Geophysical Research Abstracts
Vol. 14, EGU2012-8690-5, 2012
EGU General Assembly 2012

Performance of Basic Geodynamic Solvers on BG/p and on Modern Mid-sized CPU Clusters

S. Omlin (1), V. Keller (2), and Y. Podladchikov (3)

(1) University of Lausanne, Lausanne, Switzerland (samuel.omlin@unil.ch), (2) Section of Informatics, Swiss Federal Institute of Technology Lausanne, Lausanne, Switzerland (vincent.keller@epfl.ch), (3) University of Lausanne, Lausanne, Switzerland (yury.podladchikov@unil.ch)

Nowadays, most researchers have access to computer clusters. For the community developing numerical applications in geodynamics, this constitutes a very important potential: besides that current applications can be speeded up, much bigger problems can be solved. This is particularly relevant in 3D applications.

However, current practical experiments in geodynamic high-performance applications normally end with the successful demonstration of the potential by exploring the performance of the simplest example (typically the Poisson solver); more advanced practical examples are rare. For this reason, we optimize algorithms for 3D scalar problems and 3D mechanics and design concise, educational Fortran 90 templates that allow other researchers to easily plug in their own geodynamic computations: in these templates, the geodynamic computations are entirely separated from the technical programming needed for the parallelized running on a computer cluster; additionally, we develop our code with minimal syntactical differences from the MATLAB language, such that prototypes of the desired geodynamic computations can be programmed in MATLAB and then copied into the template with only minimal syntactical changes.

High-performance programming requires to a big extent taking into account the specificities of the available hardware. The hardware of the world's largest CPU clusters is very different from the one of a modern mid-sized CPU cluster. In this context, we investigate the performance of basic memory-bounded geodynamic solvers on the large-sized BlueGene/P cluster, having 13 Gb/s peak memory bandwidth, and compare it with the performance of a typical modern mid-sized CPU cluster, having 100 Gb/s peak memory bandwidth. A memory-bounded solver's performance depends only on the amount of data required for its computations and on the speed this data can be read from memory (or from the CPUs' cache).

In consequence, we speed up the solvers by optimizing memory access and CPU cache use: we avoid random memory access and multiple read of the same data by rearranging mutually independent computations. More precisely, we group operations that act on the same small parts of data as much as possible together, assuring that these small data parts fit into the CPU cache. In fact, reading from CPU cache requires nearly no time compared to reading from memory.

We also optimize the technical programming needed for a parallelized running of the solvers on a computer cluster. The parallelization of a solver requires a spatial decomposition of the computational domain; each processor solves then the problem for one sub-domain, synchronizing at every iteration the sub-domain's boundaries with the ones of its neighbours. We optimize boundary synchronization between processors by developing optimal methods based on the full range of advanced MPI-features (MPI is the standard interface for developing parallel applications on CPU clusters with distributed memory).

A geodynamic solver solves at every iteration a system of equations. This can be solved implicitly - by using a direct solver - or explicitly - by updating all variables in the system of equations based on a update rule derived from the system. We compare the performance of implicit and explicit solving for our applications.