17th European Symposium on Computer Aided Process Engineering – ESCAPE17
V. Plesu and P.S. Agachi (Editors)

Experience on gridification and hyper-infrastructure experiments in optimization and process synthesis

Du Du, Siyu Yang, Antonis C. Kokossis and Patrick Linke

Chemical and Process Engineering, University of Surrey, GU2 7XH, U.K., a.kokossis@surrey.ac.uk

Abstract

The systematic development of new products and processes relies upon the extensive experimentation with simulation and optimization models. Integration between stages is currently manual as resources are heterogeneous (experimental databases, PDFs, computer models, property and cost data, software, technical reports, and images) and widely distributed (separate or distant R&D groups sharing complementary or similar knowledge and expertise). Grids and hyper-infrastructure environments offer particularly attractive technologies with a potential to enable integrated applications and the design of distributed experiments in industrial environments. The paper addresses the design of synthesis experiments in hyper-infrastructure clusters that support GT4.0. The experiments are designed around homogeneous sections of the Markov chains and are distributed over using advanced middleware and upperware (Superscalar and Gridway). The implementation enables the visualization of results and the analysis of solutions.

Keywords

Grid, distributed computing, optimization, Grid middleware, Globus toolkits, Gridway, Grid Superscalar

1. Introduction

The systematic development of new products and processes relies upon the extensive experimentation with simulation and optimization models. Integration between data generation and analysis, knowledge acquisition and pre-conceptual screening and application is currently manual as resources are heterogeneous and widely distributed. Practices are limited due to the lack of computing power, and a similar lack of integration in the underlying methods and resources.

Grids are clusters of resources, middleware and upperware with an objective to function as a stable, dependable, secure and efficient environment to apply distributed computing. (I. Foster, 2002). Grids are designed to share heterogeneous resources and, although they function as self-contained systems that are integrating and are coordinating resources, every node of the Grid remains independent. In the form of hyper-infrastructures, Grids are customized to provide services to businesses and technical applications, making use of additional service layers between distributed computing and users. Figure 1 highlights standard connections between servers and computing resources as well as the flow of computations being launched by users and undertaken by hardware. Grids and hyper-infrastructure environments offer particularly attractive technologies with a potential to enable integrated applications and the design of distributed experiments in industrial environments.

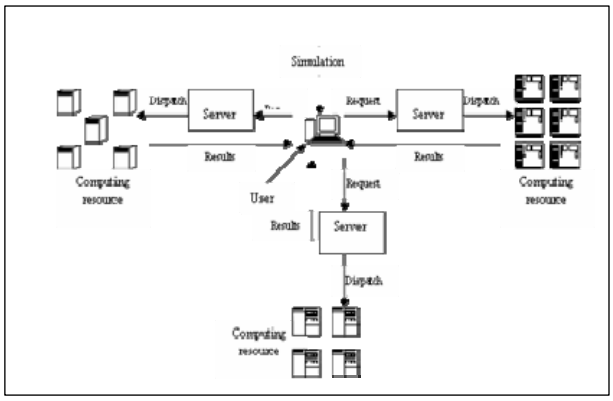

Figure 1 the Architecture of an Engineering Grid

2. Background

Grid technologies originated as meta-computing and involved applications that were built directly on Internet protocols. With the emergence of Open Grid Services Architecture

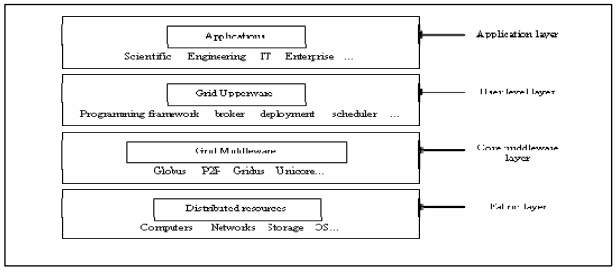

Figure 2 Four layers of a Grid

(OGSA) and shared virtual systems, Grids promise to provide application-oriented frameworks. Users access the resources remotely and through the employment of information service and resource broker layers. Large-scale databases at disposal generally help the grid system operate efficiently. End-users do not have to know or care where the tasks are performed or where their data is stored.

The purpose of middleware is intended to enable remote access to the instrumentation and to support the virtual environment. From bottom-up, Figure 2 explains that the Grid fabric stands for; (i) globally distributed resources; (ii) the core middleware that provides core service such as allocation, management, discovery and security; (iii) the user level middleware that comprises programming tools, resource brokers and deployment devices; and (iv) applications (I. Foster, 2005).

From a computer science the challenges are to address the refinement of the OGSA model, to build additional building block services, and to develop compilation and execution tools for Grid environments (debuggers, monitors and libraries; scale the Grid to larger numbers of entities and to smaller devices). From a users' perspective, the challenge is to: (i) customize algorithmic and computational tools to fully exploit the potential offered by the computing environment, and (ii) prototype application environments – experimenting with test beds – to enable intensive environments that make extensive use of computers. As the latter challenge has been addressed in [9], this paper addresses the development of a prototype bed for chemical engineering synthesis.

3. The distributed application

3.1. Problem description

The problem relates to the synthesis of complex reactor networks using stochastic optimization. Figure 3 presents the components of the methodology whereas Figure 4 illustrates a conceptual input-output structure with respect to the synthesis models, the problem data (kinetics), and the algorithmic choices involved. Each stochastic experiment involves stages of simulation, evaluation, perturbation and

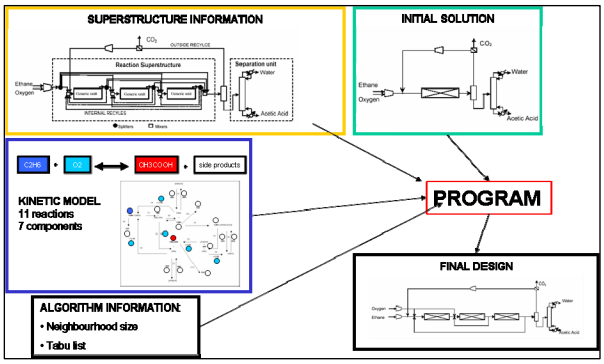

Figure 3 the components of the methodology

acceptance/rejection. The number of experiments is a function of the Markov chain and the cooling schedule. In the case of annealing, for example, the whole set of experiments has to be repeated for longer Markov chains and until

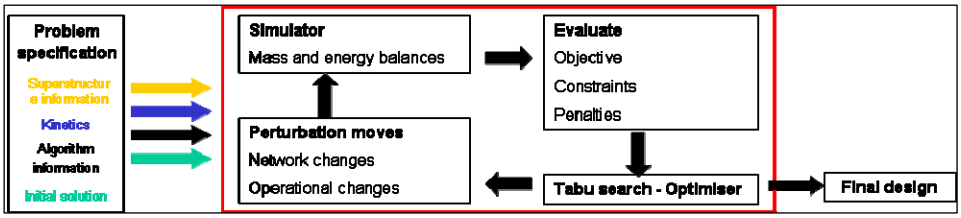

Figure 4 a conceptual input-output structure

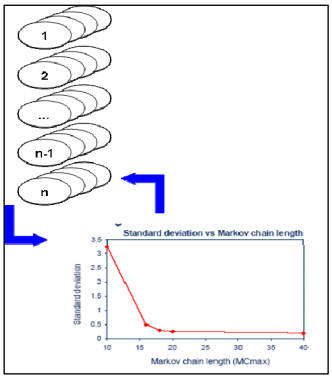

Figure 5 the experiments design

convergence criteria - that would guarantee the (statistical) quality of the solution - are met (Figure 5). Moreover, the different input streams of Figure 4 (kinetic parameters, problem constraints, feed conditions) have to be questioned and tested with respect to the sensitivity that they bring to the selected design. Hence, a distributed environment requires the development of capabilities to support: (i) the experiments involved in the iterative scheme of Figure 4; (i) the experiments involved in the iterative experiments of Figure 3; and (iii) the various parametric experiments around the input streams of Figure 5.

3.2. Test bed and Grid architecture

Although heavily dependant on the actual process and problem, the number of experiments in (i), (ii), and (iii) are in the order of 10^3, 10^2, and 10^3 .

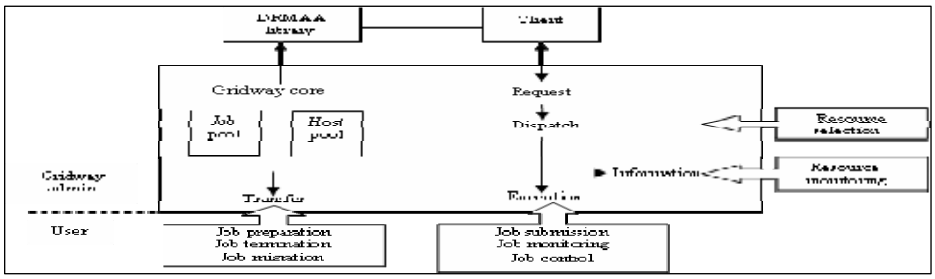

Figure 6 Global view of the project structure

respectively, leading to an overall number of experiments in the proximity of 10^8. The scheduling of these experiments in a Grid is set as a primal challenge.

Moreover, experiments in (i) may fail (to converge), whereas experiments in (ii) and (iii) need be monitored automatically (as they are currently handled manually). Additional challenges therefore relate to the management of the experiments and the systematic development of diagnostics. The design of the Grid application is outlined in Figure 6. The design features the integration of middleware for Gridway and Superscalar that are explained in the following sections.

3.3. Middleware and upperware

Gridway is a meta-scheduler of computing resources that can be used by different distributed resource management (DRM) including Condor. Its design is shown in Figure 7 [12].

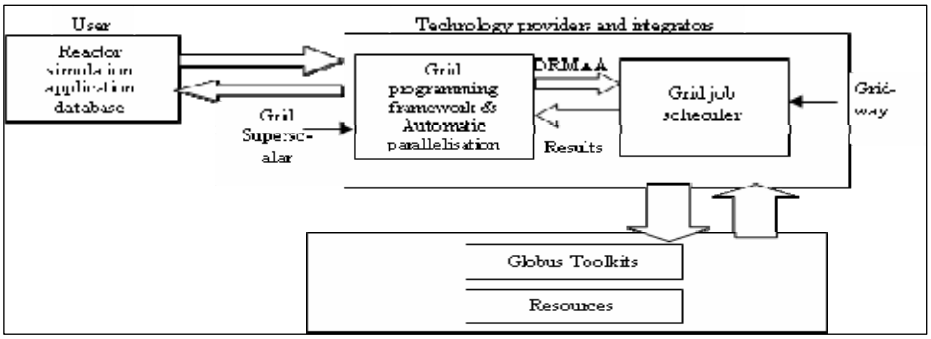

Figure 7 the infrastructure of GridWay

The Grid Superscalar is a programming paradigm that enables applications with user interfaces, and a run-time deployment center. The user interface comprises three stages: task definition, task parameters definition (identify input and output) and the sequential program.

Figure 8 explains the deployment of these stages. During run-time the system maintains capabilities of data dependence analysis, file renaming, disk sharing, task submission and check-pointing.

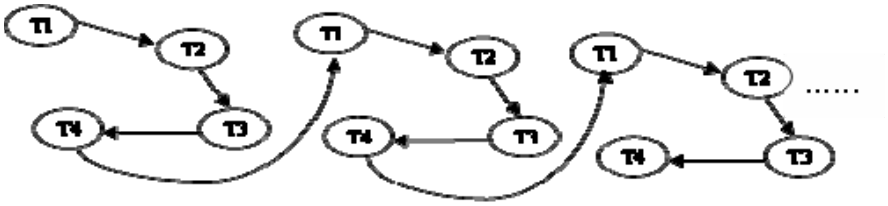

Figure 8 the task progress in Grid Superscalar

3.4. Implementation and preliminary results and discussion

Preliminary results have been obtained on distributed environments that support Condor and Globus. On the basis of 100 tests, run times have been accelerated by 50 times offering completed results in the range of 25 minutes as against 20 hours of computations required with conventional implementations. Current use of the middleware remains to focus on the acceleration of the jobs that are late or fail to converge, on the better use of the controls offered by the middleware, and on the employment of visual controls to monitor the progress of the optimization. Examples and experience is currently gathered using simple reaction schemes before the intention is to soon embark on larger applications from industry.

Acknowledgement

We acknowledge support from the EU project BeinGrid (FP6-2005-IST-5) that funds the development of the test-bed described in the paper. We remain grateful for the collaboration with the Barcelona Supercomputing Center (BSC), the Distributed System Architecture Group in UCM, and the Leibniz-Institute fur Katalyse of Germany.

References

1. Nabrzyski J., Schopf J.M. and Weglarz J., Grid Resource Management, Kluwer Academic Publishers, 2003
2. Foster I. and Ashley C. (2004) The GRID, 2nd Edition, Blueprint for a New Computing Infrastructure, 2004
3. Kokossis A. and Linke P., 'On the robust application of stochastic optimization technology of the synthesis of reaction/separation systems' Computers & Chemical Engineering, Volume 27, Issue 5(2003), Pages 733-758.
4. Yang S., Kokossis A. and Linke P., 'Toward a novel optimization approach with simultaneous knowledge acquisition for distributed computing environments', Computer-Aided Chemical Engineering, 21A(2006), pages 327-332.
5. Badia R.M., Labarta J., Sirvent R., Perez J.M., Cela J.M. and Grima R., 'Programming Grid Applications with GRID Superscalar', Journal of Grid Computing 1: 151-170, CEPBA-IBM Research Institute, UPC, Spain, 2003
6. Montero R.S., Huedo E. and Llorente I.M., 'Grid Scheduling Infrastructure with the GridWay Metashceduler', Gridworld, Distributed Systems Architecture Group, Universidad Complutense de Madrid, Spain, 2006