Monitoring Cache Behavior on Parallel SMP Architectures and Related Programming Tools

T. Brandes, H. Schwamborn, M. Gerndt, J. Jeitner, K. Kereku, W. Karl, M. Schulz,
J. Tao, H. Brunst, W.E. Nagel, R. Neumann, R. Müller-Pfefferkorn, B. Trenkler, H.-C. Hoppe

Abstract

The gap between CPU performance (increasing approximately by a factor of 1.5 per year) and memory performance (factor of 1.07 per year) has increased dramatically and might continue to increase during the next years. Caches placed between memory and processor provide the possibility to access data much faster by duplicating parts of main memory in smaller and faster memory. But the advantages of the cache are only given if temporal and spatial data locality is exploited in the user program.

Re-using data in the cache provides temporal locality and can speed up a program by a factor of 6 to 10 on many computers. Straight-forward written programs not taking the cache hierarchy into account achieve only a small fraction of the theoretical peak performance. Tuning the program for better cache utilization has become an expensive part of the software development cycle. Furthermore, the modular and extendible design of software conflicts directly with the locality needed for the exploitation of the cache. Identification and understanding of bottlenecks in a program due to cache problems is one of the most critical issues. General information about cache misses is not very useful as they give the user only the information that something goes wrong but not where and why.

The project EP-CACHE (funded by the German Federal Ministry of Education and Research, BMBF) is intended to overcome this problem. By exploiting hardware monitors and related monitor control techniques the user can gain more useful information about the cache behavior in his program. Related tools for monitor controlling, performance visualization and optimization provide new and advanced possibilities for the identification of memory-cache problems and their elimination. The tools address the analysis and the optimization of programs for cache architectures, especially for SMP clusters. The rest of the paper is organized as follows. Section 2 describes our approach of hardware supported cache monitoring. The performance analysis outlined in Section 3 uses the VAMPIR tool for visualization of the monitored data. In Section 4 we outline our tools for optimizing the source program for better cache usage. We present first evaluation results in Section 5 and we give a summary in Section 6.