Load-balancing for mesh-based applications on heterogeneous cluster computers

J. Fingberg [a,*], K. Nakajima [b], C. Walshaw [c]

[a] C&C Research Laboratories, NEC Europe Ltd., Rathausallee 10, D-53757 St. Augustin, Germany
[b] Research Organisation for Information Science and Technology, 1-18-16, Hamamatsucho, Minato-ku, Tokyo 105, Japan
[c] Department of Computing and Mathematical Sciences, University of Greenwich, Park Row, Greenwich, London, SE10 9LS, UK

Abstract

This paper discusses load-balancing issues when using heterogeneous cluster computers. There is a growing trend towards the use of commodity microprocessor clusters. Although today's microprocessors have reached a theoretical peak performance in the range of one GFLOPS/s, heterogeneous clusters of commodity processors are amongst the most challenging parallel systems to programme efficiently. We will outline an approach for optimising the performance of parallel mesh-based applications for heterogeneous cluster computers and present case studies with the GeoFEM code. The focus is on application cost monitoring and load balancing using the DRAMA library.

Keywords: Load-balancing; Finite element; Heterogeneous PC cluster; DRAMA; Jostle

1. Introduction

The availability of fast commodity microprocessors and high-bandwidth networks at relatively low prices enabled the development of cost efficient cluster computers with the potential for high performance. This performance can only be delivered with software tools that enable the exploitation of the full capacity of the hardware. Cluster environments naturally become heterogeneous as faster machines are added to the system or replace slower nodes. A major resulting requirement for HPC software is for heterogeneous load balancing.

With the first release of the DRAMA load-balancing library [1] into the public domain, the aim was to enable a widespread exploitation of the library as a tool to allow efficient use of general HPC platforms. Current research is focused on heterogeneous load balancing.

The GeoFEM [2] group is currently developing a parallel finite element environment for the simulation of solid earth phenomena. The final target machine is the Earth Simulator, a parallel vector processor system, developed in an inter-disciplinary project at the Earth Simulator Research and Development Centre (ESRDC) in Japan [3].

The results presented here have been obtained with a prototype version of the GeoFEM code instrumented to use the DRAMA library.

2. High-performance cluster computing

Besides network capacity, a vital key to high performance commodity cluster computing is the availability of scalable operating systems and application level software. On the single node level, an increasing processor-memory performance gap makes it difficult to achieve a substantial percentage of the theoretical peak performance. Hierarchical memory designed to allow the use of slow DRAM at the access time of fast SRAM technology, requires the optimisation of the application data layout for efficient cache usage. On the parallel level, different CPU speeds and hierarchical networks have to be considered. The complicated cluster hardware requires a special software technology that is able to compensate for its shortcomings and bridge the gap between peak and sustained performance. This is a challenge for both operating systems and tools like dynamic load balancing (DLB) libraries.

* Corresponding author. Tel.: +49 (2241) 925263; Fax: +49 (2241) 925299; E-mail: fingberg@ccrl-nece.de

Computational Fluid and Solid Mechanics
K.J. Bathe (Editor)

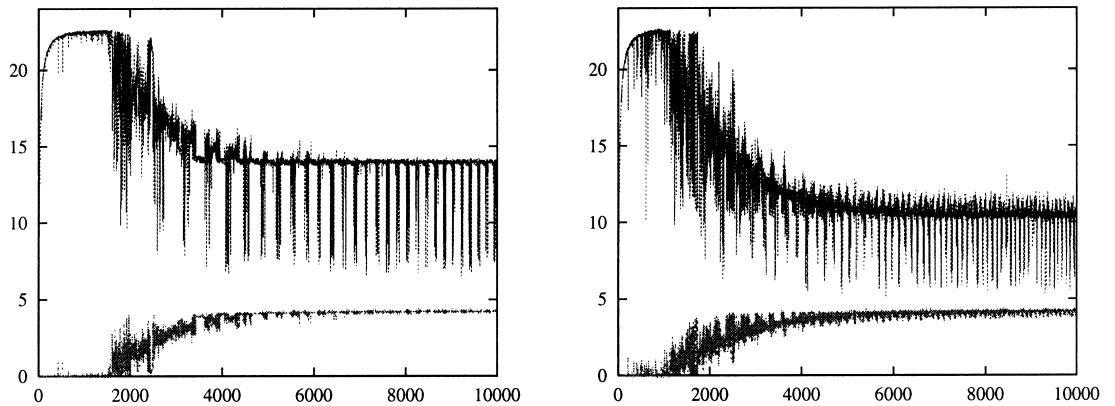

Fig. 1. Performance in MFLOPS/s (upper curve) and total number of L2-cache misses per FLOP (lower curve) scaled by a factor of 20 for single (left) and dual (right) CPU usage for a sparse matrix-vector multiplication in CSR format versus the outer loop length (corresponding to the number of internal nodes).

3. Hardware calibration

Calibration is necessary to determine the different processor speeds and the network capacity. Ideally it should be possible to measure the characteristic hardware parameters like CPU speed and network bandwidth by running short, automatic tests. However, it is not always possible to decouple hardware from software parameters. In general, the MFLOPS rate depends on the application. Network modelling is also not straightforward. It is possible to assign a cost matrix to the network with an entry for each processor pair. It is not necessarily a good representation of the physical computer network [4]. For load balancing, the purpose of this matrix is to guide both the vertex (node/element) to sub-domain and the sub-domain to process mapping. For DRAMA we consider a hierarchy of possibilities ranging from automatic procedures to full user control:

(1) CPU speeds:
   (a) processor clock rate;
   (b) timing of an application specific computational kernel (for example a sparse matrix vector product);
   (c) inverse method (define CPU speeds as MFLOP rates for equal execution time of a kernel);
   (d) user supplied CPU performance vector.
(2) Network matrix:
   (a) path length (number of links) between processor pairs;
   (b) pair-wise measurement of transfer times (Ping-Pong test);
   (c) user supplied network cost matrix.

4. Performance monitoring

An essential prerequisite to successful dynamic load balancing is an accurate cost monitoring procedure. It is of special importance to use high-resolution timers, which are thread-safe and allow the measurement of user and system times separately in order to avoid contamination from changing background loads.

As an example, which is relevant for a sparse iterative solver, we analyse the single/dual CPU performance of a sparse matrix-vector product in compressed sparse row (CSR) format. The timing and the total number of L2-cache misses are based on the PAPI library [5]. Fig. 1 illustrates the problems: the performance depends on the problem size and performance degradation and L2-cache misses are more severe in SMP mode.

5. Heterogeneous partitioning

The geometric module of the DRAMA library has been extended to allow heterogeneous partitioning with different CPU speeds by introducing artificial *bonus* loads inversely proportional to the processor speed. Full heterogeneous mapping is supported through the internal interface to Jostle [6], which is able to take both different CPU speeds and the network matrix into account. Several heterogeneous mapping configurations are tested in [4]. The power of the process to compute such a mapping appears to stem from the *global* properties of the multilevel algorithm. Edges which cross expensive links are penalised heavily within the cost function and so vertices at either end of such an edge tend to migrate to more *adjacent* processors and create a sort of buffer zone. However, because this occurs high up in the multilevel process, where each vertex represents many vertices in the original graph, the buffer zone which may start off only one vertex wide, can actually represent reasonably broad regions in the mesh. In this way, the partition is given a good global quality on the coarse graphs, which is refined on the finer graphs.

6. Case studies

Tests have been performed on a PC-cluster at NEC consisting of 16 dual processor SMP machines, 13 have 200 MHz Pentium-Pro processors (*slow* nodes), and three have 600 MHz Pentium-III processors (*fast* nodes). A Myrinet network composed of four switches each connecting four machines provides a maximum communication bandwidth of 1.28 Gb/s.

The application is a parallel adaptive CFD code [7] that has been developed as part of the GeoFEM project. A partitioner tool using the DRAMA library has been integrated to provide dynamic load balancing and data migration.

We study a simple case to test the hypothesis that heterogeneous load balancing can be successful without complicated calibration/tuning using only the clock rate (assuming one FLOP per cycle) as computational speed. The load balancing strategy is to balance the number of nodes with using graph (Jostle) or geometric (RCB) partitioning.

The analysis is over a window of 3000 iterations with three adaptation steps separated by 1000 time steps after an equilibration of 30,250 cycles.

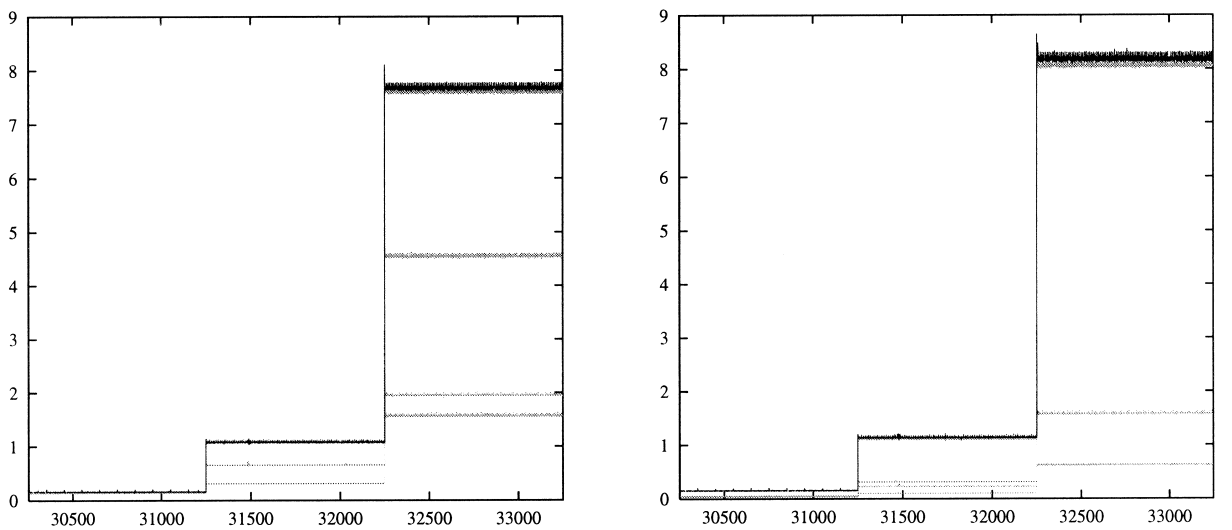

Fig. 2. Total time in seconds per solver iteration (upper line) including communication and the computational costs (lower four lines) as a function of the iteration count without load balancing. The left side shows the homogeneous situation (case 1); the right side shows the heterogeneous case (2) all in single CPU per machine mode.

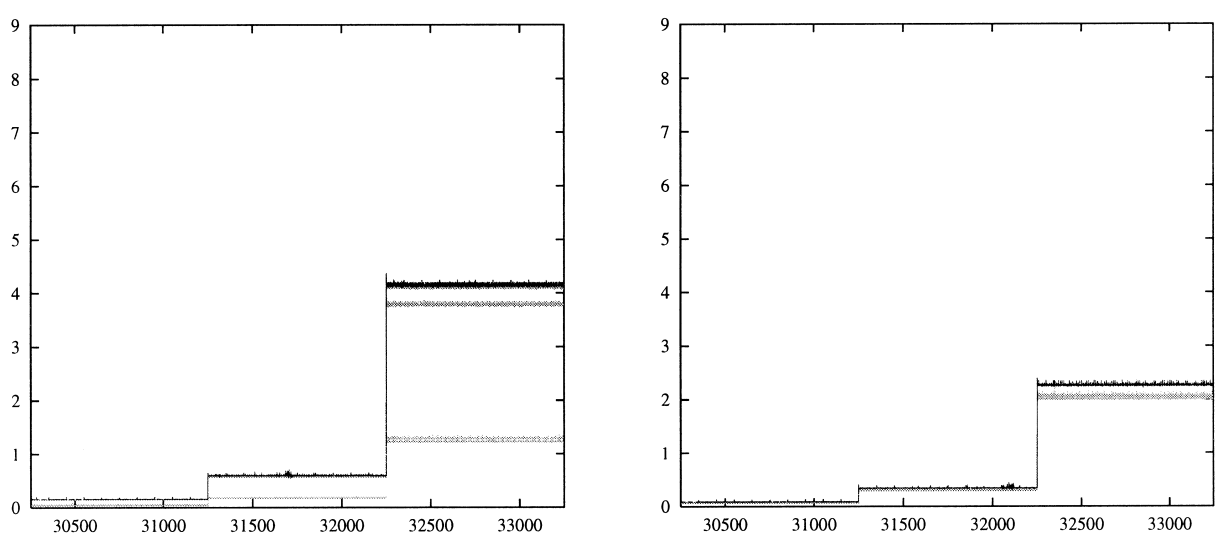

Fig. 3. Homogeneous (left, 3) and heterogeneous (right, 4) partitioning for 2 slow and 2 fast processors. The solver time per iteration (upper line) and the purely computational costs (lower 4 lines) are shown for each processor as a function of the iteration count.

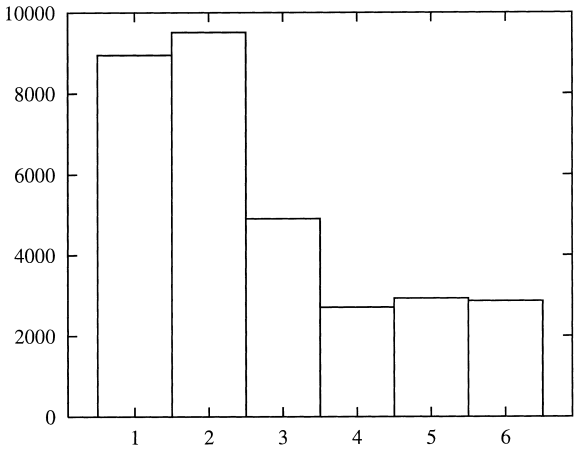

Fig. 4. Total solver time in seconds for the six test cases.

The following six test cases, all performed on four processors, are compared:

(1) 4 slow nodes without DLB;
(2) 2 slow + 2 fast nodes without DLB;
(3) 2 slow + 2 fast nodes with homogeneous DLB;
(4) 2 slow + 2 fast nodes with heterogeneous DLB (homogeneous network);
(5) 2 slow + 2 fast nodes with heterogeneous DLB (heterogeneous network, SMP mode);
(6) 2 slow + 2 fast nodes with heterogeneous RCB.

The comparison in Fig. 2 shows that adding fast machines to the cluster does not improve performance without load balancing. The slowest processor determines the speed because of explicit (time step) and implicit (boundary exchange) synchronisation between processes. Fig. 3 shows the advantage of heterogeneous over homogeneous partitioning and Fig. 4 illustrates that the heterogeneous methods (4, 5, 6) give almost equally good solver performance.

7. Concluding remarks

The results presented in Section 6 demonstrate that heterogeneous dynamic load balancing using the DRAMA library gives a significant increase in efficiency on a typical PC-cluster. Initial tests with a small number of processors indicate that a simple strategy without hardware parameter tuning is sufficient. Of course, the promising initial results have to be verified on a larger number of processors, where the network capability becomes more important.

Finally, it should be pointed out that the DRAMA library is further developed and maintained. It is hoped that feedback from present and future DRAMA users will help to steer these and other future developments.

References

[1] The GeoFEM Web-site: http://geofem.tokyo.rist.or.jp
[2] The DRAMA Web-site: http://www.ccrl-nece.de/DRAMA
[3] The ESRDC homepage: http://www.gaia.jaeri.go.jp
[4] Walshaw C, Cross M. Multilevel Mesh Partitioning for Heterogeneous Communication Networks, to appear in Future Generation Computer Systems (originally published as Univ. Greenwich Tech. Rep. 00/IM/57), 2000.
[5] The Performance Data Standard and API Web-site: http://icl.cs.utk.edu/projects/papi
[6] The Jostle homepage: http://www.gre.ac.uk/Jostle
[7] Parthasarathy V, Kallinderis Y, Nakajima K. Hybrid adaptation method and directional viscous multigrid with prismatic-tetrahedral meshes, AIAA Paper 95-0670, Reno, NV, January 1995.