Sparse Matrix-Vector Multiplication and Conjugate Gradient Algorithm on Hybrid Computing Platforms

David DuBois, Andrew DuBois, Carolyn Connor Davenport, Stephen Poole, HPC-5

Sparse matrices, derived from systems of partial differential equations (PDEs), occur in physics, mechanical engineering and other fields where a physical phenomenon needs to be mathematically described. These PDEs are used to understand phenomena such as fluid flow, the growth of crystals, gravitation, diffusion, and the behavior of electromagnetic fields.

The solution to a nonsingular linear system $\mathbf{A}x=b$ lies in a Krylov space whose dimension is the degree of the minimal polynomial of **A** (where **A** is a matrix, x & b are vectors). If this minimal polynomial of **A** has a low degree, a Krylov method has the potential of rapid convergence [1]. When solving a system of linear equations $\mathbf{A}x=b$, if the coefficient matrix **A** is large and sparse, the time required to solve the system by direct methods is too high and requires too much storage.

Krylov methods, or solvers like Conjugate Gradient (CG) [2], are particularly well suited for use on large-scale scientific simulation codes that are defined by sparse linear systems. These codes call solvers such as CG, which in turn repeatedly perform sparse matrix-vector multiplication (SMVM) operations to converge on a result for each time step [3]. To facilitate convergence, CG uses the gradient descent method to minimize a residual vector (Fig. 1) [4].

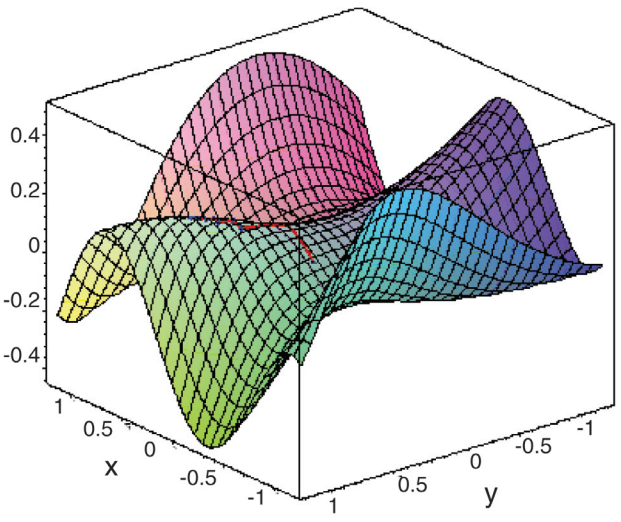

Fig. 1.
The "gradient" descent-method applied to a function. This image shows the surface of the function (3-D interpretation) [4].

Double-precision floating point SMVM is the time dominant computational kernel used in iterative solvers like CG. It is imperative that the SMVM operations be computed efficiently, yet the poor data locality exhibited by sparse matrices along with the high memory bandwidth requirements of SMVM result in poor cache utilization in general-purpose processors. Field programmable gate arrays (FPGAs) and the heterogeneous multicore architecture of the Cell processor offer possible alternatives. The Cell architecture is a hybrid computing innovation used in Roadrunner, the IBM supercomputer now being installed at Los Alamos National Laboratory for the Advanced Simulation and Computing (ASC) program.

We have developed a FPGA-based implementation of a double-precision, non-preconditioned, conjugate gradient solver for three-dimensional finite-element or finite-difference methods [5]. Our work uses the SRC Computers, Inc., MAPStation hardware platform (Fig. 2) and the "Carte" software programming environment.
We have demonstrated that an FPGA-based system can perform on par with today's processors while running over 30 times slower (i.e., 100 MHz vs 3.4 GHz), which makes the FPGA-based design much more power efficient [6]. This is possible because an FPGA-based system can be designed to more optimally match the computational units to available memory bandwidth, providing a more balanced system. It still suffers from the basic physical constraints of limited I/O and limited memory bandwidth

for this and other memory bandwidth-intensive classes of problems. To efficiently use the peak computational capability of FPGA, hybrid (e.g., Cell), or CPU-based systems for this class of problems requires tremendous amounts of memory bandwidth.

Conclusion

The FPGA results we have presented are deterministic (i.e., not variable) and will scale directly with any improvements in the system user logic frequency, memory bandwidth, and/or memory depth. We are working to port a preconditioned CG algorithm onto the Cell processor. Our goal is to exploit the improved memory bandwidth available on the Cell processor, giving us increased performance over both the traditional, cache-based implementations and our FPGA result.

For more information contact Andrew DuBois at ajd@lanl.gov.

[1] I. Ipsen and C. F. Meyer, "The Idea Behind Krylov Methods," *Am. Math. Mon.* **105**, 10, 889 (1998).
[2] J. Shewchuk, "An Introduction to the Conjugate Gradient Method Without the Agonizing Pain," School of Computer Science, Carnegie Mellon University (1994).
[3] J. Mellor-Crummey and J. Garvin, *Int J High Perform C*, **18**, 2, 225 (Summer 2004).
[4] http://en.wikipedia.org/wiki/Gradient_descent (definition)
[5] D. DuBois, et al., "An Implementation of the Conjugate Gradient Algorithm on FPGAs," to be published.
[6] D. DuBois, et al., "Sparse Matrix-Vector Multiplication on a Reconfigurable Supercomputer," submitted for publication to IEEE Transactions on Parallel and Distributed Systems, LA-UR-06-5312.

Funding Acknowledgements

This project was supported through the LANL ASC Weapons-Supported Research project.

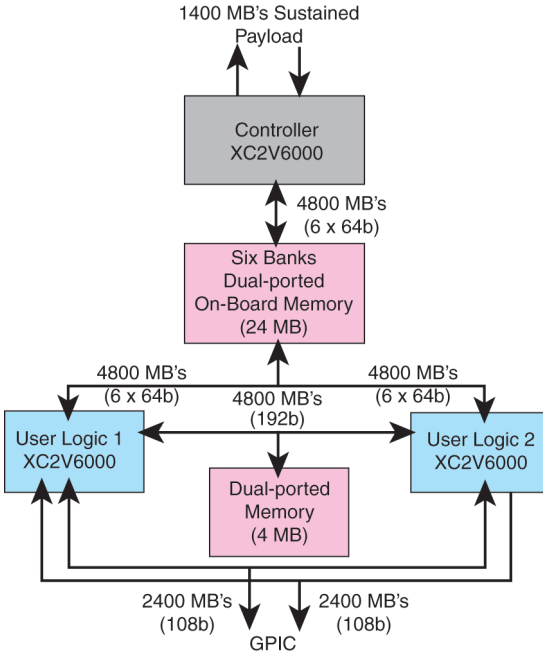

Fig. 2.
SRC Computers, Inc., MAPStation's MAP Processor block diagram [6].