Parallel Distributed Computing in Modeling of the Nanomaterials Production Technologies*

V.V. Krzhizhanovskaya, V.V. Korkhov, M.A. Zatevakhin, Yu.E. Gorbachev

Simulation of physical and chemical processes occurring in the nanomaterial production technologies is a computationally challenging problem, due to the great number of coupled processes, time and length scales to be taken into account. To solve such complex problems with a good level of detail in a reasonable time, not only high-performance computational resources are needed, but also special parallel methods should be developed to efficiently utilize parallel distributed resources of computer Grids. This paper presents a Virtual Reactor problem solving environment developed for simulation of plasma chemical deposition of nanomaterials, describes an adaptive workload balancing algorithm with user-level scheduling for parallel applications on heterogeneous Grid resources, and shows some performance results.

1. Introduction

The ever-growing requirements to the nanomaterial quality standard and production efficiency lead to a dramatic need for efficient modeling and simulation approaches and methodologies. Industry component suppliers demand materials with high uniformity, low defect density and high precision in material composition and structure. To achieve this, the process parameters, installation design and operating conditions shall be fine-tuned. To simulate the material production, a great number of coupled processes shall be taken into account, occurring in a wide range of spatial and time scales. On the macro-scale, reactor geometry, pedestal configuration and electrodes position influence the properties of the produced material. On the nano-scale, particle formation, chemical kinetics and film structure and defects shall be modeled. For adequate modeling of reacting gas flow and plasma discharge, 2D and in many cases 3D problems shall be solved. For example, plasma usually induces 2D non-uniformities, while gas flow causes 3D effects. All this makes it a computationally challenging problem. To achieve the results at realistic times, high performance parallel computing is required.

Grid computing technologies opened up new opportunities to access virtually unlimited computational resources, but as any new concept it comes with a very limited support for efficient utilization of these resources. The problem is two-side: On the one hand, the efficiency of parallel applications on heterogeneous and dynamic resources of the Grid is usually very low without special adaptation of the parallel algorithms. On the other hand, the built-in resource management tools proved to be insufficient and inefficient in most cases, thus significant improvements are required in resource scheduling and resource matching mechanisms.

To help dealing with the modeling complexity, to analyze the vast amount of data, and to provide control over the simulation process, advanced problem-solving environments (PSEs) are indispensable. A user-friendly environment can guide end-users through the multiple stages of solving the problem under investigation: defining a physical engineering problem, finding appropriate simulation components, choosing optimal numerical methods, accessing data bases, initiating simulations, discovering computational resources, submitting and monitoring the jobs, analyzing and archiving the results. All these tasks can be semi-automated within generic problem solving environments or virtual laboratories for e-Science [1]. Efficient resource management –the key issue in Grid computing– can be facilitated by a user-level scheduling algorithm with resource selection mechanism built in the problem-solving environment.

A Virtual Reactor problem solving environment presented in this paper was built for simulation of industrially important technology of plasma chemical deposition used for nanomaterials production. It incorporates a number of components distributed among different organizations and requires high-

* The research was conducted with financial support from the Russian Foundation for Basic Research and the Dutch National Science Foundation, projects # 047.016.007 and 047.016.018, and with partial support from the Virtual Laboratory for e-Science project [1]

performance and high-throughput computing, as well as interactivity and advanced visualization. The application has been ported to the Grid environment and tested on a highly heterogeneous infrastructure of the RidGrid testbed [2-4].

The paper is organized as follows: Section 2 outlines the workload balancing algorithm developed and our approach to the user-level scheduling on the Grid. Section 3 describes the Virtual Reactor PSE. Section 4 gives some details on incorporating our Virtual Reactor into the Grid and presents a resulting engineering experiment scenario. Section 5 concludes the paper and draws the plans for future research.

2. Adaptive Workload Balancing and User-Level Scheduling on the Grid

In [5,6] we developed an application-level adaptive workload balancing algorithm (AWLB) for parallel applications on heterogeneous resources and validated its MPI implementation on the Grid. Intensive experimentation showed the necessity of a combined resource management on system and application levels. In [7,8] we introduced a User-Level Scheduling environment (ULS) which selects resources according to the application requirements, while the application controls and updates workload distribution. This environment acquires, maintains and controls resources on the application level. The application requirements are determined during the execution and used to allocate available Grid resources in the most optimal way. The selected resources join the user-controlled resource pool, immediately available to the user applications. With this, a virtual layer of ready-to-use resources is created, and the application requests resources not from the global Grid resource manager, but from this user-controlled layer. This helps eliminating large overheads and idle times.

A User-Level Scheduling environment contains application-specific knowledge, therefore it provides customized resource selection and control mechanisms – a feature indispensable for enabling high performance applications on the Grid. The application consists of a set of parallel tasks that process the workload scheduled by the Job Master bound to the ULS environment. The ULS operates information about the available resources and monitors the application responses. As the application comprises heterogeneous tasks executed at different times with different performance characteristics, the total application performance requirements may vary at runtime. Similarly the capacity of Grid resources may vary over time due to inherent Grid dynamics. The ULS responds to changing application or resource conditions, and selects a more suitable set of resources for the next computational time step or iteration. This is a distinctive feature of our approach, in contrast to the traditional parallel programs where resources are allocated once and fixed during the execution (unless special migration libraries are used like Dynamite [9]). To support the replacement of resources during run-time, the concept of user-level resource pool is employed. This resource pool is maintained and supplied by the ULS environment which dynamically selects the resources most suitable for the application. The suitability of resources is determined by the application requirements; for traditional parallel computing applications considered here as a test case, it depends on the processing power and network connectivity correlated with the application communication to computation ratio.

After resources have been selected and assigned to the tasks, we perform the workload balancing, using the Adaptive Workload Balancing (AWLB) algorithm [5,6]. The computation is performed as an iterative process; every iteration, the distribution of the workload is re-evaluated on a new set of resources, and the AWLB parameters are re-estimated. The AWLB algorithm provides an optimal distribution of the divisible workload between participating processors according to the computing environment characteristics and the application requirements:

- The application parameter $f_c = N_{comm}/N_{calc}$, where N_{comm} is the total amount of application communications, i.e. data to be exchanged (measured in bit) and N_{calc} is the total amount of computations to be performed (measured in Flop);
- The resource parameters $\mu_i = p_i / n_i$, where p_i is the available performance of the i^{th} processor (measured in Flop/s) and n_i is the network bandwidth to this node (measured in bit/s).

The AWLB algorithm is based on benchmarking the available resources capacity, defined as a set of individual resource parameters $\mu = \{\mu_i\}$, and experimental estimation of the application parameter f_c. The value of the application parameter f_c is determined by application analysis either at run-time (running through the space of possible f_c and finding the value f_c^* which provides minimal runtime of

the application on this set of resources) or in advance (preliminary benchmarking for computation/communication ratio). The analysis of the Virtual Reactor application is presented in [6].

The combination of μ and f_c^* determines the distribution of the workload between the processors. To calculate the amount of workload per processor, we assign a weight-factor w_i to each processor according to its processing power and network connection. It determines the final workload for a processor given by $W_i = w_i W$, where W is the total application workload. The expression for the weighting factors and the resource heterogeneity metric are described in [6].

3. The Virtual Reactor PSE

The Virtual Reactor problem solving environment is based on the previously developed models [10-14]. Initially it was designed for homogeneous parallel clusters [12,15], and recently ported to the Grid [2-6]. Virtual Reactor includes the basic components for problem description, reactor geometry design; computational mesh generation; plasma, flow and chemistry simulation; editors of chemical processes and gas properties connected to the corresponding databases; pre- and postprocessors, visualization, interaction, optimization and archiving modules [2]. These modules of the PSE have been virtualized as services and made accessible on the Grid. To hide the complexity of the underlying components, an advanced graphical user interface has been developed that seamlessly integrates the disparate distributed modules into one transparent user environment, presented to the user via a Web-interface and a Grid portal. The PSE architecture supports interaction on various levels: interaction with the workflow through the user interface, interactive visualization, control over the simulation processes and interactive job control on the middleware level.

The basic Virtual Reactor functional components and a scheme of a typical workflow cycle are sketched in Fig. 1. The arrows indicate some of the interdependencies of the components. For instance, the solvers for plasma and reactive flow simulations use as input data: description of the physical and chemical properties created by the editor components; parameters of the plasma chemical deposition processes; generated computational mesh; and computational parameters (e.g. Courant number, numerical scheme parameters, number of processors for parallel computing, etc.)

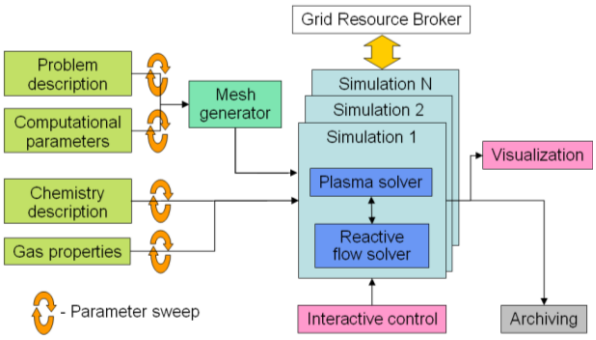

Fig. 1. Scheme of the Virtual Reactor workflow.

The solvers are submitted for execution using the User-Lever Scheduling environment that takes care of the computational resource discovery, registration, selection and allocation. The two solvers within one simulation in Fig. 1 regularly exchange the variable fields in order to track the mutual influence of plasma, chemical and flow transport processes. The output of the simulation goes through the post-processing module to optimization, visualization and results archiving components. To study the influence of various parameters on the simulated processes, a user can run a number of simulations in parallel (shown in Fig. 1 as "Simulation 1" … "Simulation N" blocks) with the assistance of Nimrod/G [16]. The sets of parameters are formed by the components shown on the left side of Fig. 1 (marked with a Parameter sweep circle).

A complete list and detailed description of the Virtual Reactor PSE components and their interdependencies can be found in [3], along with the functional decomposition and integration details. All the components are stand-alone modules that can be used separately or in various arrangements in other applications. They are platform independent and can be compiled and run on any operating system using public-domain compilers and libraries.

The development of the Virtual Reactor application workflow follows the service-oriented paradigm, which claims functional decomposition of the Virtual Reactor to a set of loosely coupled services and allows integrating them from within various workflow systems: VLAM-G, Ptolemy/Kepler,

Triana, etc. Simple workflows can be also organized with non-specialized systems and software tools, such as the CrossGrid Migrating Desktop portal utilities and the Nimrod/G with the workflow support.

4. Results

4.1. Adaptive load balancing performance and resource pool dynamics

Thorough testing of the different applications on different sets of resources showed a strong influence of the level of resource heterogeneity on the results achieved. We performed a series of targeted experiments varying the resource heterogeneity both in the processor power and the network links bandwidth. As a sample of these tests, in Fig. 2 we show the dependency of the load balancing speedup on the processing power heterogeneity metrics with the network heterogeneity fixed (see [6] for details). As we see, the speedup grows superlinearly with the heterogeneity level, thus indicating that our approach is beneficial on strongly heterogeneous resources, such as the Grid resources.

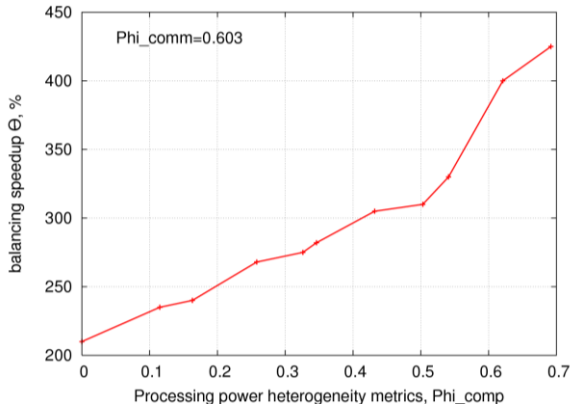

Fig. 2. Dependency of the load balancing speedup on the resource heterogeneity metrics.

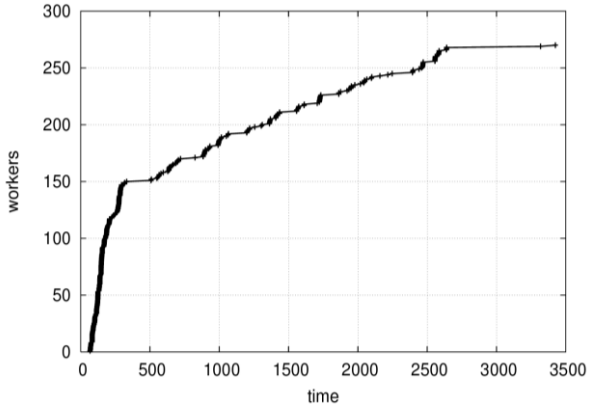

Fig. 3. Dynamics of the resource pool population.

The core feature of the ULS environment is the ability to replace resources during the execution runtime: each iteration can be run on a new resource set. Figure 3 shows how the resources are gradually added to the resource pool, depending on their availability. We can also notice that some workers are removed from the pool: the number of workers used is not growing steadily, but experiences dips from time to time.

4.2. Incorporating the Virtual Reactor into the Grid

We have incorporated our Virtual Reactor into the Grid using the CrossGrid architecture and the Migrating Desktop (MD) Grid portal, built on advanced middleware programming and Web technologies. The MD is a user-friendly interface to the Roaming Access Server backend. It provides a transparent user working environment, independent of the system version and hardware, allowing the user to access Grid resources and local resources from remote computers, via a back-end access service. It allows the user to run applications, manage data files, and store personal settings, independently of the location or the terminal type. With the use of MD we achieved secure Grid access, site discovery and registration, Grid data transfer, application initialization, interactive control and visualization.

Among other services, the MD provides plug-in support for simple workflow and visualization. It enabled distributed execution of the Virtual Reactor PSE components within a workflow model. Figure 4 shows a running experiment consisting of a number of components: The Gas Editor component is executed on one of the Grid sites, which provides a chemical database. Generated by the Editor output data files are automatically transferred to a private storage or a virtual directory space operated by the MD, according to the specifications of the workflow planned (see also Fig. 1). Next, an interactive problem description user interface is called, where one can also define a reactor geometry and computational parameters. This user interface initiates the parallel simulation solvers, using the data files

from the private storage. The interactive controller allows monitoring the execution by visualizing intermediate calculation results. This feature is enabled by periodic retrieval (refresh) of the simulation output files provided by the MD. When the simulation is complete, the result files can be retrieved from the private storage or moved to the experiments archive. All the interactive graphical components use common virtual display, so geographically distributed tasks appear on the same screen. One of the MD utilities gives the possibility to monitor the execution of the submitted tasks using special monitoring tool shown in the left lower corner of Fig. 4. A list of submitted jobs is shown, each provided with detailed status information.

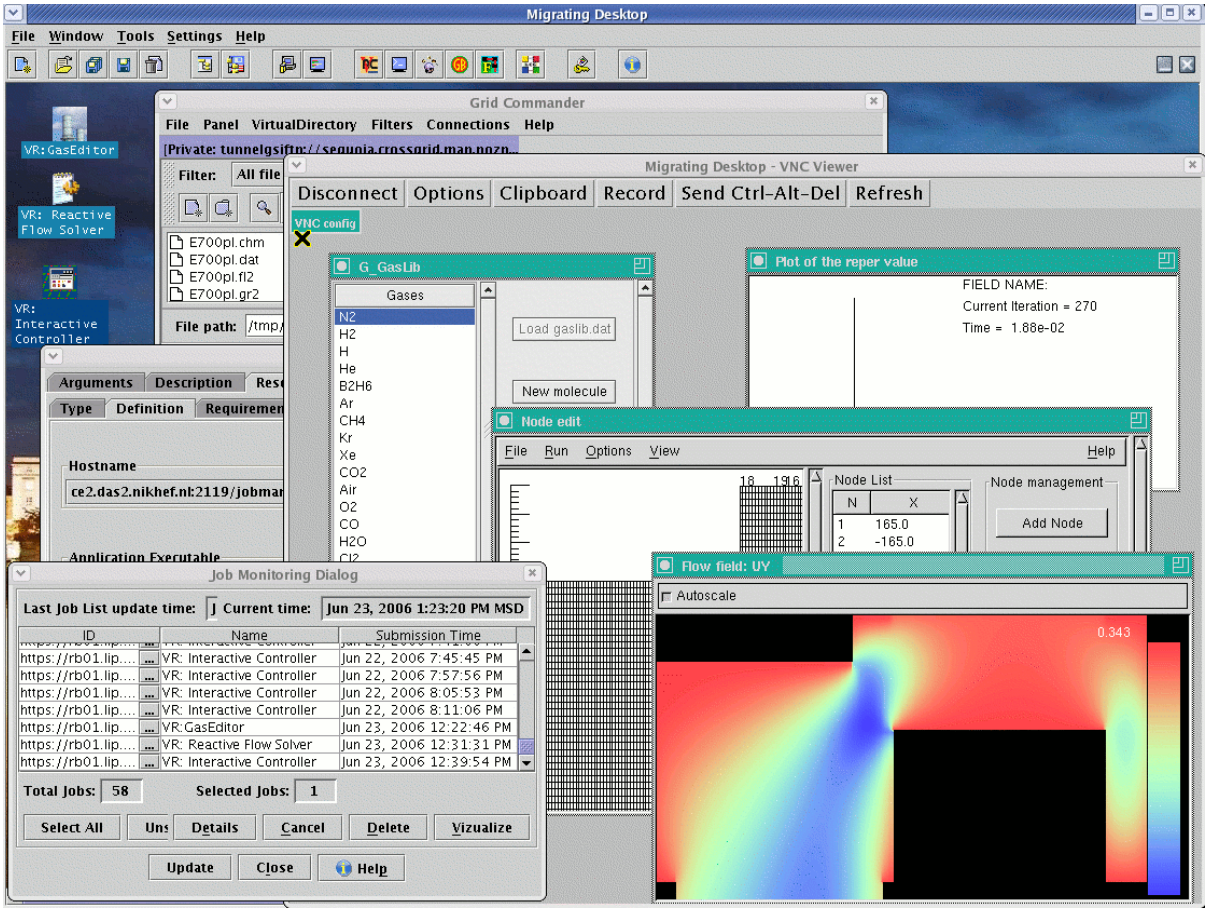

Fig. 4. A running experiment of the Virtual Reactor on the Grid. The distributed components of the application, including the interactive editors and visualizers, are accessible via the virtual folder of the Migrating Desktop portal. Separate PSE components and simulations are running on distributed Grid resources, but the graphical output is displayed on the same virtual screen of an individual user.

5. Conclusions

The workload balancing algorithm with a user-level scheduling environment proved to be beneficial for parallel distributed applications on strongly heterogeneous resources. The Virtual Reactor problem solving environment developed for simulation of plasma chemical deposition, can aid the design of industrial reactors, optimization of the nanomaterials properties, minimization of the prime costs and provides means for real-time control over PECVD reactors by running the software in parallel with the real deposition processes and tuning various parameters in interactive, manual or automatic mode.

References

1. The Virtual Laboratory for e-Science project. http://www.vl-e.nl
2. Krzhizhanovskaya V.V., Sloot P.M.A., Gorbachev Yu.E. Grid-based Simulation of Industrial Thin-Film Production // Simulation: Transactions of the Society for Modeling and Simulation International, Special Issue on Applications of Parallel and Distributed Simulation in Industry, 2005, V. 81, No. 1, pp. 77-85
3. Krzhizhanovskaya V.V. and Korkhov V.V. Problem solving Environments for Simulation and Optimization on Heterogeneous Distributed Computational Resources of the Grid // Proceedings of the Third International Conference on Parallel Computations and Control Problems (PACO '2006), Moscow, Russia, October 2-4, 2006
4. Krzhizhanovskaya V.V., Korkhov V.V., Tirado-Ramos A., Groen D.J., Shoshmina I.V., Valuev I.A., Morozov I.V., Malyshkin N.V., Gorbachev Y.E., Sloot P.M.A. Computational Engineering on the Grid: Crafting a Distributed Virtual Reactor // 2nd IEEE International Conference on e-Science and Grid Computing. Amsterdam, the Netherlands, 2006. pp.101. IEEE CS Press
5. Korkhov V.V., Krzhizhanovskaya V.V. Benchmarking and Adaptive Load Balancing of the Virtual Reactor Application on Russian-Dutch Grid // Lecture Notes in Computer Science, Vol. 3991, pp. 530-538. Springer 2006
6. Korkhov V.V., Krzhizhanovskaya V.V. and Sloot P.M.A. A Grid Based Virtual Reactor: A case study of parallel performance and adaptive load balancing // Journal of Parallel and Distributed Computing 2007, in press.
7. Krzhizhanovskaya V.V. and Korkhov V.V. Dynamic Load Balancing of Black-Box Applications with a Resource Se-lection Mechanism on Heterogeneous Resources of the Grid // Proceedings of International Conference on Parallel Computing Technologies (PaCT-2007), in LNCS, V. 4671, pp. 245–260, Springer Berlin / Heidelberg 2007
8. Korkhov V.V., Moscicki J.T., Krzhizhanovskaya V.V. User-Level Scheduling of Divisible Load Parallel Applications with Resource Selection and Adaptive Workload Balancing on the Grid. // Submitted to the IEEE Systems Journal - Special Issue on Grid Resource Management
9. Iskra K.A. et al.: The implementation of Dynamite - an environment for migrating PVM tasks // Operating Systems Review, V. 34, N 3 pp. 40-55 2000
10. Gorbachev Yu.E., Zatevakhin M.A., Kaganovich I.D. Simulation of the growth of hydrogenated amorphous silicon films from rf plasma // Technical Physics. 1996. N. 12. –P. 1247–1258.
11. Gorbachev Yu.E., Zatevakhin M.A., Krzhizhanovskaya V.V., Schweigert V.A. Special features of the growth of hydrogenated amorphous silicon in PECVD reactors // Technical Physics. 2000. – Vol. 45, N. 8, –P. 1032–1041.
12. Krzhizhanovskaya V.V., Zatevakhin M.A., Ignatiev A.A., Gorbachev Y.E., Goedheer W.J., Sloot P.M.A. A 3D Virtual Reactor for Simulation of Silicon-Based Film Production // Proceedings of the ASME/JSME PVP Conference. ASME PVP. 2004. Vol. 491-2, P. 59-68
13. Gorbachev Yu.E. Effect of Oligomers on the Growth of Amorphous Silicon Films in a PECVD Reactor // Technical Physics. 2006. –Vol. 51, N. 6, –P. 733-739.
14. Nienhus G.J., Goedheer W.J. // Modelling of a large scale reactor for plasma deposition of silicon // Plasma Sources Sci. Technol. 1999. N. 8, –P. 295-298.
15. Krzhizhanovskaya V.V., Zatevakhin M.A., Ignatiev A.A., Gorbachev Y.E., Sloot P.M.A. Distributed Simulation of Silicon-Based Film Growth // Lecture Notes in Computer Science, Vol. 2328, pp. 879-888. Springer-Verlag 2002. ISBN 3-540-43792-4.
16. Nimrod/G: Tools for Distributed Parametric Modelling. http://www.csse.monash.edu.au/~davida/nimrod/