

Towards a Virtual Shared Memory Programming Environment for Computational Grids^{*}

Håkan Mattsson¹ and Christoph Kessler²

¹ Department of Natural Science and Technology
Gotland University, Sweden

`hakan.mattsson@hgo.se`

² PELAB Programming Environments Laboratory
Department of Information and Computer Science
Linköping University, Sweden

`chrke@ida.liu.se`

Abstract

Grid computing has become an area of extensive research. Major efforts are necessary to develop suitable parallel programming environments for computational grids. The current practice in grid programming uses message passing, which unfortunately leads to low-level and unstructured code that is difficult to understand, debug, and optimize. For computational grids to become commonly accepted also as more general-purpose parallel computation platforms, there is a need for simple, well-known, easy-to-use programming environments.

This paper describes research in progress on a new parallel programming environment for computational grids, called *GridNestStep*. Building upon previous work on the parallel programming language NestStep, GridNestStep adopts a well-established parallel programming model, namely bulk-synchronous parallel (BSP) computing, which, by enforcing a clear organization of the program into parallel supersteps, provides an easily analyzable structure of GridNestStep programs. More flexibility is added by supporting nested parallelism via nesting of BSP supersteps. This allows several groups of processors to temporarily work independently of each other in parallel sub-supersteps, using only local communication and synchronization. This important property should be exploited especially in computational grids with a weak global interconnection network. Furthermore, GridNestStep supports a virtual shared memory layer on top of a computational grid system, which provides a comfortable, high-level programmer interface. The software-emulated virtual shared memory interface relaxes memory consistency in a BSP-compliant way, which means that global communication due to updating of remote copies is limited to superstep boundaries, which matches the restricted interprocessor communication possibilities in computational grids quite well.

A major challenge for the GridNestStep runtime system consists in partitioning and scheduling of supersteps, *i.e.* mapping (virtual) BSP processors to physical grid node processors such that the workload is well balanced. We suggest a simple method that is, with minor modifications, also applicable to peer-to-peer computing systems in the style of SETI@home.

GridNestStep is intended to become the computational platform of a forthcoming grid-based modeling and simulation environment for *Modelica*. Modelica is an object-oriented, equation-based language for modeling physical systems. The Modelica compiler generates a large system of ordinary and partial differential equations that is submitted to an iterative solver in order to run a simulation. Together with domain-specific parallelized solver components and with grid access and scheduling software, GridNestStep will finally constitute a complete programming environment for the GridModelica system.

^{*} Research partially supported by VINNOVA, project GridModelica.