Parameter Space Exploration Using Scientific Workflows

David Abramson, Blair Bethwaite, Colin Enticott,
Slavisa Garic, and Tom Peachey

Faculty of Information Technology,
Monash University,
Clayton, 3800, Victoria, Australia
{david.abramson,blair.bethwaite,colin.enticott,
slavisa.garic,tom.peachey}@infotech.monash.edu.au

Abstract. In recent years there has been interest in performing parameter space exploration across “scientific workflows”, however, many existing workflow tools are not well suited to this. In this paper we augment existing systems with a small set of special “actors” that implement the parameter estimation logic. Specifically, we discuss a set of new Kepler actors that support both complete and partial sweeps based on experimental design techniques. When combined with a novel parallel execution mechanism, we are able to execute parallel sweeps and searches across workflows that run on distributed “Grid” infrastructure. We illustrate our new system with a case study in cardiac cell modelling.

Keywords: distributed workflows, parameter exploration, Kepler.

1 Introduction

In recent years there has been a great deal of interest in “scientific workflows” [10]. These allow scientists to specify large computational experiments involving a range of different activities, such as data integration, modelling and analysis, and visualization. Activities can be composed, often using a graphical programming environment, so that the output of one stage can be passed as input to the next, forming a pipeline of arbitrary complexity. Scientific workflow engines manage the execution across a range of distributed resources, and leverage Grid computing middleware and approaches [1][2][3]. For example, it is possible to extract data from a scientific instrument, pass it through some analysis software running on a high performance cluster, store the results in a distributed data repository, and then visualize it on a large display wall. Scientific workflows have been used to great effect in a number of different disciplines including Computational Chemistry [7], Ecology[8] and Bioinformatics [9].

Importantly, many workflow engines double as programming environments for the Grid. Whilst there are no standard ways of doing this, a number of engines effectively expose the Grid middleware APIs. For example, Kepler [5] exposes a variety of middleware layers, from Globus through to ad-hoc interfaces like SSH. Other engines, such as Triana [19] and Taverna [9] allow users to invoke services as Web Services, but provide no explicit support for Grid middleware.

Prior to the wide adoption of workflow engines, we developed a family of tools, called Nimrod, for performing parameter sweeps with computational models. Nimrod supports the execution of a specific type of workflows in which a single computation is performed multiple times to allow exploration of some design space. Nimrod includes tools that perform a complete parameter sweep across all possible combinations (Nimrod/G) [4], or search using non-linear optimization algorithms (Nimrod/O) [20] or experimental design techniques (Nimrod/E) [21]. Importantly, the degree of parallelism, can be varied at run time, as the Nimrod scheduler places tasks on the available resources then. Nimrod had been applied to a wide range of disciplines from public health policy to quantum chemistry [22].

However, Nimrod was not designed to execute arbitrary workflows of the type discussed above. Thus, it is difficult to run sweeps *over* workflows, and workflows *containing* sweeps. Likewise, most workflow systems do not support the parallel execution of tasks that are supported in Nimrod, and are not well suited to parameter sweeps and searches.

In this paper we discuss how we have added parameter sweeps and searches to existing workflow tools. We focus on the techniques that allow a range of scenarios to be explored by adding a few simple components to an existing workflow. Specifically, we discuss the design of a new family of Kepler “actors” that support sweeps across parameter ranges. The resulting system is extremely flexible, and allows the creation of decision support systems of arbitrary complexity.

The paper begins with a discussion of parameter sweeps and the techniques that are commonly used, illustrating the ideas through the Nimrod family of tools. We then discuss workflow engines, and in particular, we describe Kepler. We then introduce a new family of actors, and show how these can be combined with existing workflows. Finally, we demonstrate the applicability of the ideas using a case study in cardiac modelling.

2 Parameter Sweeps

2.1 Full Parameter Sweeps

Many current scientific and engineering problems can be formulated as computational models with a great deal of accuracy. Changing the inputs allows a user to explore a range of design scenarios, giving a picture of how the system behaves. This is typically performed as a sweep over the input parameters. Although the model may be computationally expensive, parallel execution of the jobs can dramatically speed up the execution and allow very large systems to be studied.

In this context, it is useful to think of a given computational model as a function that accepts a set of input parameters and produces a set of outputs. Input parameters are typically simple types, such as integers, floats and text strings. A full parameter sweep takes all combinations of the parameters, and allows exploration of the entire design space within some finite resolution. Such parameter sweeps have been used very effectively, for example in environmental modelling [23], bioscience [24], engineering [20] and chemistry [25].

2.2 Partial Parameter Sweeps

Although the favoured method for exploring the parameter space of a model is a full parameter sweep, this may be impractical where there are many parameters, especially if the model is computationally intensive. Experimental design techniques may enable meaningful results to be obtained from a suitably chosen subset of jobs.

The most widely used method of experimental design is the fractional factorial design [11]. Suppose the experiment has input parameters (called factors in the literature) $A, B, C, \dots$ and produces a numerical output $f(A, B, C, \dots)$. The underlying model is that f is the sum of several terms, a constant term f_0 , then the “main effect” terms $f_1(A), f_2(B), \dots$, each dependent on only one factor, then “two-factor interaction effects” terms, $f_{12}(A, B), f_{13}(A, C), f_{23}(B, C), \dots$, each dependent on two factors, then “three-factor interactions”, and so on. Long experience, with both physical and computer-based experiments, suggests that the higher-order terms, interactions of three factors and more, are usually negligible. In that case, only the lower-order terms need to be estimated, so fewer jobs are needed to obtain these estimates and hence obtain reasonable approximations to the output. Significant savings are possible; with 20 parameters, each at two levels, the number of jobs required is reduced from over a million to 512.

2.3 The Current Nimrod Tool Set

Over the past 15 years we have constructed a tool set called Nimrod [4], that automates both of the parameter sweep techniques discussed above. One of the Nimrod tools, Nimrod/G, supports complete parameter sweeps. It operates either as a tool, or a middleware layer in its own right. If the former, then it is usually operated from a Web portal, and this allows a user to create a plan file, set up a testbed, manage certificates etc., and organize input and output files from an experiment. Nimrod/G, however, can also serve as a job management system for other software, including the other members of the Nimrod family. This structure is shown in Fig. 1.

The Nimrod/E tool, on the other hand, automates the design of fractional factorial experiments. Here, the user specifies the factors and which interactions can be ignored. Then, one component produces an efficient design generating the parameter values for the resulting jobs in a form suitable for Nimrod/G. When all jobs are complete, another component produces analyses of the results for each output value of interest. It creates graphs showing the relative size of the various main effects and interactions. The Daniel Plot [13], Fig. 6, plots the effects so that the negligible ones, with values just experimental error, will form the central straight line, significant effects produce points that deviate from this line. The Lenth Plot [14], Fig. 5, shows effects in order of absolute size and horizontal lines giving significance levels. Points outside the outer lines are most probably significant. The tool also estimates results of a full parameter sweep, which may then be used by visualization software.

Fig. 1. Nimrod tool chain

3 Workflow Engines and Kepler

There has been considerable interest in scientific workflow engines over the past few years, in particular, in Grid workflow engines. Workflows allow a user to build arbitrary computations from a set of connected components, or *actors*. Actors may represent computations, but also facilitate access to distributed databases and scientific instruments. In most systems, data moves along the edges that connect the actors as *tokens*, and can thus be streamed from instruments and databases, through a range of computational processes. Grid workflows allow these actors to be executed on distributed resources, and launched in a variety of ways, facilitating virtual applications that span multiple organizations, data sources and computers.

To date, many scientific workflow tools have been built [6][9][10]. Grid workflow systems allow general applications to be constructed, with examples ranging from ecology to medical imaging. In this project we focus on Kepler [5][6][7][8], which allows a user to weave a “virtual” application from a set of otherwise distinct components, and its workflow engine orchestrates their execution in a controlled and repeatable manner. Kepler builds upon Ptolemy II [11], a Java-based software framework with a graphical user interface called Vergil. Ptolemy II is used for the modelling, simulation, and design of concurrent, real-time embedded systems. The focus is on assembly of concurrent components. Kepler inherits a number of orchestration mechanisms from Ptolemy, providing an extensive range of execution mechanisms. These are controlled through devices called “directors”.

In spite of their significant power, Kepler, and many other current workflow systems, do not support dynamic parallel execution of a workflow and its components. This means that users must explicitly code a workflow to cause it to execute elements in parallel – either by replicating the workflow statically, or adding looping constructs that scatter and gather threads. Both of these techniques significantly complicate the workflow and obscure the underlying business logic. In another recent paper, we have

shown how to augment Kepler with a Tagged Data Flow Architecture Director (TDA) [18]. This new system, called Nimrod/K, extends Kepler by providing powerful mechanisms for exposing and managing parallelism in the workflows. This provides an ideal platform for using workflows for parameter sweeps. Unlike the current Nimrod tool chain, Nimrod/K makes it possible to run sweeps over workflows, and workflows that contain sweeps. It leverages Kepler’s power in building complex workflows, and Nimrod’s ability to execute sweeps over grid resources. In the next section we show how we have created some specific new Kepler actors that facilitate parameter sweeps. When combined with Nimrod/K’s parallel execution mechanisms, we have a powerful new tool for parameter sweeps.

4 New Kepler Actors

4.1 Actor Design

The Nimrod/G Actor. As discussed in section 2.3, the current Nimrod tools provide parameter sweep tools to complement Nimrod/G’s Grid execution and meta-scheduling capabilities. One of these tools performs a parameter sweep that creates a Grid job for each of the parameter combinations in the parameter space. Each of these parameter combinations is represented in Kepler using datasets, called “tokens”. We have created a *ParameterSweep* actor, that uses the same parameter syntax as the Nimrod/G implementation, to generate parameter combinations in a workflow. This actor works with the current Ptolemy directors, however, when used in combination with the Tagged Dataflow mechanism in Nimrod/K, parallel execution of the different parameter combinations allows the workflow to execute efficiently on the Grid.

The new *ParameterSweep* actor is quite general. For example, multiple actors can be chained together to create a sub parameter sweep inside a larger parameter sweep. Although it is possible to create such a sweep with a single *ParameterSweep* actor, staging them allows for preliminary calculations of shared data to be performed by the parent sweep prior to the start of the sub-sweep. Also, chaining them together enables a more dynamic parameter range options, for example, if the sub sweep is calculated from the parameters given by the parent sweep. This environment produces sweeps that meet the parameter sweep requirements of workflows.

Nimrod/E Actors. The workflow for a typical Nimrod/E experiment is shown in Fig. 2. The user enters details of the parameters to be varied and the effects to be estimated, into the *FractionalFactorialDesigner* actor. This designs the experiment and specifies the jobs required, passing these job specifications (as workflow tokens) to the actor(s) that perform the model execution. If the model is a simple formula then the model execution may be performed by a Kepler numerical actor. More complex modelling may require execution of an external model via the Nimrod/K framework.

When all jobs are complete, downstream processing can begin. There may be several numerical aspects of the model output of interest, so at this stage the dataflow may branch. Fig. 2 shows the case where there are two outputs requiring analysis.

Fig. 2. Nimrod/E experiment

One output from each model is needed to estimate the effects for that output. Tabulation of the effects may be sufficient, or these estimates may be used as inputs for a Daniel plot, Lenth plot and/or computation of the full sweep results.

4.2 Implementation

Nimrod/G. The *ParameterSweep* actor is built on the PtolemyII source class that provides an output port and a trigger. The output port sends a *RecordToken* containing the value of each parameter for a single parameter combination. The actor has a property that contains the parameter sweep string identical to the syntax of the parameter sweep commands in the Nimrod plan file syntax. To implement the token tagging functions, the *ParameterSweep* actor is built on the classes and functions provided by the Nimrod/K TDA director. These classes have no functional affect when used with other Ptolemy directors. Under the Nimrod/K director each parameter combination is executed in parallel. This is further discussed in [18].

Nimrod/E. The actors for Nimrod/E were built using the same principles as the *ParameterSweep* actor. Because *nimrodFracDes* is computationally intensive there is a clear advantage in retaining the C code for this function. Further, the code is quite complex, so that re-writing in Java within the Kepler framework would be a large undertaking. Consequently, we decided to use the existing C code, using a Java wrapper to produce a design actor. The information that it produces as a plan file is redirected as a tag to the output token for downstream processing. This actor is named the *FractionalFactorialDesigner* in Fig. 2.

For the analysis section most computation is relatively light. Further, the graphs produced would benefit from a re-coding in Java, using the Kepler graphing functionality. When there are a large number of factors, the plots produced may display estimates for many effects and be quite cluttered; Kepler graphs provide zooming to allow the user to explore the fine detail. On the other hand, the production of the analysis matrix does require some of the complex computation used in *nimrodFracDes*. Accordingly, we decided to recode the analysis section within the Kepler framework. The *EffectEstimator*, *LenthPlotter*, *Daniel-Plotter* and *FullSweepInterpolator* actors were added to perform the analysis discussed in Section 2.3.

5 Case Study

In this section we illustrate the new actors we have discussed to date, using a case study previously performed with the Nimrod/G and Nimrod/E tools. The case study allows us to demonstrate the utility of the new approach and its expressive power in Kepler.

Fig. 3. Complete parameter sweep

Fig. 4. Experimental Design

Mathematical models of the heart show considerable promise for understanding the underlying mechanisms and for clinical diagnosis and treatment [15][12]. The model chosen [16] concerns excitation and contraction in rabbit ventricular muscle cells. Intracellular flows of calcium, sodium and magnesium ions were modelled as a system of ordinary differential equations using Matlab. Earlier experiments with this model are reported in [17]. Fig. 3 shows the Kepler workflow used to perform a parameter sweep over the nine factors, *A* to *J*. This used two values for each factor, producing an experiment of 512 jobs, and the values are generated using the *ParameterSweep* Actor.

A second experiment used the experimental design functionality to further investigate the response function. The *FractionalFactorialDesign* actor in the workflow shown in Fig. 4 was used to produce a design that would estimate the main effects and two-way interactions of these factors. This required only 128 jobs.

Fig. 5 shows the Lenth plot from the experiment. Points for the effects of all nine inputs lie outside the confidence lines and hence all inputs make a significant

Fig. 5. Lenth plot

Fig. 6. Daniel plot

contribution. Of the interactions, only, *AB*, *DE*, *GH*, *GJ* and *HJ* are definitely significant, the other interactions are indistinguishable from noise. This reveals the structure of the computational model. The final output is the sum of four terms, the first affected only by *A* and *B*, the second by *C*, the third by *D*, *E* and *F* and the last by *G*, *H* and *J*, explaining why many interactions have no effect. However the lack of significant interactions between *D* and *F*, and between *E* and *F*, requires further investigation. Fig. 6 shows the same results on a Daniel plot. Here the deviation of points from the vertical line gives a measure of the significance of the effects plotted.

Fig. 7 shows the parallelism in the complete sweep when executed under the Nimrod/K Director. Our cluster was saturated at just over 100 concurrent executions. The system takes some time to build up the parallelism, and this is evident in the initial ramp up phase. This is caused by the overheads in starting remote computations using the Globus framework.

This case study demonstrates that Kepler provides a natural and easy mechanism for specifying the use of parameter sweep techniques over existing workflows. We have used a fairly simple computation model in this paper to clearly show the techniques, but this does not really demonstrate the significant potential of the approach.

Fig. 7. Parallelism in complete sweep

All of the machinery developed to date is capable of sweeping across parameter combinations regardless of the complexity of the computational steps of the workflow. Thus, much more complex pre-existing workflows can be modified using our new actors to perform large-scale complex parameter sweep experiments. We have recently begun work on modifying a complex computational chemistry workflow by adding the actors discussed in this paper.

6 Conclusions

In this paper we have discussed the design and implementation of some new actors that facilitate parameter sweeps in scientific workflows. The solution builds on the existing execution frameworks in workflow systems, and we have demonstrated its applicability in Kepler. The simple case study shows how easy it is to create and execute a sweep over an existing scientific computation.

Our system leverages a separate addition to Kepler, called Nimrod/K, which supports parallel execution of the different instances. By combining the new Actors, and

the new TDA Director, we can execute each of the parameter combinations in parallel. This makes it possible to compute complex design experiments quickly if there are sufficient resources available.

Acknowledgements

The Nimrod/K project is supported by Australian Research Council ARC Discovery grant. We are grateful to our colleague, Ilkay Altintas, from the San Diego Supercomputing Center for her assistance, and Anushka Michailova and Saleh Amirriazi, from the UC San Diego, for their work on the cardiac model.

References

1. Foster, I., Kesselman, C.: *The Grid: Blueprint for a New Computing Infrastructure*. Morgan Kaufmann, San Francisco (1999)
2. Foster, I., Kesselman, C.: Globus: A Metacomputing Infrastructure Toolkit. *Int'l J. of Supercomputer Applications* 11(2), 115–128 (1997)
3. The Globus Toolkit, <http://www-unix.globus.org/toolkit/>
4. Abramson, D., Giddy, J., Kotler, L.: High Performance Parametric Modeling with Nimrod/G: Killer Application for the Global Grid. In: *Int'l. Parallel and Distributed Processing Symposium (IPDPS)*, Cancun, Mexico (2000)
5. Kepler Project, <http://kepler-project.org>
6. Ludäscher, B., Altintas, I., Berkley, C., Higgins, D., Jaeger-Frank, E., Jones, M., Lee, E., Tao, J., Zhao, Y.: Scientific Workflow Management and the Kepler System. In: *Concurrency and Computation: Practice & Experience, Special Issue on Scientific Workflows* (2005)
7. Altintas, I., Birnbaum, A., Baldridge, K., Sudholt, W., Miller, M., Amoreira, C., Potier, Y., Ludaescher, B.: A Framework for the Design and Reuse of Grid Workflows. In: Herrero, P., Pérez, M.S., Robles, V. (eds.) *SAG 2004. LNCS*, vol. 3458, pp. 120–133. Springer, Heidelberg (2005)
8. Altintas, I., Berkley, C., Jaeger, E., Jones, M., Ludäscher, B., Kepler, M.S.: Towards a Grid-Enabled System for Scientific Workflows. In: *the Workflow in Grid Systems Workshop in GGF10 - The 10th Global Grid Forum*, Berlin (2004)
9. Taverna Project, <http://taverna.sourceforge.net>
10. Yu, J., Buyya, R.: A Taxonomy of Workflow Management Systems for Grid Computing. Technical Report GRIDS-TR-2005-1, Grid Computing and Distributed Systems Laboratory, Univ. of Melbourne (2005)
11. Box, G.E., Hunter, W.G., Hunter, J.S.: *Statistics for Experimenters*. Wiley, Chichester (1978)
12. Amirriazi, S., Chang, S., Peachey, T., Abramson, D., Michailova, A.: Optimizing Cardiac Excitation-Metabolic Model By Using Parallel Grid Computing. In: *Biophysics 2008*, Long Beach, California (2008)
13. Daniel, C.: *Applications of Statistics to Industrial Experimentation*. Wiley, Chichester (1976)
14. Lenth, R.V.: Quick and Easy Analysis of Unreplicated Factorials. *Technometrics* 31, 469–473 (1989)

15. Hunter, P.J., Kohl, P., Noble, D.: Integrative Models of the Heart: Achievements and Limitations. *Philosophical Transactions of the Royal Society of London: Mathematical, Physical and Engineering Sciences*, 2001, 10.1098/rsta.2001.0816 (2001)
16. Michailova, A., Lorentz, W., McCulloch, A.: Modelling Transmural Heterogeneity of KATP current in Rabbit Ventricular Myocytes. *AJP-Cell Physiology*, 293, C542–C557 (2007)
17. Amirriazi, A., Chang, S., Peachey, T., Abramson, D., Michailova, A.: Optimizing Cardiac Excitation-Metabolic Model By Using Parallel Grid Computing. In: *Biophysics 2008*, Long Beach, California (February 2008)
18. Abramson, D., Enticott, C., Altinas, I.: Nimrod/K: Towards Massively Parallel Dynamic Grid Workflows. In: *IEEE Supercomputing 2008*, Austin, Texas (November 2008)
19. Taylor, I., Shields, M., Wang, I.: Resource Management of Triana P2P Services. In: *Grid Resource Management*. Kluwer, Netherlands (2003)
20. Abramson, D., Lewis, A., Peachey, T., Fletcher, C.: An Automatic Design Optimization Tool and its Application to Computational Fluid Dynamics. In: *SuperComputing 2001*, Denver (November 2001)
21. Peachey, T.C., Diamond, N.T., Abramson, D.A., Sudholt, W., Michailova, A., Amirriazi, S.: Fractional Factorial Design for Parameter Sweep Experiments using Nimrod/E. *Journal of Scientific Programming* 16(2,3) (2008)
22. Message Lab list of current and former e-Science projects,
<http://www.messagelab.monash.edu.au/EScienceApplications>
23. Lynch, A.H., Abramson, D., Beringer, K.J., Uotila, P.: Influence of savanna fire on Australian monsoon season precipitation and circulation as simulated using a distributed computing environment. *Geophys. Res. Lett.* 34, L20801 (2007)
24. Sher, A., Abramson, D., Enticott, C., Garic, S., Gavaghan, D., Noble, D., Noble, P., Peachey, T.: Incorporating local Ca²⁺ dynamics into single cell ventricular models using Nimrod/O. In: Bubak, M., van Albada, G.D., Dongarra, J., Sloot, P.M.A. (eds.) *ICCS 2008, Part I. LNCS*, vol. 5101, pp. 66–75. Springer, Heidelberg (2008)
25. Sudholt, W., Baldridge, K., Abramson, D., Enticott, C., Garic, S.: Parameter Scan of an Effective Group Difference Pseudopotential Using Grid Computing. *New Generation Computing* 22, 125–135 (2004)