CLUE – A Performance Evaluation Tool for Clusters of SMPs

Helmut Hlavacs[1]

Dieter F. Kvasnicka[2]

Christoph W. Ueberhuber[3]

[1]Institute of Applied Computer Science and Information Systems,
University of Vienna
hlavacs@ani.univie.ac.at
[2] Institute for Physical and Theoretical Chemistry,
Vienna University of Technology
dieter@theochem.tuwien.ac.at
[3]Institute for Applied and Numerical Mathematics,
Vienna University of Technology
christof@uranus.tuwien.ac.at

Computer clusters built of of-the-shelf PC components are becoming more and more popular, as they provide a low-cost alternative to traditional high-performance computers.

Unfortunately, as the performance of parallel software depends crucially on the underlying hardware, it is very difficult to design a balanced system, delivering the maximum achievable performance when running the parallel software of the cluster‘s buyer.

When deciding which components to include into the hardware, designers have to choose from a huge variety of available hardware components, including CPU, cache, memory and network cards. It is generally not known in advance, which impact the choice of one certain product will have on the performance of the parallel programs. Some choices, though seemingly attractive, may well have a devastating impact on the performance of certain parallel programs. If the cluster has been bought to mainly run some special software only, which is often the case at institutions being involved in computational science, the whole cluster may be annoyingly slow and considered worthless.

The aim of CLUE, the newly developed tool, is to simulate the impact of certain hardware decisions on the performance of parallel programs being run on such systems. CLUE is based on MISS-PVM ([1], [2]), a simulation layer between PVM based parallel programs and the PVM ([3]) library. PVM programs do not

have to be rewritten, but must include a different `pvm3.h` header file, redirecting all PVM calls to the virtual layer. MISS-PVM compiled programs may run on one machine only or may be distributed to several machines. Due to the simulation of a virtual time, independent PVM processes still communicate by exchanging messages. The order of these messages is preserved and messages arrive at their destination in correct order, and at the correct virtual time point.

Though MISS-PVM is based on PVM only, it is possible to simulate also the behavior of MPI based programs, if they are available in both PVM and MPI form. Examples will be given at the conference.

By simulating different scenarios, each representing different hardware configurations, the cluster designer is able to evaluate the impact of these configurations on the performance of his parallel software.

Up to now, two Beowulf clusters have been examined. One cluster, being maintained due to a cooperation between the Institute for Applied and Numerical Mathematics and the Institute for Physical and Theoretical Chemistry, Vienna University of Technology, contains two Pentium II 350 MHz processors per node, having 5 nodes in total. The nodes are interconnected by a standard 100 Mbit/s LAN. The second cluster being situated at RWTH Aachen contains 16 nodes, each holding two Pentium II 400 MHz. The interconnection is accomplished by a 640 Mbit/s SCI network having a 2-dimensional torus topology.

On both clusters, parallel versions of the Cholesky factorization, LU factorization and matrix-matrix multiplication have been run. The results are encouraging and indicate good performance estimation for a wide variety of algorithms.

References

[1] Hlavacs H., Ueberhuber C.W., *Simulating Load Balancing on Heterogeneous Workstation Clusters*, in Parallel Computation, Proceedings of the 4th International ACPC Conference, Springer-Verlag Wien, 1999, pp. 533-540.

[2] Kvasnicka D., Ueberhuber C.W., *Developing Architecture Adaptive Algorithms using Simulation with MISS-PVM for Performance Prediction*, Conference Proceedings of the 1997 ACM/SIGARCH International Conference on Supercomputing, Vienna, Austria, July 7-11, 1997, pp. 333-339.

[3] Sunderam V.S., Geist G.A., Dongarra J., Manchek R., *The PVM concurrent computing system: Evolution, experiences and trends*, Parallel Computing 20-4 (1994), pp. 531-545.