 Introduction

There are number of issues with investigating the flap dynamics of HIV-1 protease computationally by traditional single trajectory long time-scale methods. For a start, the flap motion itself occurs over time-scales considerably longer than it is traditionally practical to run a molecular dynamics simulation (a very long scale molecular dynamics simulation is at most hundreds of nanoseconds of simulated time, limited both by available real-world time and by decreasing accuracy) meaning that the chance of observing rare event such as an interesting flap motion is very low. In addition, the flexible nature of the protease means that the configurational space is extremely large and so exploring it properly is problematic. Ensemble methods enhance the sampling of the system (and therefore the chance of observing rare events) but consume additional computational resource linearly with the number of replicas in the ensemble.

Replica exchange molecular dynamics is a suitable approach for investigating the flap dynamics of HIV-1 protease because it increases the chance of a flap event being sampled, whilst hopefully allowing for accurate modeling. Unfortunately it is extremely resource intensive with many full molecular dynamics simulations all running concurrently. This means that until recently it has only been used in bio-chemistry to study very short proteins. With the availability of the Grid and new, extremely large computational resources, the possibility of studying the protease in this way becomes available.

7.2 Method

7.2.1 System

The structure selected for this study was a non-liganded 1HHP crystal structure from the Protein Data Bank in water. The replicas are constrained to a particular temperature by being coupled with a Langevin bath, and constrained to a particular pressure (1 atmosphere) using a Nosé/Hoover [31, 32]barostat. SHAKE (see section 3.5.6) is used to keep the bonds to the hydrogen atoms at a constant length. The timestep used was 2 femtoseconds, and exchanges were tried every 1000 time-steps. The pair selection process for attempting replica exchange steps was initially pair-wise, and after the investigation of swapping behaviour in 7.2.4 this was retained as random pair selection selection did not not significantly improve the probability of swapping occurring.

Simple benchmarks running a single replica for a short period of time to determine the computer time required to run a nanosecond of simulation time were performed. Due to computational constraints and the results of these preliminary benchmarks and those in chapter 6, it was initially decided to run thirteen replicas over the temperature range 290K to 350K, each with twenty-four processors. As more computational resources became available, this was adjusted to suit the available resources. Eventually it was extended to sixty replicas of thirty-two processors each.

7.2.2 Equilibration

The system was equilibrated using existing tools for equilibration of HIV-1 protease systems as described in chapter 8. In addition, each replica was then heated to its selected temperature over a period of 20 picoseconds. This will be referred to henceforth as the "pre-heat" stage.

7.2.3 Computational resources

Initial estimates of the necessary run time for the simulation is a total of twenty-six hours of wall-clock time across a total of 312 processors spread between the IA-64 clusters at NCSA and SDSC. These systems were selected because they have a similar architecture and similar performance per node. Later it was decided to do large scale production runs on single, large computational resources. First HPCx (the UK's national supercomputer resource, a "constellation"-style cluster of IBM Power based SMP machines), then Ranger (Ranger is a Sun Linux/AMD cluster at TACC with 62,976 cores, 123 terabytes of RAM and a peak performance of 504 TeraFLOPS) and finally the "Abe" cluster at NCSA. Abe is a Dell Linux/Intel cluster with 1200 nodes/9,600 cores and 14.4 terabytes of RAM (half of the nodes have 16 gigabytes of RAM and half have 8 gigabytes). Abe's peak performance is 89.47 TeraFLOPS. That LAMMPS works reliably on all these resources and with good performance and scaling is a testament to its portable nature.

Due to the difference between the architectures of the compute resource used for heating up the replicas after the equilibration phase, and those used for production runs it was necessary to convert the restart files to ASCII rather than using the standard LAMMPS binary formats. This may have induced a small loss in precision as well as increasing the size of the output files but meant that it was possible to move them between platforms.

The force-field used was the CHARMM force-field. This was used because of problems converting data-files output by NAMD using the AMBER force-field into LAMMPS input files.

7.2.4 Replica exchange issues

An initial production run was made in late 2007 on the HPCx resource. Analysis of the output showed that the replicas were not swapping regularly. Figure 7.1 shows a graphical representation of the swapping behaviour of the HPCx run. The simulations are coloured according to their temperature. This graphic clearly shows that the only replicas exchanging configurations are the hotest ones (replicas nine to thirteen on that graphic) and that these exchanges happen infrequently.

This was interesting because earlier tests and benchmarks performed while developing the code (see chapter 6) showed good swapping. These results indicated that it was necessary to perform some test runs (with the right number of simulations, but for much shorter time-scales) to determine hat factors affect the rate of exchange between the replicas.

Figure 7.2 shows the swapping behaviour of a number of runs. Top left shows the behaviour with the pre-heating phase where each replica is heated to the correct temperature and 5K temperature separation between the replicas. This run was terminated early due to issues on the machine, but shows no sign of swapping at all. Top right shows a test run for the same system, but starting all the replicas from the 300K replica rather than from their pre-heated systems. As can be seen, the swapping frequency is much higher, but by the end of the run the swapping frequency has decreased. Bottom left shows a similar system to the first two, but with the selection algorithm changed so that potential pairs to be exchanged are selected pseudo-randomly rather than alternating between odd and even pairs. As can be seen exchanges happen less often than alternating between odd and even pairings. Bottom right shows the swapping of a system with replicas which have a smaller temperature gap of 2K between the replicas. Here it can be seen that swapping appears to continue to the end of the run.

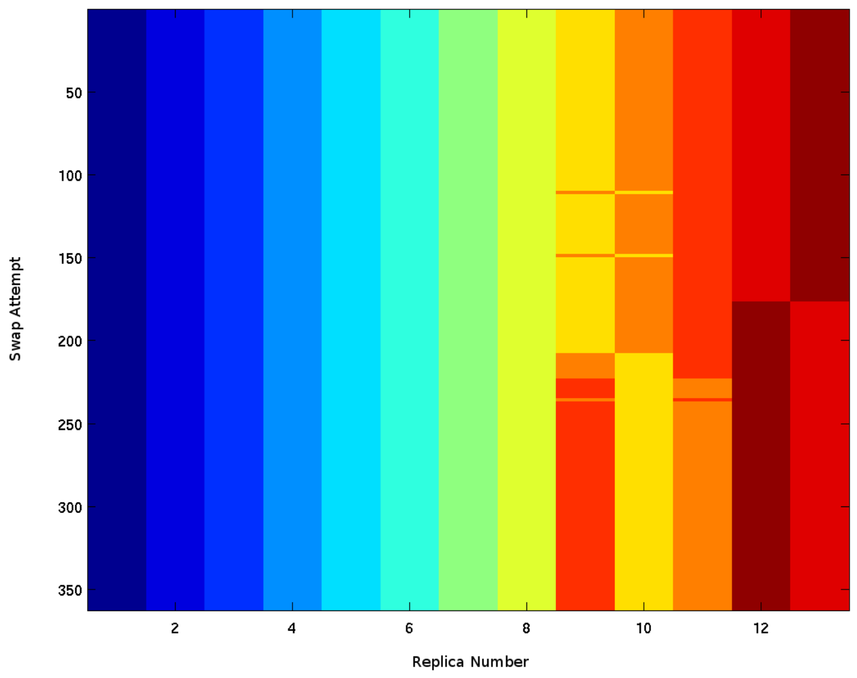


Figure 7.1: Swapping behaviour of HPCx run. The replicas are coloured based on temperature from coldest (dark blue) to hottest (dark red). The x-axis denotes the replica number from 1 to 13 ordered in terms of starting temperature from 290K to 350K. The y-axis denotes the exchange attempt number so the system can be thought of as evolving from the state at the top to that at the bottom. The y-axis in this graphic represents a much longer time-scale than in figures 3 or 5 as in this system the exchanges were tried once every 1000 time-steps instead of once every 100 in the later test systems.

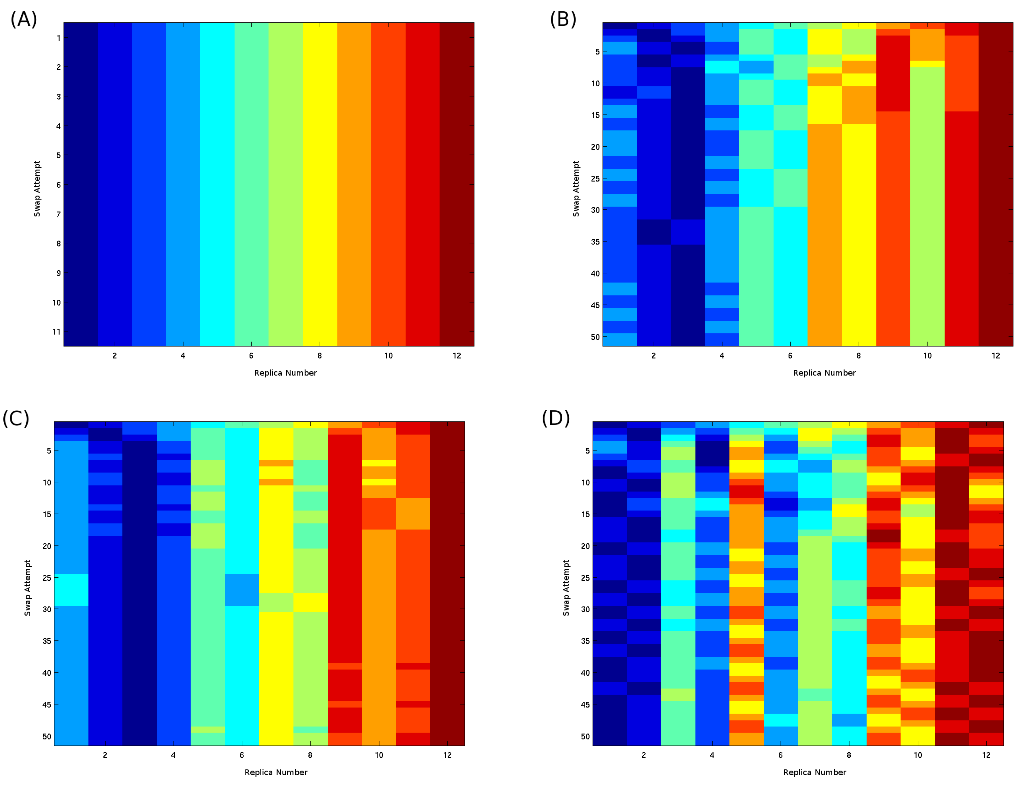


Figure 7.2: Swapping behaviour of test runs. Top-left (A): 5 degrees separation, preheated. Top-right (B): 5 degrees separation, no preheat. Bottom-left (C): 5 degrees separation, random, no preheat. Bottom-right (D): 2 degrees separation, no preheat. The replicas are coloured based on temperature from coldest (dark blue) to hottest (dark red). The x-axis denotes the replica number from 1 to 12 ordered in terms of starting temperature from 290K to 345K in the 5K separation systems and 290K to 312K in the 2K separation systems. The y-axis denotes the exchange attempt number so the system can be thought of as evolving from the state at the top to that at the bottom. Note that in the case of the top-left simulation, this was run for fewer steps than the others (due to application termination on the machine used) but all others had shown swapping by the time they reached the same stage as it terminates and the longer HPCx run (figure 6.1) showed very little additional swapping behaviour for this system over a much longer time-scale.

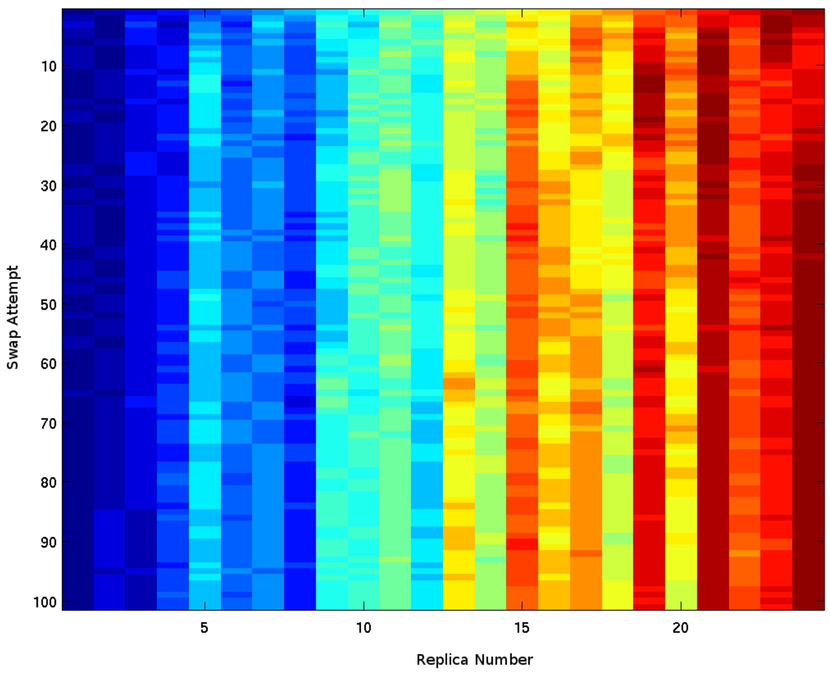


Figure 7.3: Swapping behaviour of a system run for twice as many swapping opportunities as in figure 7.2, with 24 replicas, pair-wise swapping, 2K separation and no preheat. The replicas are coloured based on temperature from coldest (dark blue) to hottest (dark red). The x-axis denotes the replica number from 1 to 24 ordered in terms of starting temperature from 290K to 324K. The y-axis denotes the exchange attempt number so the system can be thought of as evolving from the state at the top to that at the bottom.

If we want to cover the same temperature range as was initially planned, then we need to have more replicas than initially planned. Figure 7.3 shows the swapping behaviour for a system with 24 replicas with the smaller temperature gap (2K) and a much longer simulation time. As can be seen, in this case in general swapping continues to the end of the run. A notable exception to this behaviour is for the hottest temperature where swapping appears to die out. It seems likely that this is due to that replica evolving into a state where the swapping test always fails because its configuration is too different from all the other replicas.

A much more extensive production run was carried out during early user access to Ranger. A computational resource this large is ideal for replica exchange because the availability of processors allows many replicas to be run. In addition, early tests with LAMMPS showed that its performance on Ranger was very good, scaling further than other TeraGrid sites. This, combined with the lessons about swapping frequency, led to the increase the number of replicas to 60 over a temperature range of 290K to 349K, each with 32 processors for a total of 1920 cores. This production run was then extended a further nanosecond using the Abe cluster. The use of restart files allowed the simulations to be moved between the two sites to be continued.

7.3 Results and analysis - production

7.3.1 Swapping behaviour

As a first analysis step, the swapping activity of the 60 replicas was plotted in the same way as before (figure 7.4). The replicas were run for 1 nanosecond with pair-wise swapping, 1K separation between 290K and 349K. This showed good and continued swapping for the entire duration of the run. Reassuringly, this plot shows that the system is evolving so that over time, swapping is happening from hot and cold temperatures into mid-range temperatures and *vice-versa* which is a good indicator of good sampling conditions.

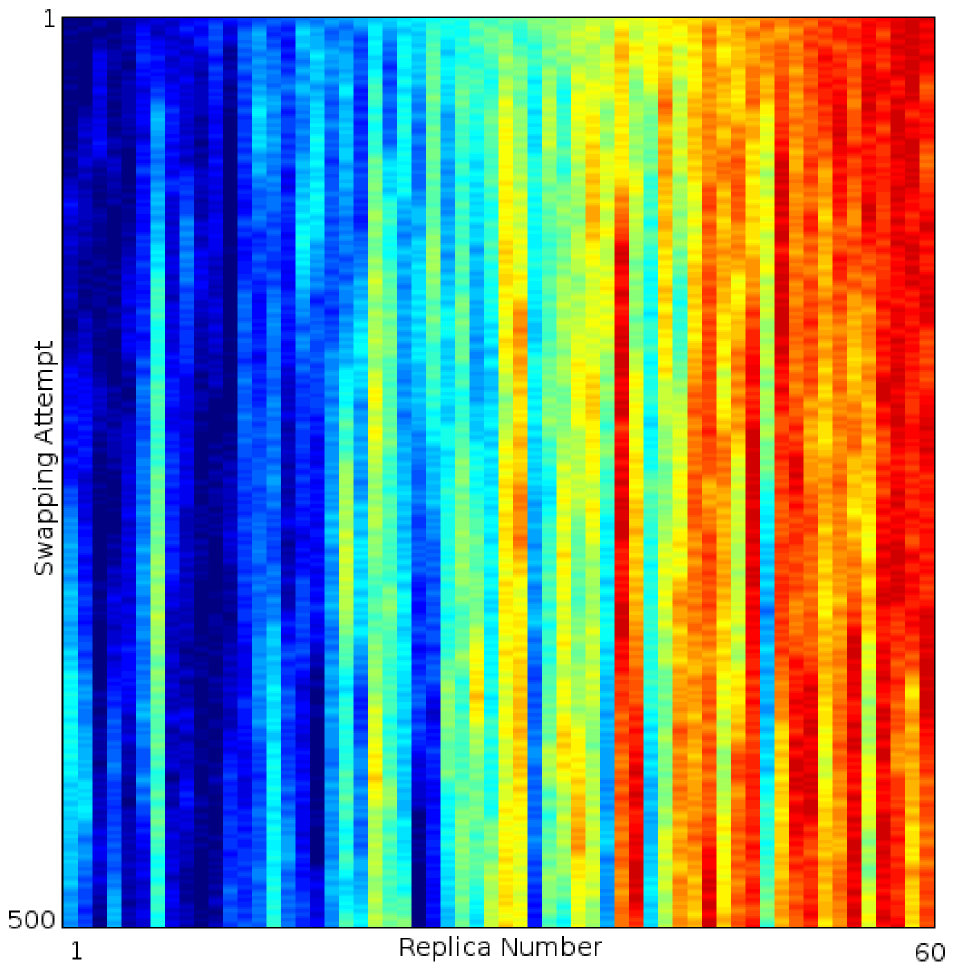


Figure 7.4: Swapping behaviour 1 nanosecond production run on Ranger with 60 replicas, pair-wise swapping, 1K separation between 290K and 349K, coloured based on temperature from coldest (dark blue) to hottest (dark red). This plot shows good and continued swapping even over an entire nanosecond of simulation and 500 swapping steps.

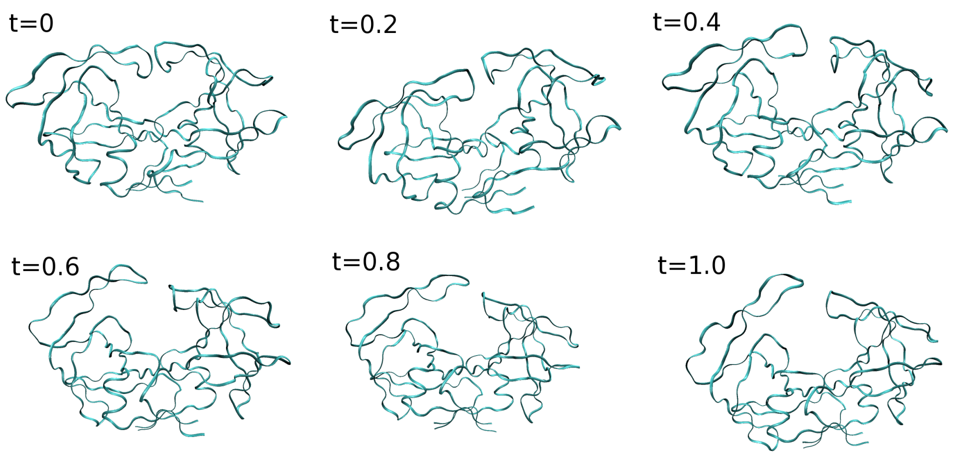


Figure 7.5: Configuration of 348K replica from production run on Ranger at (left to right, top to bottom) 0 nanoseconds, 0.2 nanoseconds, 0.4 nanoseconds, 0.6 nanoseconds, 0.8 nanoseconds and 1 nanoseconds showing flap opening event.

7.3.2 Visual inspection

With adequate swapping behaviour confirmed, the next analysis stage might be to perform a visual inspection of the trajectory files using the VMD visualisation tool and categorising the behaviours demonstrated by the different replicas. Unfortunately, while this method may seem sensible with a single trajectory simulation, in a replica exchange system, the researcher would have to visually inspect a very large number of trajectories. It is a much better approach to determine some metric for computationally flagging trajectories that may show an opening event (see section 7.3.3) and then visually inspecting only those trajectories.

7.3.3 Flap tip separation

In determining whether or not a flap opening event has been observed, a more objective metric than visual inspection needs to be determined. In [44], it is proposed that the distance between the tips of the flaps (residue ILE-50C^{α} to ILE-50'C^{α}) may be used to quantitatively determine the state of the flaps. They propose that a distance of $\sim$4.3 angstroms represents a semi-open state while $\sim$5.8 angstroms represents a closed state

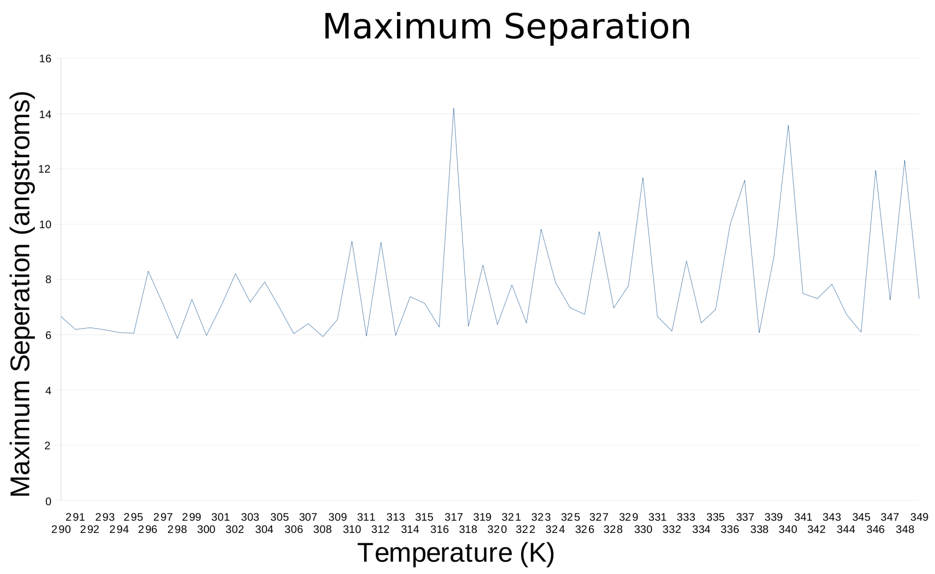


Figure 7.6: Maximum separation of the ILE-50C^{α} and ILE-50'C^{α} residues (i.e. the flap tips) in angstroms for each of the replicas over the full two nanoseconds. From this graph it can be seen that the replica with the largest separation is the 317K replica.

but offer no suggestions for the open state. The greater distance in the closed state is due to the way the flaps close. For the purposes of this thesis, a tip separation of over 8 angstroms will be considered as possibly being in an open state.

Figure 7.6 shows the maximum separation of the ILE-50C^{α} and ILE-50'C^{α} residues (i.e. the flap tips) in angstroms for each of the replicas over the full two nanoseconds. From this graph it can be seen that the replica with the largest separation is the 317K replica.

Categorised in this way, the replicas which have a separation indicating that a flap opening event has occurred are 310K, 312K, 317K, 319K, 323K, 327K, 330K, 333K, 336K, 337K, 339K, 340K, 346K, and 348K. This can then be cross-referenced with the visual inspection results giving a number of candidate replicas including the 317K replica as well as 327K, 330K, 336K, 337K, 340K and 348K replicas

7.4 Conclusions

From the results of this study it is clear that the replica exchange molecular dynamics method has considerable merit, both in terms of efficiently exploiting modern massively parallel computational resources where traditional molecular dynamics methods exhibit scaling issues and in terms of scientific merit allowing the researcher to more effectively sample configurational space than by traditional methods.

The conformations created here could have numerous purposes in future studies. As it stands there is a considerable lack of experimentally derived open structures available. Replica exchange molecular dynamics provides a suitable method of deriving these structures computationally.

In this study, open structures have been derived from a single HIV-1 protease structure but many structures are available in the protein databank. The HIV virus shows considerable variation due to its high rate of mutation and these mutations have a considerable effect on the resistance of the virus to the the numerous clinically available drugs. Generating open structures from only one starting structure limits the knowledge gained to that particular strain of the virus, because mutations to the flaps are a major contributor to resistance in the virus and could have a considerable impact on the flap dynamics. It is therefore probably not appropriate to use tools such as those provided in the BAC (see chapter 8) to introduce mutations into the open structures that have been generated computationally using replica exchange. The most suitable course of action for investigating open structures of other mutants of HIV protease is to do a replica exchange study starting from the chosen structure (either from the protein databank or generated by the BAC) to generate an open version of that structure using the tools described here.

There are numerous experimental methods available for measuring drug efficacy[45]. Some, such as EC_50 are done in blood plasma and therefore have the advantage that they are representative of the conditions within the patient. However these methods are complicated by practical issues in performing the measurements. Others, such as I_50 are performed in solution which provides for much easier measurements, but potentially less representative results. Although both methods are commonly used, neither has a direct link to rate constants. Being able to computationally derive open structures to allow computational drug docking to estimate drug efficacy could compliment these methods considerably.

The open conformations shown here show some potential open conformations for the the protease but due to the difficulty in obtaining a comprehensive database of experimentally derived open conformations it is difficult to determine whether these conformations are representative of the most common open conformations for the protease. More study is needed in this area.

Chapter 8

Study of Computational methods for Approximating Binding Affinity of Lopinavir to HIV-1 Protease

8.1 Introduction

The primary problem in the treatment of HIV patients is the occurrence of drug-resistant mutants of the virus affecting drug efficacy. Currently, patients are genotyped (i.e. the DNA of the strain of HIV they are infected with is obtained experimentally) to determine which mutations are present in the virus they have contracted and then the clinician makes a decision on which drug or drugs to prescribe using a combination of experience, intuition and, sometimes, a decision support system which is designed to help process the vast quantities of data available. Some of these packages are commercial while others (for example those under development as part of the ViroLab project

[1]) are open source solutions. Such software products are in the form of expert systems, themselves entirely based on historical data of drug resistance caused by particular mutations stored in a number of large databases. Unfortunately, this method has an inherent flaw when mutations are new, or combinations of mutations are present in that there is a level of complexity which cannot be easily handled by an approach based on statistical inference rather than an instance specific model. The ability to simulate the patient's strain of the virus bound to the various available drugs and provide a reasonably accurate ranking (i.e. a ranking in the correct order, but not necessarily the exact value) of the potential efficacy of those drugs against that patient-specific strain of the virus to aid in drug selection would be an extremely helpful tool. Until relatively recently, the computational power to do this on a clinically useful scale has not been available, but with the inevitable march of technological innovation, computers have reached the point where this may now be possible. In order to do this a metric needs to be selected to give a reasonable predictor of drug efficacy. It needs to be tested well, proved to hold up to experiment and capable of coping with multiple mutants while still providing a good ranking of the available drugs on a clinically appropriate time scale (approximately two weeks).

A potential metric for determining resistance to a particular drug is the binding affinity[34, 35, 36] between the drug and the protein it binds to (in the case of protease inhibitors, HIV-1 protease). The binding free energy (ΔG_b) may be approximated using the MM-PBSA method.

For the purposes of ranking of mutants of HIV-1 protease when bound to a particular drug, a better metric of binding is "relative binding affinity" ($\Delta\Delta G$), i.e. the difference in binding affinity between the mutant and the wild-type:

$$\Delta\Delta G = \Delta G_{mutant} - \Delta G_{wildtype} \tag{8.1}$$

[1]ViroLab project web-site: http://www.virolab.org/

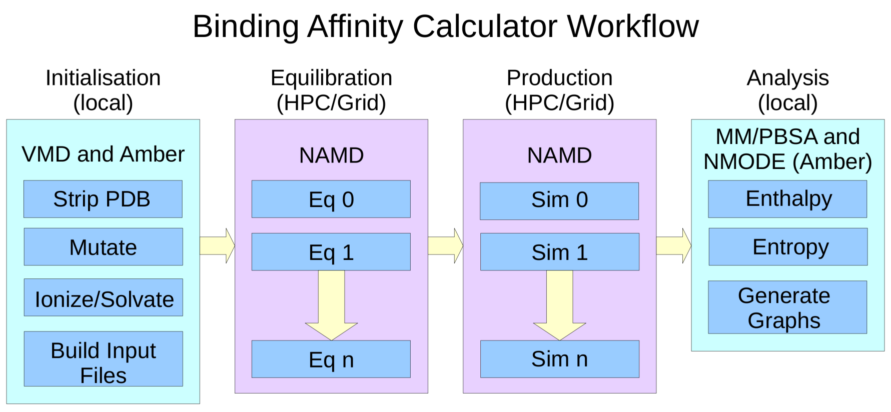


Figure 8.1: Workflow of the Binding Affinity Calculator showing standard stages and codes used, as well as which parts of the workflow may utilise High Performance Computing or Grid resources.

Ohtaka[46] describe experimental measurement of binding affinities for lopinavir (and a number of other protease inhibitors with six different systems. These values are used for comparison in this study.

In this study, the wild-type is structure HXB2 from the Protein Databank. A wild-type mutant in a virus such as HIV which shows considerable mutation may seem difficult to select, but for experimental purposes, what is desirable is a mutant that statistically shows no known resistance-causing mutations to use as a base structure for comparison with mutants that have the mutations that are of interest to this study.

This method has shown some promise with protease inhibitor known as “saquinavir”[51, 52].

8.1.1 The Binding Affinity Calculator

The Binding Affinity Calculator (or "BAC")[47] is a set of tools for calculating binding affinities of ligands with proteins developed in the CCS for constructing complete systems of HIV-1 protease bound to ligands from the Protein Data Bank, adding in TIP3P[53] water molecules, hydrogen atoms, mutations and constructing configuration files for NAMD and job submission scripts for a large range of supercomputer resources. This allows a user to set up and run molecular dynamics simulations of HIV-1 protease in NAMD and then analyse them to calculate the binding affinities in an automated and uniform way.

The BAC wraps around binaries provided with VMD (a visualisation package), AMBER and NAMD and is written mainly in Perl; it includes wrappers for a number of analysis stages after the output from the molecular dynamics runs has been retrieved. These include tools for converting the output (standard DCD files) from NAMD into AMBER trajectory files and then calculating the binding free energy (ΔG_b) using the MM-PBSA method described in equation 8.1 below.

$$\Delta G_b = \Delta G^{MM}_{vdW} + \Delta G^{MM}_{ele} + \Delta G^{sol}_{pol} + \Delta G^{sol}_{nonpol} - T\Delta S \tag{8.2}$$

ΔG^{MM}_{vdW} is the van der Waals component and ΔG^{MM}_{ele} is the electrostatic component of the free energy difference. ΔG^{sol}_{pol} is the polar component of the solvation free energy and G^{sol}_{nonpol} is the non-polar component. Finally, $-T\Delta S$ is the component from the change in entropy.

$\Delta G^{MM}_{vdW} + \Delta G^{MM}_{ele} + \Delta G^{sol}_{pol} + \Delta G^{sol}_{nonpol}$ contributions to equation 8.1 using the MM-PBSA module from the AMBER package and the $-T\Delta S$ using NMODE. Because of the lack of scaling of MM-PBSA and NMODE (the versions currently in use are serial, although work on getting a parallel version of NMODE into the BAC is ongoing), the analysis steps are preformed on Mavrino, 96 core, local CCS Linux cluster. Work on extending this toolkit is on-going, for example: adding the automated generation of graphs (using the free graphing tool GNUPlot) of the output of the analysis steps, reducing a time-consuming and sometimes fiddly process to a single command. The General AMBER Force Field[22] is used to parametrise the ligands the standard AMBER forcefield for biorganic systems[23] is used to parametrise the protease.

Future plans for development of the BAC include integrating it into the Application Hosting Environment (or "AHE") which is a graphical user interface designed to ease the use of Grid resources. This would automate job submission, file staging and a number of other tasks which currently have to be managed (or at least initiated) manually, simplifying the user interface of the BAC still further.

8.2 Method

8.2.1 Computing infrastructure

The vast majority of the computational work for this study was carried out on the "Ranger" terascale cluster at the Texas Advanced Compute Center (or TACC). Ranger is a Linux cluster with 3,936 compute nodes, each of which is a single Sun SMP blade with four four-core AMD Opteron processors for a total of 62,976 compute cores. Nodes are connected with an Infiniband interconnect and the theoretical peak performance of the entire system is 504 TeraFLOPS. In addition to this resource, some early simulation, and the equilibration phases were run on the Manchester and Leeds National Grid service clusters (256 cores each) and analysis was carried out on both a local cluster with 96 cores and the Leeds NGS cluster.

Wall-clock time on Ranger equated to 8 hours per nanosecond of simulated time on 32 cores meaning that excluding equilibration and analysis, a fifty nanosecond, single trajectory run took at total of 400 wall clock hours. In comparison, the four nanosecond replicas within the ensemble complete in 32 wall clock hours each and can be be run simultaneously (32 cores times 300 replicas is 9600 cores) on a resource as large as Ranger.

8.2.2 Protease mutants and lopinavir

The starting structure for this study is the structure 1MUI which comes from the Protein Data Bank already bound to lopinavir. This structure was mutated with mutation S37N which turns it into 1HXB.

The Ohtaka study measures binding affinities experimentally by measuring heat involved in the reaction between the protease and the ligands. Binding affinity is related to drug resistance because it measures how strongly the protease binds to the drug or to the substrate. Therefore the change in the strength of this binding due to mutation can be used to provide a metric to show how much a mutation affects the ability of the drug to bind into the active site and prevent the protease from functioning.

There are six mutants for which experimental binding affinities are available in Ohtaka. The first of these is the "wild-type" with none of the six mutations present. Then there are three double mutants, each with a pair of the six mutations. These are L10I/L90M which are a pair of mutations that occur in the dimerisation region of the protease, M46I/I54V which occur in the flap region and V82A/I84V which occur in the region of the active site. In addition, a mutant with the active site and flap mutations only (i.e. M46I/I54V/V82A/I84V, referred to as the "quad-mutant") and a mutant with all six (referred to as the "hexa-mutant") were also studied. These six mutants were chosen for two reasons. Firstly, because they allow comparison with the experimental results in Ohtaka and secondly because they allow the investigation of the change in resistance caused by combinations of mutations that may affect each other. Figure 8.2 shows a rendering in VMD of the hexa-mutant backbone bound to lopinavir (coloured in green), with each of the six mutation structures shown and highlighted in different colours. L10I is shown in blue, L90M in purple, M46I in red, I54V in orange, V82A in yellow and I84V in pink. One of the I54V mutations has bent over slightly which is

Mutant	ΔG **(kcal mol^{-1})**
Wild type	-15.1
Hexa-mutant	-11.3
Quad-mutant	-12.8
V82A/I84V	-13.9
M46I/I54V	-14.9
L10I/L90M	-14.9

Table 8.1: The experimental binding affinities of the six mutants with lopinavir from table 2 published by Ohtaka in [46]. The error claimed by Ohtaka in the Enthalpy contribution is +/- 0.2 kcal^{-1}.

normal behaviour for a part of the structure exposed to the solution (in this case water). The Ohtaka study showed that these individual pairs of mutations lowered the binding affinity of a number of drugs (including lopinavir) by a relatively small amount, but that the hexa-mutant lowered the binding affinity by considerably more than the sum of all those changes. The relatively small change in the quad-mutant implied that the L10I/L90M mutations in the dimerisation region have a strong effect on the combined resistivity of the hexa-mutant.

In order to validate the computational method, these are the same six systems that will be used in the lopinavir computational study. The binding affinities measured by Ohtaka are reproduced in table 8.1.

8.2.3 Equilibration

The BAC creates systems from crystal structures from the protein databank of various mutants of HIV-1 protease bound to the specific inhibitor that the user requires. The mutations that the user wishes to study are then added into the structure by the BAC and this means that the equilibration protocol has to allow these mutations to settle into a stable state in addition to the usual equilibration routine.

The systems were equilibrated using the standard equilibration protocol for the BAC described fully in [47]. Each system was minimised for 2000 iterations using the conju-

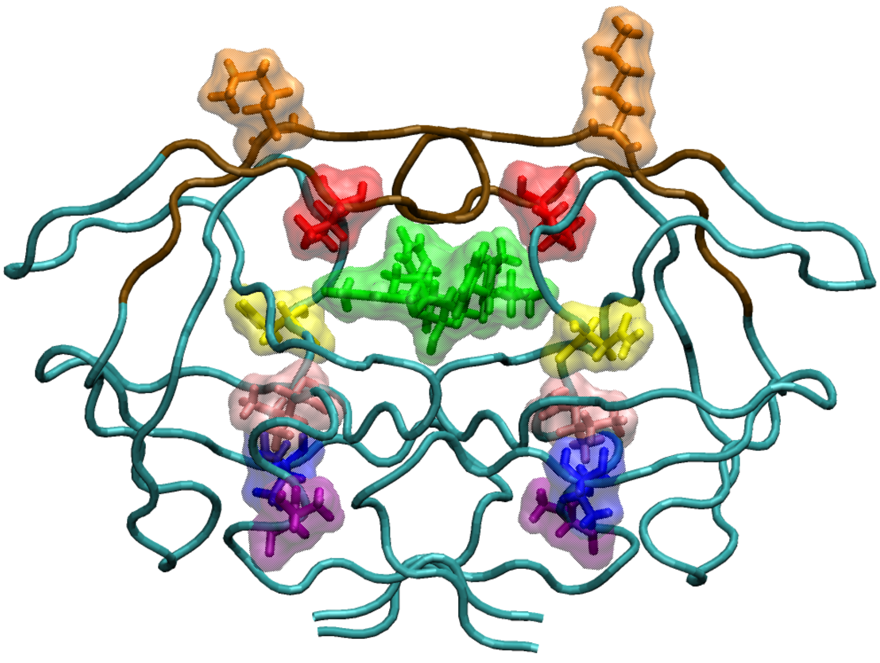


Figure 8.2: Multi-drug resistant hexa-mutant of HIV-1 protease bound to lopinavir (green), showing the locations and structures of the six mutations - L10I (blue), L90M (purple), M46I(red), I54V(orange), V82A(yellow), and I84V(pink). The flaps are highlighted in brown. Image rendered in VMD.

gate gradient method and line search modules in NAMD. 50 picoseconds of simulation time was then spent heating the system from 50K to 300K. Throughout equilibration a 2 femtosecond time-step was used. Once this step was complete, the pressure was controlled by coupling the simulation to a Langevin thermostat[30] and the pressure was controlled with a Berendesen barostat[48] forming an isobaric (or NPT) ensemble.

Following this stage, a complex sequence of mutation-related relaxations was carried out with each of the amino acids mutated as part of the system construction protocol relaxed in turn over a period of 50 picoseconds to allow each mutation to settle into a more equilibrated position. While each amino acid was being relaxed all the others were constrained.

The forces on the ligand and then the protease were relaxed over a period of 200 picoseconds and 150 picoseconds respectively. Finally, the entire system was run without the constraining forces (other than the thermostat and barostat required to maintain the NPT ensemble) leaving only physical forces (such as electrostatics etc.) for two nanoseconds to complete the protocol.

This equilibration process was determined to be sufficient because inspection of physical parameters (temperature, pressure) showed that by the end of the equilibration process these parameters had become relatively stable.

8.2.4 Protonation study

Protonation is an important stage in the preparation of molecular dynamics simulations from crystal structures where H^+ ions are added to the system. For HIV-1 protease bound to a particular ligand (in this case lopinavir), there are multiple possible protonation states. It was therefore necessary to perform a small-scale protonation study to determine which should be used for the full-scale lopinavir study.

When bound to lopinavir, there are two aspartic acid sites within the HIV-1 protease homodimer that may be protonated, which for ease of reference shall henceforth be referred to as “a” and “b”. This means that there are four possible protonation states. Either monomer may be protonated (“a-protonated” or “b-protonated” - known collectively as “mono-protonated”), neither (“unprotonated”) or both (“di-protonated”). This possible imbalance in the way the otherwise rotationally symmetrical structure of HIV-1 protease is protonated is caused by the addition of the drug which makes the drug/protease complex asymmetric.

In order to determine which of the four states was most appropriate for the lopinavir study, four systems were set up (one for each protonation state) with the wild-type bound to lopinavir and then run for four nanoseconds. These were then analysed with MM-PBSA and NMODE and the system with the most stable looking (i.e resulting in the lowest ΔG) entropies/enthalpies selected for the full study. For the lopinavir system, this investigation indicated that of the four possible protonation states, the most probable state was “mono-protonated” with the “b” monomer protonated.

8.2.5 Long time-scale molecular dynamics

Each system was run for an extended period of time post equilibration with NAMD. The protocol used the same isobaric protocol as was used for the last two nanoseconds of the equilibration protocol; the pressure was controlled by NAMD’s Berendsen barostat (and kept at approximately 1 bar), while the temperature was controlled by a Langevin thermostat at 300K. SHAKE was used to keep the bond length for the hydrogen atoms constant in order to allow a two femtosecond time-step to be used. The algorithms by which these methods operate are more fully discussed in chapter 3.

Initial production simulation durations were ten nanoseconds. After these were analysed, the simulations were subsequently extended to twenty nanoseconds and then fifty nanoseconds with two of the systems (the wild type and the M46I/I54V mutant) extended to one hundred nanoseconds.

8.2.6 Ensemble Molecular Dynamics

In addition to the long timescale molecular dynamics, an ensemble method protocol was devised with fifty replicas of each of the lopinavir systems. Fifty replicas were created from the original PDB and subjected to the equilibration protocol individually (to ensure that they were completely independent). Each of the additional replicas was subjected to the same equilibration protocol as the long time-scale system, the only difference being the initial randomised velocities at the start of the simulation. Each replica was run for a total of four nanoseconds.

With a single, long time-scale simulation, there is a danger that the randomised velocities at the start have a considerable impact on the subset of configurational space explored by the simulation. Using ensemble methods should help avoid this situation, and the comparison of the results derived from the two methods is important in determining the most efficient way to use available computational power to arrive at an accurate result.

8.2.7 Analysis

Analysis was performed using AMBER's MM-PBSA package to calculate the enthalpic contributions to the binding affinity, and using AMBER's NMODE package to calculate the entropic contributions. The MM-PBSA analysis was performed on 100 snap-shots per nanosecond while the NMODE analysis was performed on 5 snap-shots per nanosecond. The considerably lower frequency of snap-shots used by NMODE was due to the high computational cost of that code in processing a snap-shot relative to MM-PBSA. This process was automated by scripts within the BAC and performed on Mavrino and the Leeds NGS cluster.

There does exist a parallel version of NMODE but at the time of study it had not been integrated into the BAC. The parallel version of NMODE actually consists of two parallel implementations, one of which is the most generally applicable and is written in MPI (the other is uses a shared memory model and is written in OpenMP, limiting its use to large shared memory machines which are relatively rare). Unfortunately this implementation was written targeting Sun Microsystems machines and portions of the code appear to be non-portable to other machines. The other implementation is in OpenMP and is therefore limited to shared memory machines with relatively small numbers of processors (although modern clusters are constructed out of SMP nodes, most of them are relatively small (typically two to four processors) and none exceed sixteen processors. A relatively large shared memory machine is planned as part of the UCL Legion system which may in future allow the SMP parallel version of the NMODE code to be used. Once either version is working, it should be possible to increase the number of snapshots used in the entropic part of the calculation decreasing the error in the calculation.

In addition to the analysis computational requirements, the simulations produced an extremely large quantity of geographically distributed data (the single trajectory simulations produced in excess of half a terabyte alone). This data was transferred via a combination of normal internet traffic and dedicated fibre-optic links to a local RAID array.

The data output by the two methods (single and ensemble) was compared with a Gaussian distribution to determine whether they were indeed producing a Gaussian distribution. This analysis comprised of two parts: Binning the data and then visually inspecting the data when overlaid with a Gaussian distribution curve with corresponding mean and variance, and using the Kolmogorov-Smirnov test (described below), manually implemented in Octave to compare the closeness of the distributions with the same curve.

The Kolmogorov-Smirnov statistic provides a metric for determining the closeness of the empirical distribution function ($E(x)$) of a given sample to a given cumulative distribution function ($F(x)$). It does this by calculating the Kolmogorov-Smirnov statistic which is given by $KS = SUP|F(x)-E(x)|$. SUP is the maximum limit of a series which for an experimental data set is analogous to MAX. This may more easily thought of as KS being the largest difference between the empirical distribution and the selected cumulative distribution. The KS statistic may then be compared to a look-up table of values and for a given sample size this accepts or rejects the null hypothesis that the sample is from the given data set. This is known as the "Kolmogorov-Smirnov test". This method may be modified easily into the Lillifors method with similar initial steps and a change in the interpretation of the KS statistic.

For the purposes of this analysis, the system snapshots were considered to be independent, although this may not seem to be intuitively the case. As with all other analysis, the enthalpic and entropic contributions were considered separately.

Code to calculate the Kolmogorov-Smirnov statistic was written from first principles in Octave after it was determined that the built-in library functions were using an unusual algorithm for calculating the Kolmogorov-Smirnov statistic and using methods other than using a look-up table for performing the Kolmogorov-Smirnov test upon that statistic.

8.3 Results

8.3.1 Convergence

The graphs of ethalpic (top-left) and entropic (top-right) contributions to ΔG for a single trajectory 50 nanosecond run of wild-type HIV-1 protease bound to lopinavir are shown in figure 8.4. The sliding window averages (blue/black) for the enthalpic contribution show noticeable step changes at approximately 8 nanoseconds and 28 to 30 nanoseconds. The corresponding pair of graphs for the hexamutant are shown in the middle of figure 8.4. This is more unstable but has relatively few features which stand out. Finally, the bottom pair show the same data for the quad-mutant. In this instance there are noticeable step changes in the enthalpic contribution over time period from

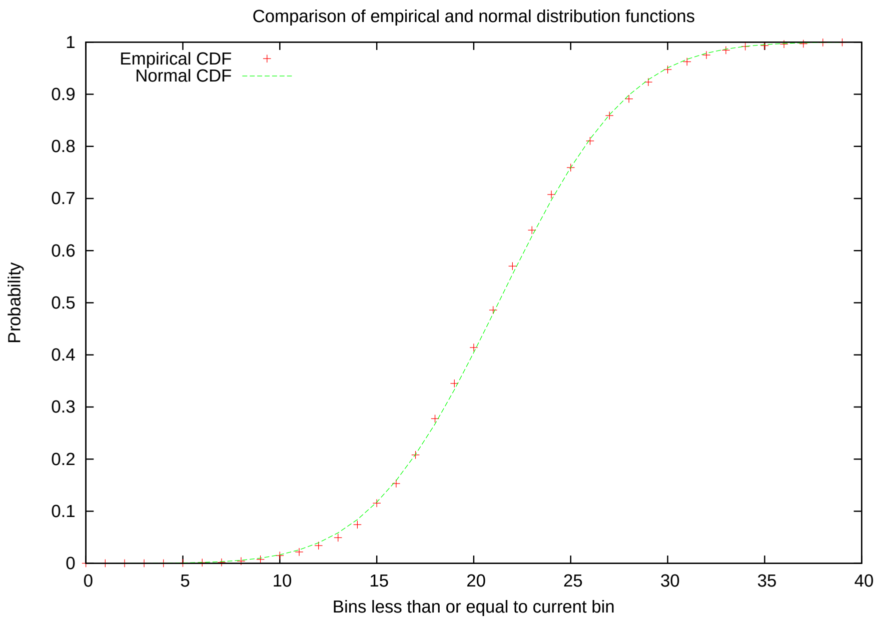


Figure 8.3: Visual comparison of the empirical cumulative distribution function of the enthalpic contribution to the first 50 nanoseconds of wild-type HIV-1 protease bound to lopinavir (replica zero/long timescale), binned into forty bins shown in red with the normal cumulative distribution function centred around the mean with the same variance as the sample (green). The Kolmogorov-Smirnov statistic is the largest difference between these two series.

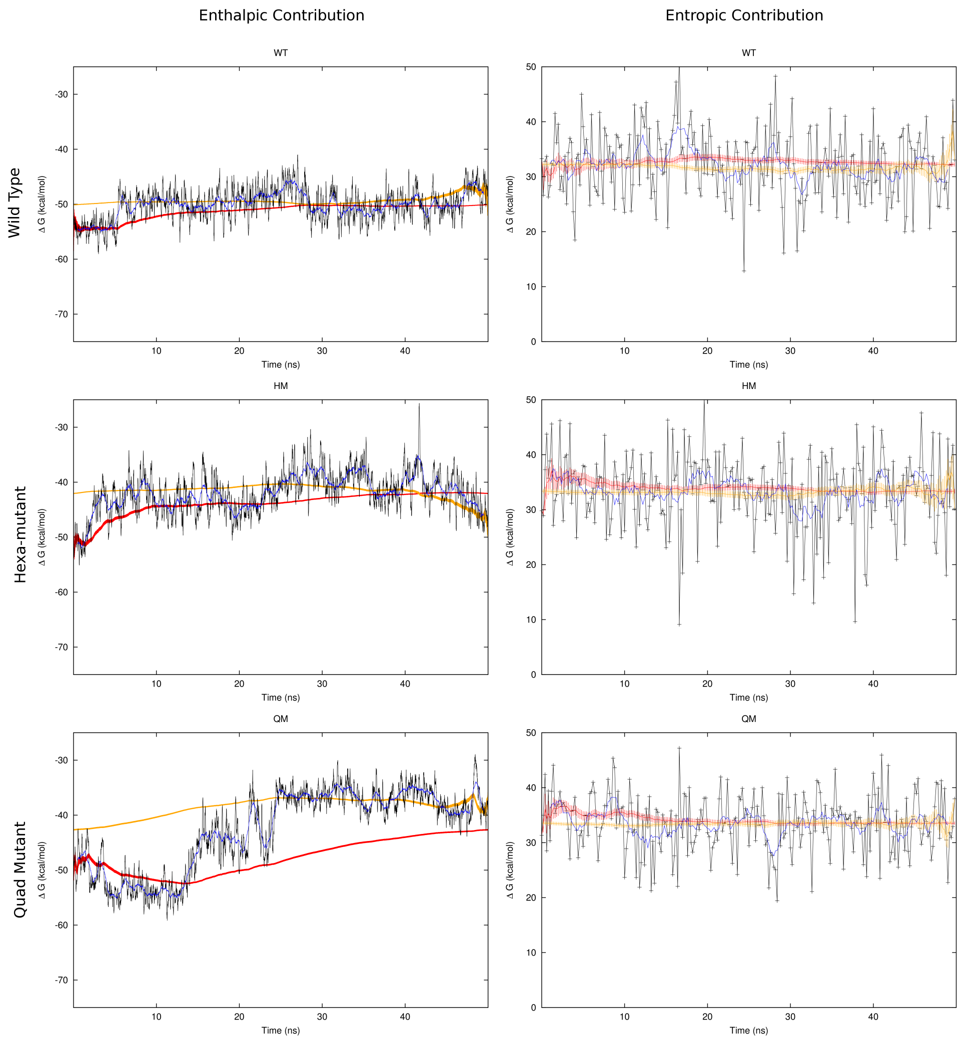


Figure 8.4: Convergence graph for the enthalpic (left) and entropic (right) contributions to ΔG of a 50 nanosecond single trajectory simulation of HIV-1 wild-type (top), hexa-mutant (middle) and quad-mutant (bottom) protease bound to lopinavir. Red and orange lines represent forward and backward running averages with the thickness of the line indicating the error. Red indicates a forward running average, while orange is a reverse running average. Black and blue lines represent ten and one hundred sample sliding window averages of the same data.

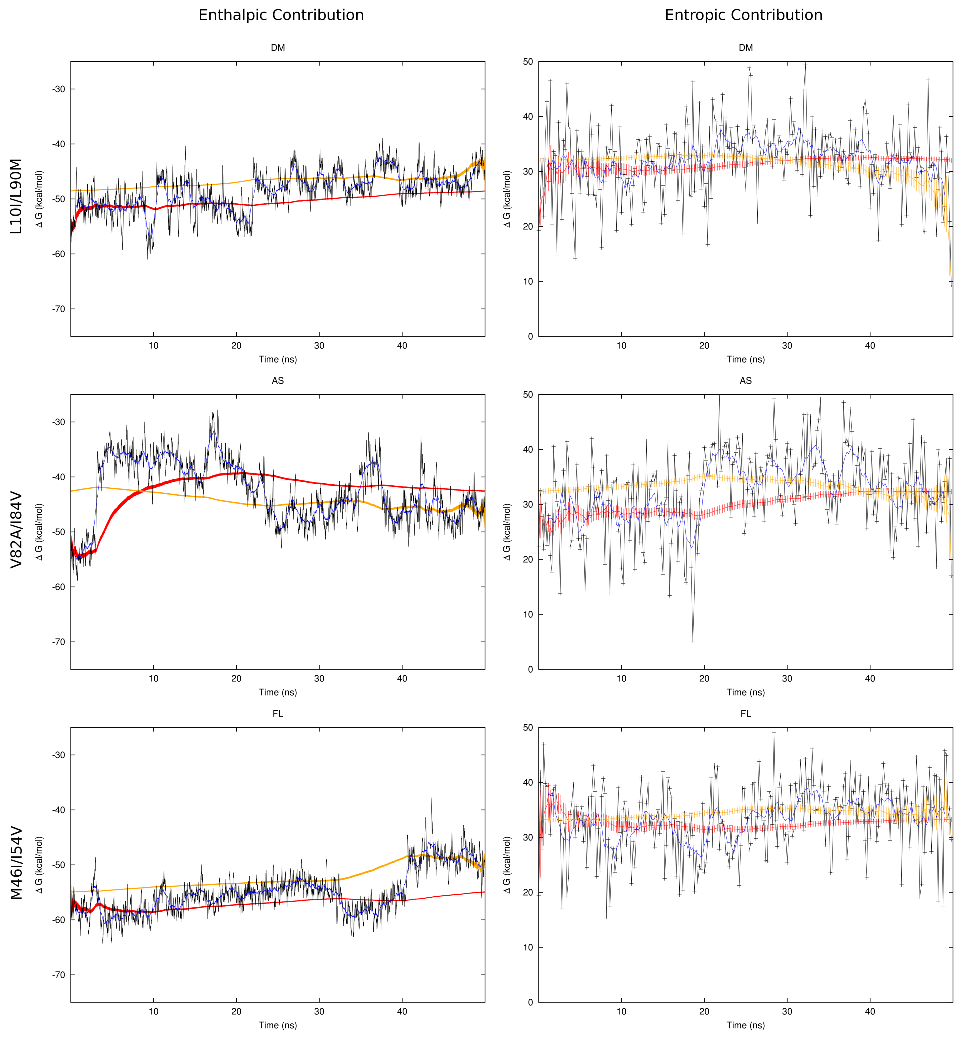


Figure 8.5: Convergence graph for the enthalpic (left) and entropic (right) contributions to ΔG of a 50 nanosecond single trajectory simulation of HIV-1 L10I/L90M (top), V82A/I84V (middle) and M46I/I54V (bottom) protease bound to lopinavir. Red and orange lines represent forward and backward running averages with the thickness of the line indicating the error. Red indicates a forward running average, while orange is a reverse running average. Black and blue lines represent ten and one hundred sample sliding window averages of the same data.

Mutant	**Ens.** ΔH	**Single** ΔH	**Ens.** $-T\Delta S$	**Single** $-T\Delta S$
Wild type	0.017795	0.0097871	0.031352	0.013251
Hexa-mutant	0.029874	0.010832	0.038078	0.039572
Quad-mutant	0.033840	0.055384	0.019851	0.021720
V82A/I84V	0.014474	0.025567	0.039575	0.046216
M46I/I54V	0.011030	0.021127	0.055807	0.033119
L10I/L90M	0.011952	0.0062954	0.036591	0.041649

Table 8.2: Kolmogorov-Smirnov statistic for the ΔH and $-T\Delta S$ contributions to ΔG in comparison to a normal (Gaussian) distribution about the mean of the sample for snapshots from the single, 50 nanosecond trajectories and from the 50 by 4 nanosecond ensembles (marked "Ens."). Lower values of the Kolmogorov-Smirnov statistic indicate better fit to the distribution.

about 13-14 nanoseconds to 24-25 nanoseconds. This is further reflected in the forward and reverse running averages which are widely spaced apart. The L10I/L90M mutant (top-left and top-right in figure 8.5) shows relatively consistent behaviour over the 50 nanoseconds. The V82A/I84V (middle-left and middle-right in figure 8.5) mutant shows considerably change in the enthalpic contribution over the entire run and a step-change in the entropic contribution at around 20 nanoseconds. The M46I/I54V mutant (bottom-left and bottom-right in figure 8.5) shows fairly constant behaviour with a step-change in the enthalpic contribution at about 41 nanoseconds.

Table 8.2 shows the Kolmogorov-Smirnov statistic for the ΔH and $-T\Delta S$ contributions to ΔG in comparison to a normal (Gaussian) distribution about the mean of the sample for snapshots from the single, 50 nanosecond trajectory's and from the 50 by 4 nanosecond ensembles. Lower values of the Kolmogorov-Smirnov statistic indicate better fit to the distribution (closer to the null hypothesis that the sample is drawn from the target distribution). From this data a number of important features may be determined. Firstly, the ensemble snapshots provide in general a better fit than the single trajectory snapshots, which would be expected due to the sampling characteristics of the two methods. Secondly the entropic contributions are a worse match to a normal distribution than the enthalpic ones. This is to be expected, because the normal mode process which calculates the entropic contributions contains a minimisation phase: a process which considerably deceases the configurational space sampled by the method.

8.3.2 Comparison with Experiment

Tables 8.3 and 8.4 show the mean enthalpic and entropic contributions to the binding affinity for the first 10, 20, 50 and the last 10 (i.e. 41 to 50) nanoseconds of the long time-scale runs, along with the standard deviation for those results. The standard deviation was used to calculate the standard error. Due to the difference in sampling frequency between MM-PBSA and NMODE, this leads to much larger errors in the entropic contributions provided by NMODE, in comparison to the errors for the enthalpic contributions provided by MM-PBSA. Table 8.5 shows a comparison of the computational values of ΔG for each of the six mutants and each of the four simulation time periods selected compared with the experimental values from Ohtaka This table illustrates that as single trajectory simulations progress, the value for ΔG converges towards the experimental value (a good example being the wild type) but for others it diverges (c.f. the quad mutant).

Figure 8.6 shows the same data as table 8.5 graphically. In this instance, the x-axis is the experimental value of ΔG and the y-axis the computational value. The magenta line $y = x$ on the graph represents “perfect correlation” between the experimental and computational value. This graph indicates considerable problems with this particular simulation of the M46I/I54V mutant which for all of the series appears to be more attractive than the wild type and also with the quad mutant which at some point undergoes a considerable change during the last ten nanoseconds drastically affecting its binding affinity. Apart from these problems it appears from this graph that a good ranking is obtained over the fifty nanoseconds (the blue line) with only the M46I/I54V mutant giving a consistently incorrect ranking. Visual inspection of the trajectory files in VMD showed evidence that some of the behaviour may be affected by the presence of water molecules in the active site.

Table 8.6 shows the mean ΔH and $-T\Delta S$ contributions from the fifty 4 nanosecond simulations making up the ensemble. Table 8.7 compares the experimental ΔG from Ohtaka with the value for ΔG calculated from long (50 nanosecond), single trajectory simulation and the ensemble systems. The single trajectory values for ΔG show considerable variation, being both more positive (HM, QM and V82A/I84V) and more negative (wild type, L10I/L90M and M46I/I54V) than the experimental values with the difference between the computational and experimental values varying between -6.8 kcal/mol and +3.7 kcal/mol (a range of 10.5 kcal/mol). In comparison, the ensemble values for ΔG are all more negative than the experimental values with a much smaller range of difference from the experimental values of 2.7 kcal/mol (between -6.5 kcal/mol and -3.8 kcal/mol). This implies that whatever experimental error exists in the ensemble systems is more consistent than the one in the single trajectory, most likely because the ensemble systems are not trapped in a single area of configurational space determined by the initial conditions.

Figure 8.7 shows the relative binding free energies ($\Delta\Delta G$) of the five mutants to lopinavir with respect to the wild type. Figure 8.8 shows the relative binding free energies ($\Delta\Delta G$) of the five mutants to lopinavir with respect to the wild type, but with the computational values only taking into account the enthalpic contribution to the relative binding free energy. This shows better ranking than figure 8.7 for the ensemble method. It provides a less satisfactory ranking than figure 8.7 for the single trajectory. In particular it is less easy to discriminate between the quad-mutant and the V82A/I84V mutant. Looking back to the entropic data for this trajectory in 8.5, there is a considerable change in the entropic term which, between 10 and 50 nanoseconds, cancels out the change in the enthalpic term and excluding this data is apparently having a negative effect on ranking ability. This is an effect on a single trajectory, based on a specific set of initial conditions which would have a smaller impact on an ensemble system.

Mutant	**First 10ns**	**First 20ns**	**50ns**	**Last 10ns**
Wild type	-52.2 (5.0)	-51.1 (5.2)	-50.1 (5.3)	-49.2 (5.3)
Hexa-mutant	-44.3 (6.4)	-43.9 (6.3)	-42.1 (6.3)	-41.4 (6.0)
Quad-mutant	-51.8 (5.4)	-50.0 (6.4)	-42.7 (8.4)	-37.3 (5.1)
V82A/I84V	-41.8 (9.2)	-39.4 (7.9)	-42.6 (7.4)	-45.5 (5.9)
M46I/I54V	-58.6 (4.7)	-57.2 (5.7)	-54.9 (5.7)	-48.4 (4.9)
L10I/L90M	-51.8 (5.2)	-50.9 (5.4)	-48.6 (5.5)	-46.5 (4.5)

Table 8.3: Mean values for ΔH in kcal mol^{-1} for the six different mutants over four different time periods with standard deviation (used to calculate error) in brackets.

Mutant	**First 10ns**	**First 20ns**	**50ns**	**Last 10ns**
Wild type	32.4 (5.5)	33.5 (6.1)	32.2 (6.1)	31.5 (5.8)
Hexa-mutant	34.2 (5.4)	34.0 (6.5)	33.4 (6.7)	34.2 (6.3)
Quad-mutant	35.6 (4.4)	33.9 (5.3)	33.5 (5.2)	33.8 (5.4)
V82A/I84V	28.7 (6.9)	28.0 (7.6)	32.3 (7.7)	32.0 (6.4)
M46I/I54V	32.0 (7.4)	31.5 (6.6)	33.3 (6.6)	34.7 (6.8)
L10I/L90M	30.6 (7.7)	30.8 (6.9)	32.0 (6.7)	29.6 (6.8)

Table 8.4: Mean values for $-T\Delta S$ in kcal mol^{-1} for the six different mutants over four different time periods with standard deviation in brackets.

Mutant	**First 10ns**	**First 20ns**	**50ns**	**Last 10ns**	**Experiment**
Wild type	-19.8	-17.6	-17.9	-17.6	-15.1
Hexa-mutant	-10.1	-9.9	-8.7	-7.2	-11.3
Quad-mutant	-16.3	-16.2	-9.1	-3.6	-12.8
V82A/I84V	-13.1	-11.3	-10.3	-13.4	-13.9
M46I/I54V	-26.5	-25.7	-21.7	-14.2	-14.9
L10I/L90M	-21.2	-20.2	-16.5	-16.9	-14.9

Table 8.5: Mean values for ΔG in kcal mol^{-1} for the six different mutants over four different time periods, compared with the values from Ohtaka [46].

Mutant	ΔH	$-T\Delta S$
Wild type	-52.5 (0.04)	32.3 (0.21)
Hexa-mutant	-48.8 (0.05)	31.5 (0.21)
Quad-mutant	-49.0 0.05)	32.42 (0.21)
V82A/I84V	-50.68 (0.04)	30.66 (0.22)
M46I/I54V	-52.3 (0.04)	32.2 (0.22)
L10I/L90M	-52.5 (0.04)	31.0 (0.22)

Table 8.6: Mean values for ΔH and $-T\Delta S$ in kcal mol^{-1} for the six different mutants over the fifty 4 ns replicas in the ensemble, each with a four nanosecond time-scale run.

Mutant	Experiment	Single trajectory	Ensemble
Wild type	-15.1	-17.9	-20.1
Hexa-mutant	-11.3	-8.7	-17.3
Quad-mutant	-12.8	-9.1	-16.6
V82A/I84V	-13.9	-10.3	-20.0
M46I/I54V	-14.9	-21.7	-20.1
L10I/L90M	-14.9	-16.5	-21.4

Table 8.7: Experimental values for ΔG from table 2 published by Ohtaka in [46] compared with computational values calculated from entropic and enthalpic contributions from a single 50 nanosecond simulation and an ensemble of 50x4 nanosecond trajectory runs for each of the six mutants of HIV-1 protease bound to lopinavir.

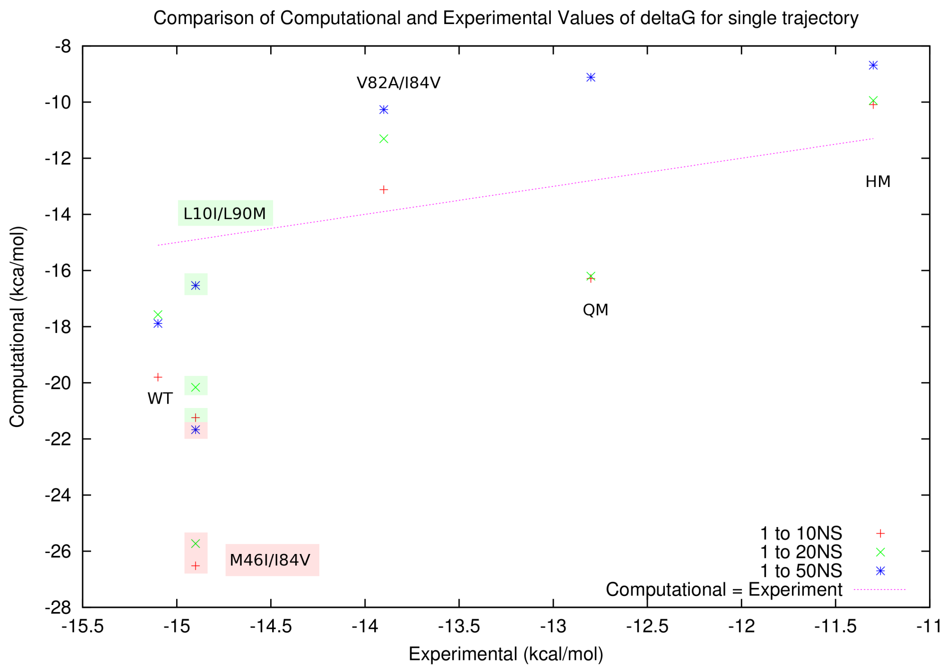


Figure 8.6: Comparison of experimental values with computational values for ΔG for the first 10 (red), 20 (green) and 50 (blue) nanoseconds of simulation time for each of the six mutants of HIV-1 protease bound to Lopinavir. The x-axis shows the experimental values for ΔG as provided by Ohtaka [46]. The y-axis shows the mean computational values over the run-time. Therefore a "good" result should resemble a straight line with a perfect result resembling the magenta line. Points for the L10I/L90M and M46I/I54V mutants are distinguished by pale green and pale red backgrounds respectively.

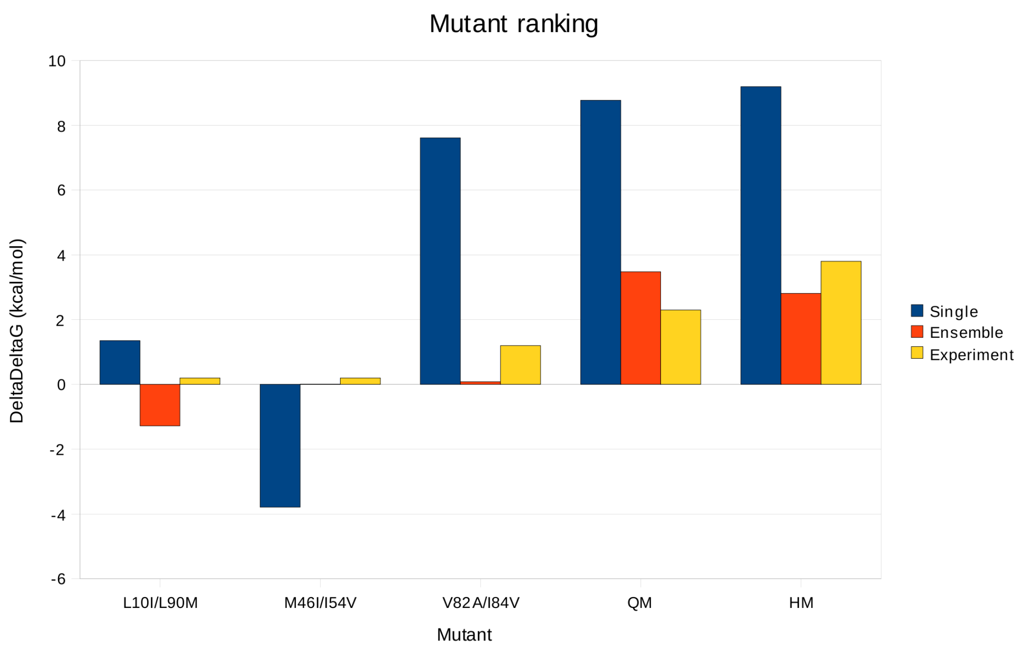


Figure 8.7: Relative binding free energies of the five mutants of HIV-1 protease to lopinavir with respect to the wild type, comparing computational values from long time-scale 50 nanosecond single trajectory runs (blue), fifty by 4 nanosecond ensemble runs (red) and the experimental values calculated from Ohtaka in [46] (yellow).

8.4 Discussion

Investigation of the binding affinity of lopinavir to the six selected mutants provided some promising results and insight into issues surrounding long time-scale molecular dynamics simulations.

The six mutants studied were chosen for a number of reasons. Firstly, they were chosen because they present a the option of investigating successive mutations (wild type, three pairs of mutations, one mutant with two pairs and one with all three) which are associated with increased resistance to a number of protease inhibitors.

In addition to investigating the binding affinity with Lopinavir, there are a number of other possible inhibitors to investigate including Amprenavir, Ritonavir, Indinavir and Nelfinavir. Experimental results for all four are available in Ohtaka for comparison. Earlier work done in the group shows promising results calculating binding affinities

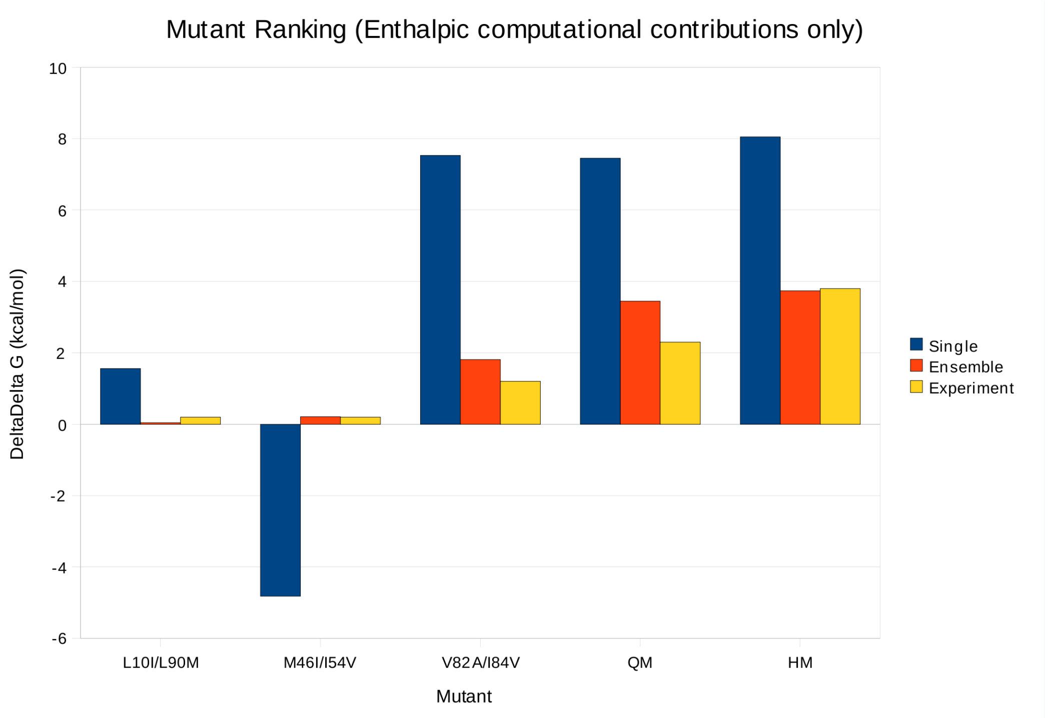


Figure 8.8: Relative binding free energies of the five mutants of HIV-1 protease to lopinavir with respect to the wild type, comparing computational values from long time-scale 50 nanosecond single trajectory runs (blue), fifty by 4 nanosecond ensemble runs (red) and the experimental values calculated from Ohtaka in [46] (yellow). Unlike figure 8.7 the computational results shown here only include the enthalpic contribution to the relative binding free energy. This produces a much better ranking than is shown in figure 8.7 for the ensemble method.

with the six mutants for a sixth inhibitor known as Saquinavir. Positive results for these inhibitors might mean that it is possible to get an estimated binding affinity by computational methods for other protease inhibitors when parameterisations of them for the Amber force-field become available.

The convergence study demonstrated a number of limitations of single trajectory investigations of binding affinity. The primary limitation was one of convergence, with the 50 nanosecond trajectories failing to converge to a reproducible result. This is most apparent in the forward and reverse cumulative averages, for example in for the V82A/I84V mutant in 8.5, whose separation demonstrates a lack of convergence.

Investigation of the distribution of ΔH and $-T\Delta S$ for the single trajectory and ensemble simulations demonstrated that the ensemble systems sampled the enthalpic contributions more effectively than the single trajectory with a much more normal distribution of ΔH. The entropic contributions did not form a normal distribution and this appears to be due to the minimisation step of the normal mode calculation limiting the exploration of configurational space to a sub-set of minima.

For the purposes of the Binding Affinity Calculator, the single trajectory results for Lopinavir are promising with four of the five mutants yielding consistently good ranking when considering their relative binding affinities to the wild-type. This is likely due to the simulations becoming trapped in local energy minima near to the start of the simulation and as a result failing to explore configurational space correctly. This means that even extending the simulations further is unlikely to yield better results. The results for the ensemble systems are even more promising, giving correct ranking for all five mutants when considering only the enthalpic contributions to the relative binding affinity $\Delta\Delta G$ and with more consistent difference in ΔG with respect to the experimental values.

8.5 Conclusions and future work

It is evident from the data presented here that computational methods for calculating relative binding affinities of HIV protease mutants to an inhibitor (in this case lopinavir) have potential future merit with the caveat that single trajectory, long time-scale simulations are a much less effective method of reproducing experimentally derived binding affinities than ensemble methods. The sampling of configurational space is greatly improved by choosing ensemble methods over single trajectory methods (although the latter does sometimes still provide a good ranking) and this means that the result is more reliable as the effect of one trajectory taking an unusual path though configurational space is considerably lessened.

Ensemble methods also have the potential for increasing the accuracy of the estimated result as larger parallel machines (or task farms of much smaller resources) by increasing the number of replicas in the ensemble whereas with a single long time-scale system, it is difficult to increase the accuracy of the simulation for three reasons. Firstly, as a molecular dynamics simulation progresses along a longer time-scale, the build-up of rounding errors, approximations and floating point errors mean that the results are a decreasing reflection of reality, and as with the results shown here, it is not clear that the reproduction of experimental results is improved at all. Secondly, increasing the length of a single trajectory is not a parallel process as each time-step depends on the previous ones. This means that increasing the length of the simulation increases the real-world time that the simulation phase takes. Although in the past it would have been assumed that the natural progression to faster and faster microprocessors would overtake this increase in run time, most of the efforts in improving the performance of a single processor currently focus around multiple cores and parallelisation. Thirdly, unlike an ensemble which is for most intents and purposes embarrassingly parallel, the speed-up obtained from increasing the number of cores available for a single trajectory molecular dynamics system is limited by the communications costs involved.

This difference in reliability means that for potential clinical use, long time-scale simulations do not appear particularly useful since the ensemble method gives better results and in terms of real-world time can be completed more quickly.

In order for these tools to be integrated into clinical practice a considerable amount of infrastructure would need to be put into place. Large computational resources would need to be set up with appropriate authentication and security to protect patient confidentiality. As computer technology progresses however, the relative scale of the resource needed decreases considerably. If simulation methods are proved to be effective then it is possible that they could become routine in supporting the decisions made by clinicians. Once it became possible for doctors to reliably obtain accurate results from the system within two weeks of the patient being genotyped, running a simulation to provide a result to inform treatment would become routine.

The ranking provided by the ensemble enthalpic contribution to the relative binding affinity provides an excellent and correct ranking. The number and size of simulations required means that the researcher is reliant on a very large amount of computational resource. In this instance terascale resources were used, but equally effective would be a large number of mid-scale resources. Ensemble methods map more easily onto these resources than single long simulations.

Future work should focus on a study to prove that the ensemble results are reproducible, and are replicated for other inhibitors.

To make this work actually useful to clinicians who have little expertise in managing simulations on a high performance resource (never mind the challenges of using a distributed, Grid infrastructure) then considerable effort needs to be made in developing the prototype tools described here into a user-friendly solution. The user must be presented with a simple interface where they enter the mutations present in the patient's strain(s) of the virus and are presented with the results (rankings) at a later date with all the complexity of the under-lying technological systems hidden from them. A workflow needs to be developed where the software constructs the systems and submits all the jobs to available resources, analyses that output and returns the results to the user without any intervention on their part. This will require considerable development effort but this is a software challenge rather than a science one.

Chapter 9

Conclusions and Future Work

9.1 Conclusions

In this thesis a number of important topics have been discussed in relation to applying Grid resources to help provide insight into biological processes.

By modifying an existing replica exchange molecular dynamics existing so that it efficiently exploits the heterogeneous nature of the communications infrastructure inherent in cross-site Grid computation, the potential of the Grid as an alternative to single large scale computational resources for some areas of biological molecular dynamics research. These methods apply to some other codes, and the replica exchange molecular dynamics code is itself being used in other fields. The code modifications to LAMMPS demonstrated here are not limited to the biomolecular sphere and are likely to be just as effective elsewhere.

An extensive replica exchange molecular dynamics study of the unliganded HIV-1 protease wild-type demonstrated enhanced sampling over traditional single-simulation molecular dynamics methods and showed what appear to be flap opening events in a number of the replicas.

Extensive investigation of the accuracy of a number of methods available to the Binding Affinity Calculator was carried out, determining that ensemble methods demonstrated a ranking of the computationally derived binding affinity of Lopinavir to six mutants of the protease that was closer to the experimentally derived values and ranking than individual simulations although considerable improvement in accuracy will be required before the tool is suitable for real-world patient-specific use.

9.2 Future work

The MPIg Grid-aware variant of LAMMPS is essentially complete although some work needs to be done moving the Grid code into the latest versions of LAMMPS. Further work is planned to use SAGA to produce a replica exchange version of LAMMPS with transparent access to disparate Grid resources independent of the middle-ware layers on the available resources. It seems likely that this will involve moving the replica exchange code out of LAMMPS and into a separate wrapper in order to free it from some of the constraints it currently suffers from. There is already a replica exchange SAGA framework for NAMD developed by the SAGA developers which has showed considerable promise in this area.

As the technical work above continues, more work needs to be carried out investigating the flap behaviours of various mutants of HIV-1 protease, and additionally the behaviours when bound to a ligand. The latter will either require the parametrisation of Lopinavir into the Charm force-field, code modification to fix problems with the tool to import Amber force-fields into LAMMPS or else the use of the replica-exchange version of NAMD.

Further work needs to be carried out investigating more thoroughly the issues apparent in the BAC's estimates of binding affinity, in particular more work needs to be done in understanding the reproducibility issues the method faces. A proper reproducibility study is ongoing and the initial results go some way to indicating methods of improving the accuracy and reproducibility of the tool.

Bibliography

[1] *"MPI: A Message-Passing Interface Standard"*, The Message Passing Interface Forum (http://www.mpi-forum.org/), 1995

[2] *"MPI-2: Extensions to the Message-Passing Interface"*, The Message Passing Interface Forum (http://www.mpi-forum.org/), 1997

[3] N. T. Karonis, B. Toonen, I. Foster *"MPICH-G2: A Grid-Enabled Implementation of the Message Passing Interface"*, Journal of Parallel and Distributed Computing, Vol. 63, No. 5, pp. 551-563, May 2003

[4] E. Gabriel, M. Resch, T. Beisel, R. Keller *"Distributed computing in a heterogeneous computing environment"*, EuroPVMMPI'98 Liverpool/UK, 1998

[5] J. Perry, A. Jackson, L. Smith, S. Booth *"QCDGrid: A grid resource for Quantum Chromodynamics"*, UK e-Science All Hands Conference, Nottingham, September 2003

[6] I. Foster *"Globus Toolkit Version 4: Software for Service-Oriented Systems"*, FIP International Conference on Network and Parallel Computing, Springer-Verlag LNCS 3779, 2-13, 2005

[7] P. V. Coveney, R. S. Saksena, S. J. Zasada, M. McKeown, S. Pickles, *"The application hosting environment: Lightweight middleware for grid-based computational science"*, Computer Physics Communications 176 (6) 406418, March 2007

[8] J. Chin, P. V. Coveney *"Towards tractable toolkits for the Grid: a plea for lightweight, usable middleware"*, National e-Science Centre Technical Paper UKeS-2004-01, 2004

[9] Yunhong Gu, Xinwei Hong, R. L. Grossman *"Experiences in Design and Implementation of a High Performance Transport Protocol"*, presented at SC 2004, Nov 6 - 12, Pittsburgh, PA, USA., 2004

[10] S. Booth. *"Grid metacomputing support on HPCx"*, private communication, originally to be released as an HPCx Technical Report in 2006

[11] B. M. Boghosian, L. I. Finn, P. V. Coveney *"Moving the data to the computation: multi-site distributed parallel computation"*, URL: http://www.realitygrid.org/publications/GD3.pdf, January 2006

[12] S. Manos, M. Mazzeo, O. Kenway, N. T. Karonis, B. Toonen and P. V. Coveney, *"Distributed mpi cross-site run performance using mpig."*, Proceedings of High Performance Distributed Computing Boston, Massachusetts, USA, June 23-27 2008

[13] J. MacLaren, M Mc Keown. *"HARC: A Highly-Available Robust Co-scheduler"*, URL: www.realitygrid.org/publications/HARC.pdf, January 2006

[14] J. Gray and L. Lamport, *" Consensus on Transaction Commit"*, ACM Transactions on Database Systems, Vol 31, No 1, Pages 133-160, March 2006

[15] K. Yoshimoto, P. Kovatch and P. Andrews *"Co-scheduling with User-Settable Reservations"*, Lecture Notes in Computer Science, Springer Berlin / Heidelberg, ISSN 0302-9743 (Print) 1611-3349 (Online), Volume 3834, 2005

[16] S. J. Plimpton *"Fast Parallel Algorithms for Short-Range Molecular Dynamics"*, J. Comp. Phys. 117, 1-19, 1995

[17] S. J. Plimpton, R. Pollock ans Mark Stevens *"Particle-Mesh Ewald and rESPA for Parallel Molecular Dynamics Simulations"*, in proceedings of the Eighth SIAM

Conference on Parallel Processing for Scientific Computing, Minneapolis, MN, March 1997

[18] H. C. Greenwell, W. Jones, P. V. Coveney and S. Stackhouse *"On the application of computer simulation techniques to anionic and cationic clays: A materials chemistry perspective"*, Journal of Materials Chemistry, 16, (8), 708-723, 2006

[19] D. A. Case, T.E. Cheatham, T. Darden, H. Gohlke, R. Luo, K.M. Merz, Jr., A. Onufriev, C. Simmerling, B. Wang and R. Woods. *"The Amber biomolecular simulation programs"*, J. Comput. Chem. 26, 1668-1688, 2005

[20] J. W. Ponder and D. A. Case. *"Force fields for protein simulations"*, Adv. Prot. Chem. 66, 27-85, 2003

[21] T. E. Cheatham and M.A. Young. *"Molecular dynamics simulation of nucleic acids: Successes, limitations and promise"*, Biopolymers 56, 232-256, 2001

[22] J. Wang, R. M. Wolf, D. A. Case, P. A. Kollman, *"Development and testing of a general AMBER force eld (GAFF)"*, J. Comput. Chem. 25, 11571174, 2004

[23] Y. Duan, C. Wu, S. Chowdhury, M. C. Lee, G. Xiong, W. Zhang, R. Yang, P. Cieplak, R. Luo, T. Lee, *"A point-charge force eld for molecular mechanics simulations of proteins based on condensed-phase quantum mechanical calculations"*, J. Comput. Chem. 24, 19992012, 2003

[24] B. R. Brooks, R. E. Bruccoleri, B. D. Olafson, D. J. States, S. Swaminathan and M. Karplus, *"CHARMM: A program for macromolecular energy, minmimization, and dynamics calculations"*, J. Comput. Chem. 4:187217, 1983.

[25] Chia-En Chang, Tongye Shen, Joanna Trylska, Valentina Tozzini and J. Andrew McCammon, *"Gated Binding of Ligands to HIV-1 Protease: Brownian Dynamics Simulations in a Coarse-Grained Model"*, Biophysical Journal Volume 90, 3880-3885, June 2006

[26] A. F. Voter *"Hyperdynamics: Accelerated Molecular Dynamics of Infrequent Events"*, Phys. Rev. Lett. Vol 78, Number 20, 3908-3911, 19 May 1997

[27] G Bussi, A Laio, M Parrinello, *"Equilibrium free energies from nonequilibrium metadynamics"*, Phys. Rev. Lett. Vol 96, Number 9, 10 March 2006

[28] A. M. A. West, R. Elber, and D. Shalloway *"Extending molecular dynamics time scales with milestoning: Example of complex kinetics in a solvated peptide"*, J. Chem. Phys 126, 145104, 2007

[29] O. A. Kenway *"Benchmarking Molecular Dynamics codes on the IBM e-Server Blue Gene system"*, EPCC MSc. Dissertation, 2005.

[30] S. A. Adelman *"Generalized Langevin theory for many-body problems in chemical dynamics: General formulation and the equivalent harmonic chain representation"*, J. Chem. Phys. 71, 4471, 1979

[31] S. A. Nosé *"A molecular-dynamics method for simulations in the canonical ensemble"*, Molecular Physics 52(2):255268, 1984

[32] W. G. Hoover *"Canonical dynamics - equilibrium phase-space distributions"*, Phys. Rev. A 31(3):16951697, 1985

[33] J. C. Phillips, R. Braun, W. Wang, J. Gumbart, E. Tajkhorshid, E. Villa, C. Chipot, R. D. Skeel, L. Kale, and K. Schulten. *"Scalable molecular dynamics with NAMD"*, J. Comput. Chem. 26:1781-1802, 2005

[34] S. Wan, P. V. Coveney and D. R. Flower, *"Peptide recognition by the T cell receptor: comparison of binding free energies from thermodynamic integration, Poisson-Boltzmann and linear interaction energy approximations"*, Phil. Trans. R. Soc. A., 363 (1833), 2037-2053, 2005

[35] P. A. Kollman, I. Massova, C. Reyes, B. Kuhn, et. al. *"Calculating Structures and Free Energies of Complex Molecules: Combining Molecular Mechanics and Continuum Models"*, Acc. Chem. Res. 33, 889-897, 2000

[36] W. Wang, P. A. Kollman, *"Computational study of protein specificity: The molecular basis of HIV-1 protease drug resistance"*, PNAS December 18, vol. 98 no. 26 1493714942, 2001

[37] M. Frigo and S. G. Johnson, *"The Fastest Fourier Transform in the West"*, MIT technical report MIT-LCS-TR-728, September 1997

[38] R. H. Swendsen and J. S. Wang, *"Replica Monte Carlo Simulation of Spin-Glasses"*, Phys. Rev. Lett. Vol 57, Number 21, 2607-2609, 24 November 1986

[39] A. Okur, L. Wickstrom, M. Layten, R. Geney, K. Song, V. Hornak and C. Simmerling, *"Improved Efficiency of Replica Exchange Simulations through Use of a Hybrid Explicit/Implicit Solvation Model"*, J. Chem. Theory Comput. 2, 420-433, 2006

[40] D. J. Earl and M. W. Deem, *"Parallel Tempering: Theory, Applications, and New Perspectives"*, arXiv:physics/0508111 v2 19, August 2005

[41] C. J. Woods, M. H. Ng, S. Johnston, S. E. Murdock, B. Wu, K. Tai, H. Fangohr, P. Jeffreys, S. Cox, J. G. Frey, M. S. P. Sansom and J. W. Essex, *"Grid computing and biomolecular simulation"*, Phil. Trans. R. Soc. A 363, 20172035, 2005

[42] V. Zeote, O. Michielin and M. Karplus, *" Relation between Sequence and Structure of HIV-1 Protease Inhibitor Comlpexes: A Model System for the Analysis of Protein Flexibility"*, J. Mol. Biol. 315, 21-52, 2001

[43] J. Snoek, C. Riva, K. Steegen, Y. Schrooten, B. Maes, L. Vergne, K. V. Laethem, M. Peeters and A. M. Vandamme, *"Optimization of a genotypic assay applicable to all human immunodeciency virus type 1 protease and reverse transcriptase subtypes"*, Journal of Virological Methods 128:4753., 2005

[44] V. Hornak, A. Okur, R. C. Rizzo, and C.Simmerling *"HIV-1 protease flaps spontaneously open and reclose in molecular dynamics simulations"*, PNAS 2006;103;915-920; originally published online Jan 17, 2006

[45] K. Bush *“Screening and Characterization of Enzyme Inhibitors as Drug Candidates”*, Drug Metabolism Reviews Vol. 14, No. 4, Pages 689-70, 1983

[46] H. Ohtaka, A. Schon, and E. Freire, *“Multidrug Resistance to HIV-1 Protease Inhibition Requires Cooperative Coupling between Distal Mutations”*, Biochemistry, 42, 13659-13666, 2003

[47] S. K. Sadiq, D. Wright, S. J. Watson, S. J. Zasada, I.Stoica and P. V. Coveney *“An automated molecular simulation-based binding affinity calculator for ligand-bound HIV-1 proteases”*, Submitted to Journal of Chemical Information and Modeling

[48] H. J. C. Berendsen, J. P. M. Postma, W. F. van Gunsteren, A. DiNola, J. R. Haak, *“Molecular dynamics with coupling to an external bath”*, J. Chem. Phys. 81, 36843690., 1984

[49] H. L. Sham, D. J. Kempf, A. Molla et. al. *“ABT-378, a Highly Potent Inhibitor of the Human Immunodeficiency Virus Protease”*, Antimicrobial Agents and Chemotherapy, 32183224 Vol. 42, No. 12, December 1998

[50] S Walmsley, B Bernstein, M King, J Arribas et. al. *“Lopinavir-Ritonavir versus Nelfinavir for the Initial Treatment of HIV Infection”*, N. Engl. J. Med. 2039-2046 Vol. 346, No. 26 June 27, 2002

[51] S. K. Sadiq, S. Wan, and P. V. Coveney, *“Insights into a Mutation-Assisted Lateral Drug Escape Mechanism from the HIV-1 Protease Active Site”*, Biochemistry, 46, (51), 14865-14877, 2007

[52] I. Stoica, S. K. Sadiq, and P. V. Coveney, *“Rapid and Accurate Prediction of Binding Free Energies for Saquinavir-Bound HIV-1 Proteases”*, J. Am. Chem. Soc. 130, 2639-2648, 2008

[53] W. L. Jorgensen, J Chandrasekhar, J. D. Madura,R. W. Impey, M. L. Klein, *“Comparison of simple potential functions for simulating liquid water”*, J. Chem. Phys. 79, 926935., 1983