Effective Performance Problem Detection of MPI Programs on MPP Systems: From the Global View to the Detail

Rolf Rabenseifner[1*], Peter Gottschling[2], Wolfgang E. Nagel[2], and Stephan Seidl[2]

[1] High-Performance Computing-Center Stuttgart (HLRS), University of Stuttgart, D-10550 Stuttgart, Germany, www.hlrs.de/people/rabenseifner, rabenseifner@hlrs.de

[2] Center for High Performance Computing (ZHR), Dresden University of Technology, D-01062 Dresden, Germany, www.tu-dresden.de/zhr, [gottschling|nagel|seidl]@zhr.tu-dresden.de

Counter-Based Profiling: This paper presents an automatic counter instrumentation and profiling module added to the MPI library on Cray T3E systems. A detailed summary of the hardware performance counters and the MPI calls of any MPI production program is gathered during execution and written on a special syslog file. The user can get the same information in a different file. Statistical summaries are computed weekly and monthly. The paper describes experiences with this library on the Cray T3E systems at HLRS Stuttgart and TU Dresden.

Scalable User-Interface: The paper focuses on the scalability aspects of the new interface: How to get the right amount of performance data to the right person in time, and how to draw conclusions for the further optimization process, e.g. with the trace-based profiling tool VAMPIR. The user interface is designed along the basic principle that the cost-benefit ratio is scalable for the user. This means that one is able to achieve the most important information about an MPI application with an effort that is roughly zero. If more information is wanted, one has several levels of support: At the first level, the user can get the MPI percentage, the total instruction rate, and the floating point instruction rate by viewing an e-mail. At the second level, he can read the detailed information in the e-mail. This e-mail is sent each weekend and summarizes all MPI jobs of the user in the previous week. At the next level, the user has to act by himself. By setting a special environment variable, the user can get the counter information at the end of each MPI application run. With another variable he can choose different hardware events, e.g. the integer instruction rate. By calling MPI_Pcontrol, intermediate printings of the counters can be issued. These levels provide the information base to decide whether other tools should be used. Therefore, the next level is not part of the automatic MPI profiling, but part of the scalability strategy: Using trace based tools, e.g. VAMPIR, the user gets detailed insight into the computation and communication structure of the application.

Feedback was given by 28 users in a survey done by phone. 93 % of them are using the profiling and 75 % of those interviewed believe that the profiling can help in the future to improve their applications. The paper presents detailed results.

Comparing the methods: The major advantages of the automatic counter-based profiling compared with the trace-based method are the usability for production jobs and the easy interface of the first usage levels.

Optimizing the User's Applications: If the user discovers in the weekly e-mail that the application does not run efficiently, normally the significant information in the per-job output corresponds to a single communication pattern around the computational kernel so that this communication pattern can be optimized. In the paper, one example for a full cycle on all levels is presented.

---

* Part of the work was done while the author was a visiting research associate at ZHR, Dresden University of Technology, Jan.-April 1999.