

# Development and deployment of an application hosting environment for grid based computational science

P. V. Coveney, M. J. Harvey, and L. Pedesseau

*Centre for Computational Science, Department of Chemistry,  
University College London, Christopher Ingold Laboratories,  
20 Gordon Street, London WC1H 0AJ, United Kingdom*

D. Mason and A. Sutton

*Department of Physics, Imperial College, Exhibition Road, London SW7 2AZ*

M. McKeown and S. Pickles

*Manchester Computing, Kilburn Building, The University of Manchester, Oxford Road, Manchester M13 9PL*

The Application Hosting Environment (AHE) is designed to help the scientist who wants to use Grid [1–3] computing by providing a higher level abstraction of the Grid than is currently offered by existing Grid middleware such as Globus [4]. Rather than providing capabilities for submitting generic computational jobs to a Grid we provide the scientist with services that will start and manage the applications he wants to use on his behalf; we do this by building a layer of middleware on top of the existing grid middleware. By providing this extra level of abstraction the scientist does not need to know the details of any particular grid middleware, so we can isolate him from any changes to the underlying grid middleware used by a particular grid.

Keywords:

## I. INTRODUCTION

We define grid computing as distributed computing performed transparently across multiple administrative domains. Transparency has been lacking from more or less all existing distributed computing infrastructures which might aspire to call themselves grids. A general computational grid should provide facile access to many different types of resources to enable users to pick and choose those required to achieve their intended scientific objectives. However, such demands have proved hard to meet to date and as a result very few scientists have wanted to get near so-called grid technology. We describe how a lightweight web-service based hosting environment residing in a WSRF::Lite [15] container forms a potentially powerful tool for the computational scientist interested in engaging in activities worthy of the name grid computing. The design of the hosting environment is influenced by allied work on WEDS (WSRF-based Environment for Distributed Simulation) [5], a hosting environment designed for operation primarily within a single administrative domain.

The flexibility and utility of the hosting environment is illustrated by the example of the Polysteer program developed as part of the Reality-Grid project. Polysteer is a new Monte Carlo (MC) program for simulation of polymers operating in a steered heterogeneous network. We exploit the fa-

cility of deploying several independent instances of a Monte Carlo simulation using the hosting environment and then use a number of clients to monitor properties and to steer the simulation.

## II. THE APPLICATION HOSTING ENVIRONMENT

The AHE is an ensemble of programs written in Perl. The purpose of the AHE is to provide a mechanism for deploying applications onto a computational grid that makes it easy for the scientist to start the application on that grid. By providing a service that will launch a particular application rather than a generic computational job it is possible to reduce the complexity of the client and make the scientist's life easier. The scientist can concentrate on science rather than spending time understanding and installing middleware. The AHE stores all the necessary information about how an application should be run on the various computational resources of a grid and provides a uniform interface to the client for running that application across those resources. This is particularly useful for applications that run on supercomputers which often have unique deployment scenarios for an application and require special runtime environments for each system.

The AHE is developed with WSRF::Lite, a Perl implementation of the Web Service Resource Frame-

work (WSRF [14]). Each time an application is started a WS-Resource [14] is created that is used to represent that instance of the application's execution. This WS-Resource provides an interface for the user to interact with that instance of the application; through the WS-Resource the user can query the status of the application, change its run status and so on. The WS-Resource will still exist after the application has finished, providing information on the location of any output files.

To ensure that it is possible to develop simple and lightweight clients for the AHE a number of constraints were set on the design. We assume that the user is using either a laptop or PDA (Personal Digital Assistant) over wireless that is using NAT (Network Address Translation [16]) and that the device is firewalled to only allow outgoing connections. This effectively means that the client cannot receive connections: all interactions must be initiated by the client. Furthermore, we assume that the user may use multiple clients to interact with the same WS-Resource, requiring that the client maintain no state. We do not require the user to install Globus (or any other grid middleware) on the client machine even if the grid uses Globus as its middleware. For simplicity we require that the client only supports HTTP [8], HTTPS [9] and SOAP [6].

Figure 1 shows the architecture of the AHE. The 'App Server Registry' maintains a registry of all the application services that are available. For each different application that has been deployed on the grid there will be an entry in the 'App Server Registry'. The client can query the registry to find the address for the service that provides a particular application; once he has found the address it can be cached for future use. Unlike the analogous WEDS broker, the registry does not perform the second task of (client-delegated) service creation which must in this case be performed directly by the client.

The client invokes the 'Prepare' operation on the 'App Server Factory' service, this creates a WS-Resource that represents the instance of the application. The 'App Server Factory' returns a WS-Addressing [21] End Point Reference (EPR) to the client which the client uses to communicate with the WS-Resource. The client invokes the 'Submit' operation on the WS-Resource to start the application. For each application the format of the 'Submi' message will be different and for simple applications is may only contain the input file needed to run the application. For more complex applications that require multiple input files or input files that are too large to put into a SOAP message we use either the 'File Staging Area' or the 'FileStore'.

The 'File Staging Area' (FSA) is designed to allow the a client to use HTTPS to 'POST' a file to Web Server so that it can be downloaded by some

other service on the grid. This is designed to handle the case where a user has input files on his client machine and he needs to transfer them to the grid. From the FSA the WS-Resource for a particular instance of an application can access the necessary input files. The client can retrieve the output files from the FSA when the AHE has moved them there after the application has completed. We consider this type of file transfer as 'pass by value' since the client actually transports the file to the 'File Staging Area'.

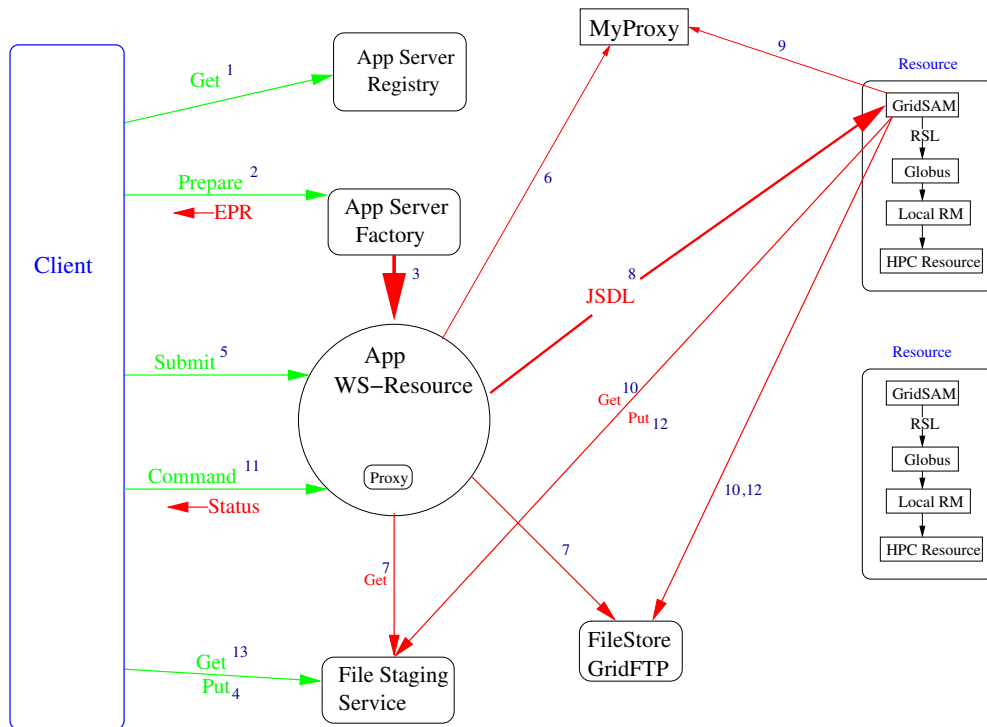
The 'FileStore' is any place on the grid where files required by the instance of the application are available through GridFTP. The 'FileStore' is used to hold large files like checkpoints that would not normally be stored on the client machine. The client passes a URL [31] to the file to the WS-Resource representing the instance of the application. We call this 'pass by reference'.

The AHE accesses the computational resources through GridSAM [12] hosted in the OMII [13] Web Service Container. GridSAM consumes Job Submission Description Language (JSDL [32]) documents and submits the job to the local resource manager. GridSAM has a plug-in architecture that allows adaptors for different types of resource manager to be used; for example, the adaptor could support Globus allowing GridSAM to submit jobs to a resource on a Globus grid as illustrated in the Figure 1. The AHE creates JSDL using the input submitted by the client and using data it stores about each resource and submits it to the GridSAM instance for a particular resource. GridSAM provides a layer of abstraction that allows the AHE to support any type of resource manager as long as there is a GridSAM adaptor for it.

By contrast, WEDS WS-Resources directly represent jobs co-located on the hosting resource and have no ability to interact with GridSAM services. Unlike WEDS, the AHE has no native support for computational steering [41], delegating this functionality to the RealityGrid steering library [33].

### III. SECURITY

Security is critical in any grid computing environment. We chose not to use WS-Security [18] for the AHE because message level security is not required as we do not have any SOAP intermediaries. Throughout we use HTTPS to provide Transport Layer Security (TLS [17]); mutual authentication is used between the client and the AHE using X.509 digital certificates [7]. We provide the Application WS-Resource with a proxy certificate [10] using MyProxy [11]. The Application WS-Resource can use the proxy certificate for authentication with any services it needs to communicate with on behalf



**FIG. 1:** The Architecture of the Application Hosting Environment. The numbers indicate the general sequence of messages, green arrows show messages initiated by the client. The red dashed line shows the separation between the services the client interacts with and the set of services that the client does not directly interact with. In this case the AHE is providing access to resources that are using Globus as their resource manager

of the client. The choice of MyProxy was governed by the fact that GridSAM uses it.

This represents a significant advance in comparison to WEDS which, whilst using TLS, performs neither credential authentication nor authorisation, severely limiting its deployment across multiple administrative domains.

#### IV. WSRF::LITE

WSRF::Lite is a Perl implementation of the Web Service Resource Framework specification which is proceeding through the OASIS [29] standardization process. WSRF::Lite is the follow-on from OGSF::Lite, the Perl implementation of the Open Grid Service Infrastructure (OGSI [19]) specification from GGF [20]. It is built on SOAP::Lite [30], the Perl module for Web Services from which it derives its name.

WSRF::Lite provides support for WS-Addressing, WS-ResourceProperties [22], WS-ResourceLifetime [23], WS-ServiceGroup [24] and WS-BaseFaults [25]. It also provides support for digitally signing SOAP messages using X.509 digital certificates in accordance with the OASIS

WS-Security standard and the OASIS X.509 Token Profile [26].

Reflecting the flexible philosophy and nature of Perl, WSRF::Lite allows the developer to host WS-Resources in a number of ways: using the Apache Web server [27], using the WSRF::Lite Container or with a simple stand-alone script. Developers also have a choice as to how they want to handle the state of their WS-Resource: using system memory, with a file or using a database. As well as providing support for WS-Security, WSRF::Lite also supports transport layer security through HTTPS.

For the AHE we host the WS-Resources using the Apache Web Server and store any state in a MySQL [28] database. As well as providing a secure and efficient container for our WS-Resources Apache also allows us to host the WS-Resources in a Web farm providing fault tolerance in the case of failure of a single Web server. The database storing the state could also be replicated to improve fault tolerance.

## V. THE POLYSTEER CODE

Polysteer is a prototypical example of a research code which exploits the opportunities offered by the description of the application as an interactive web resource rather than as an old-style monolithic batch program. It is a new code intended to develop and test new scientific theories in the field of polymer research. All run parameters which would normally be passed to such an application through a launch script are made accessible for modification through a steering GUI. We have shown [35] how this steering functionality offers a systematic procedure for quite general optimisation of the efficiency of Monte Carlo programs.

Regardless of whether a Monte Carlo simulation appears to require little or great compute resources, the confidence we can have in predictions based on the properties of the configurations generated is strongly dependent on the number of independent configurations found. This number is increased by running a single simulation for a longer time, or by running a number of independent trials. Trivial parallelisation by task-farming is therefore always beneficial. However, if it is possible to interact with a parallel job without excessive demands on the scientist, much more can be gained from task-farming. We show in section VD how the parameterisation of the Monte Carlo moves may be optimised by steering such a simulation.

A single master thread is used to issue commands to and collect simple statistical information from a number of slaves. If listener sockets are opened on the master thread, then the master operates as a single server, to which multiple client applications can be attached and detached as necessary. For our Polysteer application, we use the RealityGrid steering library [33] to open and operate data IO sockets on the master thread, and use three clients: a visualisation client to give an instant real-time 3D representation of the state of the system; a simple statistical client which calculates the density profile of the system to check that relaxation is occurring correctly, and a steering client which we use to monitor and manipulate the progress of the simulation. We can therefore make our Monte Carlo simulation into a grid application using the AHE [42], and it becomes a simple matter to launch a parallel set of simulations on an available cluster or supercomputer, and attach and detach clients running on local machines as desired.

### A. Modelling a polymer backbone as a space curve

In this section we set out the model of a polymer used. The rotational isomeric state model popularised by Flory and others represents the configuration of a polymer backbone by the rotation about each backbone bond [34]. These angles form a set of generalised coordinates, which while extremely compact are far from ideal if one wants to consider the van der Waals attraction and repulsion between chain segments, as the positions of the chain segments must be calculated by adding together bond vectors. Worse, some considerable work needs to be put into generating a set of Monte Carlo moves from direct manipulation of these generalised coordinates, as any bond-rotation move is inherently non-local unless effort is put into re-establishing the chain along the original path.

We greatly simplify modelling the backbone by drawing a hypothetical space curve through the centres of the bonds on the backbone, and manipulating the parameterisation of this space curve. We choose to represent the space curve as a cubic spline, which has the properties that

- it is uniquely defined by the position of its knots and the tangents to the curve at the ends.
- the curve, the tangent to the curve and the curvature are continuous along its length.

We can therefore fairly easily define a set of unbiased Monte Carlo moves which manipulate the spline curve and then map a polymer backbone back to the new position of the spline. Moves are accepted or rejected based on the change in energy of the backbone. We have previously used this approach to model polystyrene [35]. In this paper we take the simplification one level further and represent the backbone simply as a fixed length section of the spline. This is therefore similar in spirit to a Kradky-Porod worm-like chain model, although here the curvature is not constrained.

### B. The Monte Carlo Moves used

The arc length of the spline curve and that of the polymer backbone need not be identical, and we project the spline curve past the ends of the polymer by a few knots. This effectively defines a path for the polymer beyond the limits of its backbone, which we can use as a reptation path. If the condition that enough knots be seen each side of the polymer is breached by this move, then a reptation of the spline curve is performed, with a new knot being placed so

that the curvature in the new end segment is constant – this ensures no refitting is required. For our wormlike model only one parameter is required, the arc length to move the backbone. We found using the online optimisation procedure discussed below that the optimum shift is a random distance up to half the knot spacing. Reptation moves have been used in MC simulation of polymer melts since the 1980s [36].

A second move is the change of tangent of the ends of the spline curve. The spline curve must be refitted after such a change in tangent. However, as the spline curve is projected beyond the end of the backbone this move has a small effect on the conformation of the polymer and its acceptance rate is correspondingly high – for the simulations in section VD it averaged 0.48.

The third move affecting the backbone is the movement of a single knot. This is an internal configuration-changing move analogous to and inspired by the conformal rotation CONROT move. The knot is moved around an imagined circle perpendicular to the chord joining the adjacent knots. The spline curve must be refitted to the new knot positions after such a move, and the backbone refitted to the spline curve. While it is not necessary to constrain the angles or the maximum displacement, we added the condition that  $\cos \theta > -0.25$  to prevent the ends of the spline become too tightly folded.

### C. The System modelled

The system modeled comprised four cubic spline curves. The curves were around 50 knots in length, with mean arc length between knots of unity; making a chain sufficiently long to become entangled, but still small enough to be responsive in real time to changes in the parameterisation of the Monte Carlo moves. Interactions were considered to exist only in a fixed length of 40 units in the centre of each spline.

A van der Waals interaction between curve segments was introduced using a simple Lennard Jones 6-12 potential, with length scale  $\sigma = 1/2$ . The energy scale was set to unity. Curve segments interact with those on other splines and those on the same spline greater than an arc length of 1 unit.

The energy due to bending the spline is given by  $E_{bend} = \alpha \int (1/R^2) dl$ , where the integral runs over the length of the interacting central region of the spline;  $\alpha$  is the product of the second moment and Young's modulus and was here set to 0.02, representing a flexible chain such as polyethylene. The temperature scale was set to  $k_B T = 1$ .

A further potential was introduced at the boundary  $z = 0$ , to represent the attractive interaction of a surface with the chains. This potential was set to

be the same as the Lennard-Jones potential between chains. In the  $x$  and  $y$  directions the simulation box had periodic boundaries and was 16 units across; a hard wall was generated by the potential at  $z = 0$  and no boundary was set for large  $z$ .

### D. Monte Carlo simulations

Before any statistical sampling can be performed, the polymer system must be equilibrated. This is far from trivial with polymer simulations as the relaxation modes must include correlated movements of the backbone, and a number of algorithms for generating densely packed polymer conformations have been proposed [37–40]. Our approach to generating conformations is based on slowly growing the chains, attempting to keep them relaxed during the growth process using the backbone MC moves. We started with chains of length 50 units (10 spline knots), and increased the length of the chains by 10 units (2 knots), then relaxed to a binding conformation. For the small systems used here this procedure is found to be easily efficient enough to grow equilibrated systems within half an hour of wall clock time.

When we had generated enough independent starting configurations, we used the Monte Carlo moves described in subsection VB to sample conformations of the chain. The model was run concurrently on eight processors. A screenshot of the operation of the Polysteer program is shown in figure 2.

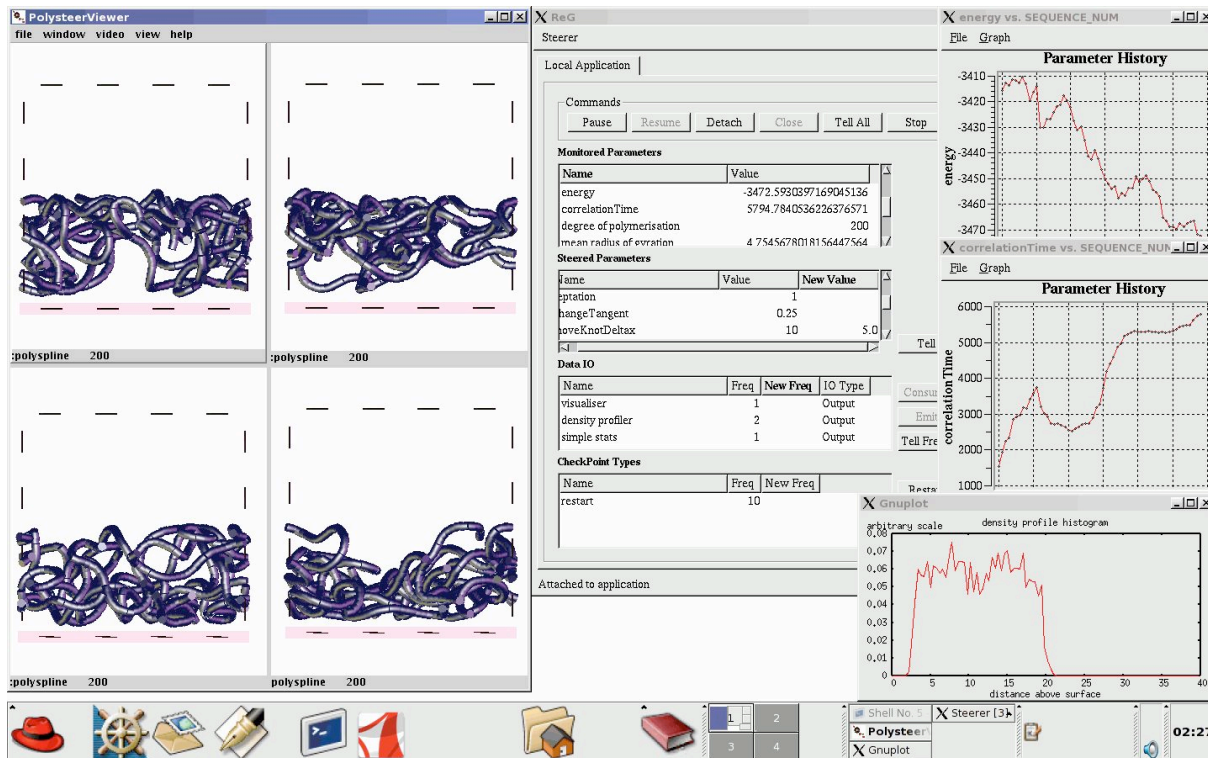
We have used the clients to understand the effect of changing the parameterisation of the Monte Carlo moves. An inner product can always be defined which will indicate the degree of similarity of one configuration to another. If we regularly store in memory snapshots of the configuration, we can look at the similarity of the current configuration to those previously seen, and from this determine a rate of movement through configuration space. If the system is moving steadily through its configuration space, then we expect that a suitably defined inner product is changing as

$$S(t) \cdot S(t - \tau) \approx \exp(-\lambda\tau) \quad (1)$$

where  $S(t)$  is some system property at time  $t$ . We require only that  $S(t) \cdot S(t) = 1$  and  $S(t) \cdot S(t') \geq 0$  for the purposes of the fitting. The position of spline  $i$  is written  $x_i(l)$ , where  $l$  is the position along the arc. We have chosen

$$S(t) \cdot S(t') = \frac{1}{N} \sum_{i=1}^N \frac{1}{L_i} \int_{l=0}^{L_i} \exp\left(-\frac{|x_i(l, t) - x_i(l, t')|}{R_g}\right) \quad (2)$$

where  $R_g$  is the mean radius of gyration of the polymers in the system. The integral is estimated by a discrete approximation.



**FIG. 2:** A screenshot of the Polysteer program and the monitoring of multiple independent Monte Carlo simulations. Here four MC simulations are running remotely on a cluster, and three clients are attached. The four threads are being visualised, a steering client is used to adjust parameters, and a third client is used to monitor the density profile of the films being modelled to check the systems are not detaching from the surface. In the visualisation the system is viewed along the x-axis, with the z-axis vertical.

We can therefore monitor a single parameter, the decorrelation time, defined as  $T \equiv 1/\lambda$ , which is small when the system is moving quickly through configuration space, and is large when the system is moving slowly. If changing a Monte Carlo parameter reduces the decorrelation time then the change is beneficial and should be accepted. In figure 3 we plot how this decorrelation time changes when the relative probability of selecting knot-move and reptation trial moves is adjusted. It is evident that an equal selection probability of each type of move is the most efficient combination.

The error bars plotted indicate the standard deviation of the eight independent results being combined by the master thread. If only a single instance of the Monte Carlo simulation were being monitored, the fluctuation in the correlation time would be so great that no useful information can be extracted. The time is measured in wall clock seconds, and figure 3 indicates that it takes about an hour to decorrelate

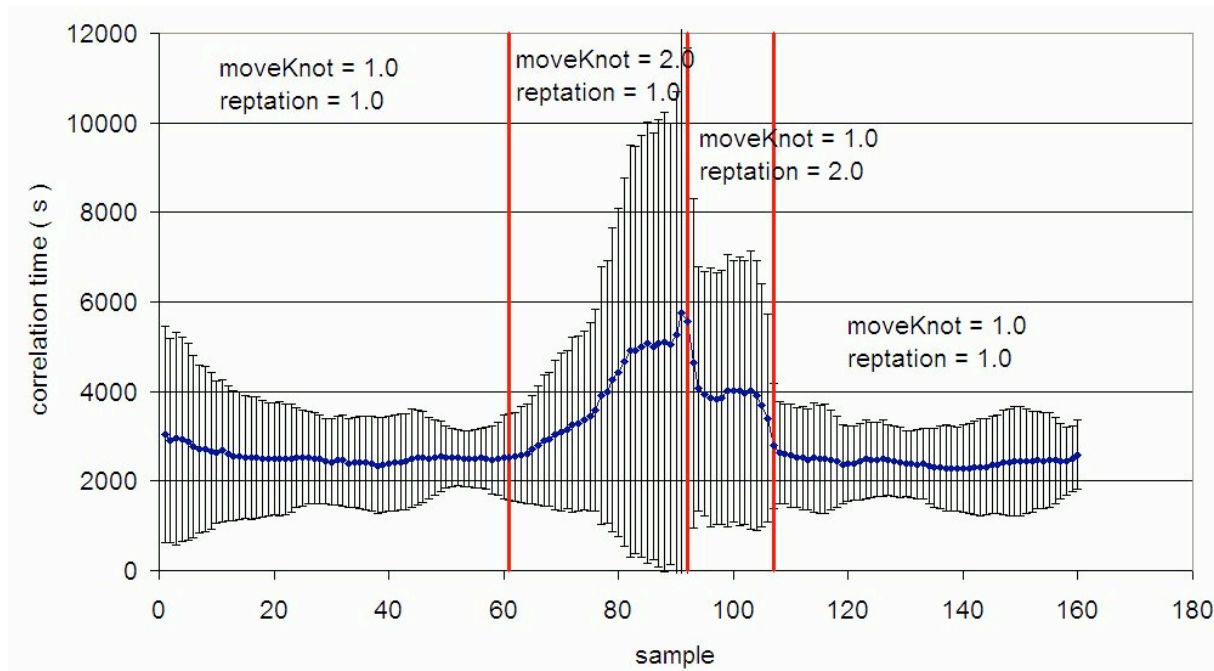
the system, giving a characteristic time for independent configurations to be drawn.

## VI. PLANS AND DISTRIBUTION

The Application Hosting Environment is being developed initially for deployment within the SPICE project, the UK component of the Joint US/UK High End computing Project 2005. It will be distributed via OMII in late 2005. WEDS is available from <http://www.realitygrid.org>.

## VII. ACKNOWLEDGEMENTS

This work is supported by EPSRC through the projects *RealityGrid* (GR/R67699) and *Rapid Prototyping Of Usable Grid Middleware* (GR/T27488/01) and through the OMII Managed Programme project *Rapid Application Hosting using WSRF::Lite*.



**FIG. 3:** The decorrelation time changes when adjustments are made to the parameterisation of the Monte Carlo moves. Here the relative probability of making a reptation trial move and moving a single knot are being adjusted. In normal operation this time is a monitored parameter, and can be graphed by the steering client. The error bars indicate the standard deviation of the eight independent results being combined.

- [2] Berman, F., Fox, G. C., Hey, A. J. G. Grid Computing: Making the Global Infrastructure a Reality, Wiley, 2003
- [3] P. V. Coveney (ed) Scientific Grid Computing, Phil Trans R Soc Lond A Vol. 363, 15 August 2005.
- [4] The Globus Alliance <http://www.globus.org>
- [5] P. Coveney, J. Vicary, J. Chin, M. Harvey, Introducing WEDS: a WSRF-based environment for distributed simulation, Scientific Grid Computing, P. V. Coveney (ed) Phil Trans R Soc Lond A Vol. 363, 15 August 2005. doi:10.1098/rsta.2005.1608.
- [6] Simple Object Access Protocol (SOAP) 1.1, <http://www.w3.org/TR/soap>
- [7] Internet X.509 Public Key Infrastructure Certificate Management Protocols, IETF RFC 2510, <http://www.faqs.org/rfcs/rfc2510.html>
- [8] Hypertext Transfer Protocol - HTTP/1.1, IETF RFC 2616, <http://www.w3.org/Protocols/rfc2616/rfc2616.html>
- [9] HTTP Over TLS, IETF RFC 2818, <http://www.faqs.org/rfcs/rfc2818.html>
- [10] Internet X.509 Public Key Infrastructure (PKI) Proxy Certificate Profile, IETF RFC 3820, <http://www.faqs.org/rfcs/rfc3820.html>
- [11] J. Basney, M. Humphrey, and V. Welch, The MyProxy Online Credential Repository, (2005), Software: Practice and Experience. <http://www.ncsa.uiuc.edu/~jbasney/myproxy-spe.pdf>
- [12] <http://www.lesc.ic.ac.uk/gridsam/>
- [13] <http://www.omii.ac.uk/>
- [14] Web Service Resource 1.2, S. Graham, A. Karmarkar, J. Mischkin, I. Robinson and I. Sedukin (2005), [http://docs.oasis-open.org/wsr/wsr/wsr\\_resource-1.2-spec-pr-01.pdf](http://docs.oasis-open.org/wsr/wsr/wsr_resource-1.2-spec-pr-01.pdf)
- [15] <http://www.sve.man.ac.uk/Research/AtoZ/ILCT>
- [16] The IP Network Address Translator (NAT), IETF RFC 1631, <http://www.faqs.org/rfcs/rfc1631.html>
- [17] The TLS Protocol Version 1.0, IETF RFC 2246, <http://www.faqs.org/rfcs/rfc2246.html>
- [18] A. Nadalin, C. Kaler, P. Hallam-Baker and R. Monzillo, Web Services Security: SOAP Message Security V1.0, (2004), OASIS Standard 2000401.
- [19] S. Tuecke, K. Czajkowski, I. Foster, J. Frey, S. Graham, C. Kesselman, T. Maquire, T. Sandholm, D. Snelling and P. Vanderbilt, Open Grid Service Infrastructure, (2003), <http://www.gridforum.org/documents/GFD.15.pdf>
- [20] Global Grid Forum, <http://www.ggf.org/>
- [21] M. Gudgin and M. Hadley, Web Service Addressing 1.0 - Core, (2005), <http://www.w3.org/TR/2005/WD-ws-addr-core-20050331/>
- [22] S. Graham and J. Treadwell, Web Services Resource Properties 1.2, (2005), [http://docs.oasis-open.org/wsr/wsr/wsr\\_resource\\_properties-1.2-spec-pr-01.pdf](http://docs.oasis-open.org/wsr/wsr/wsr_resource_properties-1.2-spec-pr-01.pdf)
- [23] L. Srinivasan and T. Banks, Web Services Resource Lifetime 1.2, (2005), [http://docs.oasis-open.org/wsr/wsr/wsr\\_resource\\_lifetime-1.2-spec-pr-01.pdf](http://docs.oasis-open.org/wsr/wsr/wsr_resource_lifetime-1.2-spec-pr-01.pdf)
- [24] T. Maguire and D. Snelling, Web Services

- Service Group 1.2, (2005), [http://docs.oasis-open.org/wsrif/wsrif-ws\\_service\\_group-1.2-spec-pr-01.pdf](http://docs.oasis-open.org/wsrif/wsrif-ws_service_group-1.2-spec-pr-01.pdf)
- [25] L. Liu and S. Meder, Web Services BaseFaults 1.2, (2005), [http://docs.oasis-open.org/wsrif/wsrif-ws\\_base\\_faults-1.2-spec-pr-01.pdf](http://docs.oasis-open.org/wsrif/wsrif-ws_base_faults-1.2-spec-pr-01.pdf)
- [26] , P. Hallam-Baker, C. Kaler, R. Monzillo and A. Nadalin , Web Services Security: X.509 Token Profile V1.0(2004), <http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-profile-1.0.pdf>
- [27] <http://httpd.apache.org/>
- [28] <http://www.mysql.com/>
- [29] <http://www.oasis-open.org/home/index.php>
- [30] <http://soaplite.com/>
- [31] Uniform Resource Locators (URL), IETF RFC 1738, <http://www.faqs.org/rfcs/rfc1738.html>
- [32] Job Submission Description Language (JSDL) Specification, Version 1.0, GGF, <https://forge.gridforum.org/projects/jsdl-wg/document/draft-ggf-jsdl-spec/en/21>
- [33] <http://www.sve.man.ac.uk/Research/AtoZ/RealityGrid>
- [34] P. J. Flory, Statistical Mechanics of Chain Molecules Hanser, Munich, (1989).
- [35] D. R. Mason and A. P. Sutton in Scientific Grid Computing, P. V. Coveney (ed) Phil Trans R Soc Lond A Vol. 363, 15 August 2005.
- [36] M. Vacatello , G. Avitabile , P. Corradini and A. Tuzi, J. Chem. Phys., 73:548–552, (1980).
- [37] M. Müller , J. Nievergelt , S. Santos and U. W. Suter, J. Chem. Phys., 114:9764–9771, (2001).
- [38] D. N. Theodorou and U. W. Suter, Macromolecules 19, 139 (1985).
- [39] W. Brostow and J. Kubat, Phys. Rev. B 47, 7659 (1993).
- [40] A. A. Gusev, M. M. Zehnder, and U. W. Suter, Macromolecules 27, 615 (1994).
- [41] J. Chin, J. Harting, S. Jha, P. V. Coveney, A. R. Porter, S. M. Pickles, Contemp. Phys. 44, 417–434 (2003).
- [42] Work done hitherto has in fact used the simpler WEDS system with security provided by the OMII container and was demonstrated at the EPSRC e-Science meeting 21-22 April 2005 at the National e-Science Centre, Edinburgh