A Fault Tolerant Gaussian Elimination Solver for the Cell Broadband Engine

James Geraci
jrgeraci@alum.mit.edu

Abstract

This talk presents a Gaussian elimination solver for the Cell Broadband Engine (CBE) that is able to handle the addition or removal of cores during computation without restarting the computation. This ability to dynamically change the number of cores participating in the solver can be used to implement fault tolerance, provide the multicore system a cooperative/preemptive multitasking environment, and help manage the thermal properties of the chip. These applications are also investigated.

Background

In [1], Bosilca et al. present an clever algorithm based fault tolerant scheme for multiprocessor systems that effectively deals with processor loss (aka erasure) failures. Their scheme is diskless and relies on checksums. It utilizes $(2p\text{-}1)$ checksum processors for every p^2 processors being used for computation. As such, this method's efficiency grows with the numbers of processors.

However, for smaller numbers of processors such as the 8, 16, 32, 64 or 80 cores found on today's multicore processors, at best, 17% of the cores must be allocated to checksum usage. This can be prohibitive for embedded systems where space and power constraints are tight.

Fault Tolerance for Embedded Multicore

The Gaussian elimination solver presented here utilizes the memory bandwidth of the CBE to continuously backup its calculations in main memory.

To minimize the overhead of the fault tolerance, the algorithm utilizes the natural synchronizations needed by Gaussian elimination to have all the cores check in with the PPE. If a core fails to check in within a predetermined time interval, it is considered to have failed.

If a core fails, that core's compute task is redistributed among the remaining cores and the solver finishes without having to restart the computation. This process is seen in Figure 1 where a sequence of matrix equations are solved by the solver. The first 5 solves utilize 6 SPEs. However, during the 6th solve, the PPE stops 4 of the SPEs. These 4 SPEs do not check in, so their remaining workload is redistributed between the remaining two cores. They finish the 6th solve and then proceed with the remaining solves.

Beyond Fault Tolerance

Because this method does not care why a core doesn't check in, it can be used for a number of different applications.

First, this technique can be used by a scheduler to reassign a few cores to a different compute task for a while. Context switching has traditionally been thought of as removing a task from a processor, running a different task for a period of time, and then returning the first task to the processor. The huge cache size and number of cores on multicore processors makes this approach inefficient. Therefore, instead of having the scheduler remove a task from all the cores, it could simply preempt some of the cores and reassign them to a different task. The algorithm cooperates in that it knows how to handle the removal or addition of some of its cores and thereby doesn't crash when its core count is changed.

Another important benefit of being able to adjust the utilized number of cores in real time can be found in thermal management. In multicore processors, active cores tend to heat up by various amounts even for the same task and heat differently for different types of tasks [3]. On chip thermal sensors can be used to detect hot cores and dynamically reassign the task to cooler cores. The scheduler can learn the thermal behavior of the cores in response to various types of operations and in the future avoid assigning certain types of tasks to the coolest cores for those tasks. It can also increase or decrease the number of cores used in response to processor thermals. Minimizing use of hot cores reduces the dollar cost of computation.

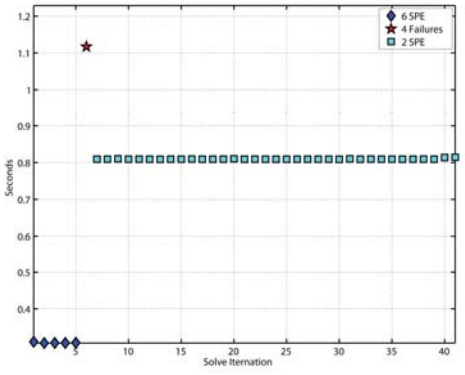

Figure 1: During the 6th solve, the solver properly handles the removal of four SPEs from its list of cores by the PPE.

References

[1] G. Bosilca and R. Delmas and J. Dongarra and J. Langou, "Algorithmic Based Fault Tolerance Applied to High Performance Computing," *LAPACK Working Note #205*, June 2008.

[2] J. Geraci, *A comparison of parallel Gaussian elimination solvers for the computation of electrochemical battery models on the cell processor*, Thesis Ph.D., MIT, 2008.

[3] E. Kursun and C. Chen-Yong, "Thermal variation characterization and thermal management of multicore architectures," *Micro, IEEE*, Vol 29, No. 1, Jan 2009.