Evaluation of Leading Scalar and Vector Architectures for Scientific Computations

Group Representative
Horst D Simon National Energy Research Scientific Computing Center

Authors
Horst D Simon *1 ・ Leonid Oliker *2 ・ Andrew Canning *2 ・ Jonathan Carter *2
Stephane Ethier *3 ・ John Shalf *2

*1 National Energy Research Scientific Computing Center
*2 Lawrence Berkeley National Laboratory
*3 Princeton Plasma Physics Laboratory

The growing gap between sustained and peak performance for scientific applications is a well-known problem in high performance computing. The recent development of parallel vector systems offers the potential to reduce this gap for many computational science codes and deliver a substantial increase in computing capabilities. This project examines the performance of the cacheless vector Earth Simulator (ES) and compares it to superscalar cache-based IBM Power3 system. Results demonstrate that the ES is significantly faster than the Power3 architecture, highlighting the tremendous potential advantage of the ES for numerical simulation. However, vectorization of a particle-in-cell application (GTC) greatly increased the memory footprint preventing loop-level parallelism and limiting scalability potential.

Keywords: Performance evaluation, vectorization, scientific computing

1. Overview

Applications scientists have observed a frustrating trend of stagnating application performance despite dramatic increases in claimed peak performance of high-performance computing (HPC) systems. This effect has been widely attributed to systems composed of commodity components, whose architectural designs are unbalanced and inefficient for large-scale scientific computations. The recent development of parallel vector systems offers the potential to bridge this performance gap for a significant number of scientific codes, and to increase computational power substantially. In order to quantify what a vector capability entails for scientific communities that rely on modeling and simulation, it is critical to evaluate it in the context demanding computational algorithms.

This work compares performance of the cacheless vector Earth Simulator (ES) versus the superscalar cache-based IBM Power 3 located at NERSC [8]. Performance results are presented from several key scientific computing domains including plasma fusion, astrophysics, material science and magnetic fusion.

2. LBMHD: Plasma Fusion

Lattice Boltzmann methods provide a mesoscopic description of the transport properties of physical systems using a linearized Boltzmann equation. They offer an efficient way to model turbulence and collisions in a fluid. The LBMHD application [1, 7] performs a 2D simulation of high-temperature conducting plasma using a mixed octagonal and square.

The LBMHD simulation has three computationally demanding components: computation of the mean macroscopic variables (integration); relaxation of the macroscopic variables after colliding (collision); and propagation of the macroscopic variables to neighboring grid points (stream). The first two steps are floating-point intensive, the third consists of data movement only. The problem is ideally suited for vector architectures. The first two steps are completely vectorizable, since the computation for each grid point is purely local. The third step consists of a set of strided copy operations. In addition, distributing the grid via a 2D decomposition easily parallelizes the method. The first two steps require no communication, while the third has a regular, static communication pattern in which the boundary values of the macroscopic variables are exchanged.

2.1. LBMHD Performance Results

Parallel LBMHD performance using a grid of 8192 by 8192 is presented in Table 1. Notice that the ES performance

Table 1 LBMHD per processor performance on 8192×8192 grid.

P	Power 3		ES	
	MF/s	%peak	MF/s	%peak
64	104	7 %	4640	58 %
256	115	8 %	4250	53 %
1024	108	7 %	3300	41 %

is more than 30X faster than the Power 3, achieving 3.4 Tflop/s – the highest performance on any evaluated system to date.

In terms of scalability, the Power 3, performance is fairly constant going from 64 to 1024 processors. This is due to two competing effects: as the data is further subdivided a larger fraction fits into the Power 3 cache, resulting in a performance increase; however performance degrades with the increasing ratio of communication to computation. The ES shows high sustained performance since the computational component remains constant as the average vector length (AVL) remains close to the maximum of 256. However scalability suffers some drop-off at higher processor counts due to the effect of increasing communication overhead (as in the Power 3).

3. CACTUS: Astrophysics

One of the most challenging problems in astrophysics is the numerical solution of Einstein's equations following from the Theory of General Relativity (GR): a set of coupled nonlinear hyperbolic and elliptic equations containing thousands of terms when fully expanded. The Albert Einstein Institute in Potsdam, Germany, developed the Cactus [2, 9] to evolve these equations stably in 3D on supercomputers to simulate astrophysical phenomena with high gravitational fluxes, such as the collision of two black holes and the gravitational waves that radiate from that event.

The core of the Cactus solver uses the ADM formalism, also known as the 3 + 1 form. In GR, space and time form a 4D space (three spatial and one temporal dimension) that can be sliced along any dimension. For the purpose of solving Einstein's equations, the ADM solver decomposes the solution into 3D spatial hypersurfaces that represent different slices of space along the time dimension. In this formalism, the equations are written as four constraint equations and 12 evolution equations. The evolution equations can be solved using a number of different numerical methods, including staggered leapfrog, McCormack, Lax-Wendroff, and iterative Crank-Nicholson schemes. A "lapse" function describes the time slicing between hypersurfaces for each step in the evolution. A "shift metric" is used to move the coordinate system at each step to avoid being drawn into a singularity. The four constraint equations are used to select different lapse functions and the related shift vectors.

Table 2 Cactus per processor performance. Power 3 achieves 85 Mflops/s (6% of peak) for single processor performance at $80 \times 80 \times 80$.

P	ES		ES	
	$80 \times 80 \times 80$		$250 \times 64 \times 64$	
	MF/s	%peak	MF/s	%peak
16	1466	18 %	2828	35 %
256	1355	17 %	2668	33 %
512	1346	17 %	2650	33 %
1024	1342	17 %	2657	33 %

3.1. CACTUS Performance Results

The full-fledged production version of Cactus was run on the ES system with results for two problem sizes show in Table 2. Performance on 1024 processors achieved an impressive 2.7 Tflops/s. This represents the highest per processor performance (by far) ever achieved by Cactus on any system that has been evaluated to date. The Power 3 on the other hand only achieves 85 Mflop/s (6% of peak) single processor performance, demonstrating the tremendous advantage of the ES for this class of applications. The problem size was scaled with the number of processors to keep the computational load the same (weak scaling). Cactus problems are typically scaled in this manner because their science requires the highest-possible resolutions.

For the ES, the AVL was entirely dependent on the size of the x-dimension of the local computational domain. Consequently, larger problem sizes ($250 \times 64 \times 64$ cell problem domain) executed with far higher efficiency because they supported a larger AVL. The oddly shaped domains did not appear to have a significant effect on scaling efficiency. Additional performance gains could be realized if the compiler was able to fuse the X and Y loop nests to form larger effective vector lengths. Otherwise, it may be advantageous to recopy data into work-vectors (as is done for GTC in Section 5).

Additional performance improvements can be obtained by vectorizing the boundary condition calculations. On scalar processors, the boundary conditions account for < 5% of the overall execution time, so they are not typically targets for optimization. However, on the ES the unvectorized boundary conditions now occupy over 20% of the execution time in the main loop. Recently, the Cactus team has devised a method to reorganize the boundary condition calculations so that they will vectorize efficiently. This will likely improve execution efficiency to 50% or more for the larger problem sizes.

4. PARATEC: Material Science

Paratec [10, 3] performs first-principles quantum mechanical total energy calculations using pseudopotentials and a plane wave basis set. The approach is based on Density

Table 3 Paratec per processor performance for a 432 Si-atom bulk system for 5CG steps

P	Power 3		ES	
	MF/s	%peak	MF/s	%peak
32	950	63 %	4763	60 %
64	848	57 %	4764	59 %
128	739	49 %	4742	59 %
256	592	39 %	4169	52 %
512			3392	42 %
1024			2077	26 %

Functional Theory (DFT) that has become the standard technique in materials science to calculate accurately the structural and electronic properties of new materials with a full quantum mechanical treatment of the electrons. Codes performing DFT calculations are among the largest consumers of computer cycles in centers around the world, with the plane-wave pseudopotential approach being the most commonly used. Both experimental and theory groups use these types of codes to study properties such as strength, cohesion, growth, magnetic, optical, and transport for materials like nanostructures, complex surfaces, and doped semiconductors.

Paratec uses an all-band conjugate gradient (CG) approach to solve the Kohn-Sham equations of DFT to obtain the wavefunctions of the electrons. A part of the calculations is carried out in real space and the remainder in Fourier space using specialized parallel 3D FFTs to transform the wavefunctions. The code spends most of its time in vendor supplied BLAS3 and 1D FFTs on which the 3D FFTs are built. For this reason, Paratec generally obtains a high percentage of peak performance on different platforms. The code exploits fine-grained parallelism by dividing the plane wave components for each electron among the different processors.

4.1. PARATEC Performance Results

Table 3 presents scaling tests for a 432 Si-atom bulk system for a standard LDA run of Paratec, for 5 CG steps of the iterative eigensolver, including the set-up and I/O steps. A typical calculation would require 20 to 60 CG steps to converge. Paratec on the ES runs about 10 times faster than the Power 3, achieving over 2 Tflop/s on 1024 processors - the fastest performance of any platform to date. The fast crossbar interconnect with extremely high bisection bandwidth, is the key to the high ES scalability. However, performance degrades for large number of processors since the ratio of time spent in communications versus computation in the FFT section increases; additionally the vector length for FFT and BLAS3 operations decreases as the data is distributed over more processors.

5. GTC: Magnetic Fusion

The goal of magnetic fusion is the construction and operation of a burning plasma power plant producing clean energy. The performance of such a device is determined by the rate at which the energy is transported out of the hot core to the colder edge of the plasma. The Gyrokinetic Toroidal Code (GTC) [4] was developed to study the dominant mechanism for this transport of thermal energy, namely plasma microturbulence. Plasma turbulence is best simulated by particle codes, in which all the nonlinearities are naturally included.

GTC solves the gyroaveraged Vlasov-Poisson (gyrokinetic) system of equations [5] using the particle-in-cell (PIC) approach. Instead of interacting with each other, the simulated particles interact with a self-consistent electrostatic or electromagnetic field described on a grid. Numerically, the PIC method scales as NlogN, instead of N^2 as in the case of direct binary interactions. Also, the equations of motion for the particles are simple ODEs (rather than nonlinear PDEs), and can be solved easily (e.g. using Runge-Kutta). The main tasks at each time step are: deposit the charge of each particle at the nearest grid points (scatter operation); solve the Poisson equation to get the potential at each grid point; calculate the force acting on each particle from the potential at the nearest grid points (gather operation); move the particles by solving the equations of motion; find the particles that have moved outside their local domain and migrate them accordingly.

The parallel version of GTC performs well on massive superscalar systems, since the Poisson equation is solved as a local operation. The key performance bottleneck is the scatter operation, a loop over the array containing the position of each particle. Based on a particle's position, we find the nearest grid points surrounding it and assign each of them a fraction of its charge proportional to the separation distance. These charge fractions are then accumulated in a grid array.

5.1. GTC Performance Results

Particle-in-cell codes are a challenge for all types of computers. While being the basis for the success and power of the PIC method, the grid-based charge deposition operation is also the source of the PIC codes limited processor efficiency observed on all superscalar and vector architectures. The need to avoid memory dependencies to achieve full vectorization of this scatter-type operation further increases the challenge. Fortunately, several methods have been developed to address of this issue during the past two decades. Our approach uses the work-vector method [6], where a temporary copy of the grid array is given an extra dimension corresponding to the vector length. Each vector operation acting on a given data set in the register then writes to a dif-

Table 4 GTC per processor performance using 100 particles per cell

P	Power 3		ES	
	MF/s	%peak	MF/s	%peak
32	135	9 %	1344	17 %
64	133	9 %	1245	16 %

ferent memory address, entirely avoiding memory dependencies. The only drawback of this method is the increase in memory used by the code, which can be from 4 to 8 times higher than for the superscalar version of the code.

By using the work-vector method along with data-flow improvement directives, a vector operation ratio of 98% and an AVL of 241 were achieved on the whole GTC code, resulting in significant increase in performance compared to the typical performance observed on the Power 3. Table 3 presents GTC performance using 100 particles per cell. Notice that the ES is up to 10 times faster than the Power 3, achieving 17% of peak compared with 9% on the Power 3

However, the increased memory footprint necessitated by vectorization inhibited the use of loop-level parallelism, thereby preventing large-scale simulation (coarse-grained domain decomposition is limited to about 64 processor). Nonetheless the ES 64-processor run was still 20% faster than using 1024 processors on the Power 3 (for the same test case), where loop-level parallelism was utilized on the Power 3 to attain the high processor count. In spite of the substantial increase in memory requirements, the ES gives the highest performance of all the platforms tested to date.

6. Conclusion

This work presented the performance of the ES cacheless vector architecture and compared it against the cache-based IBM Power 3 superscalar system, across a number of key scientific computations. Overall results demonstrate that the ES is significantly faster than the Power 3 architecture - in fact the ES per processor performance is higher than any other platform tested to date. However, the GTC particle-in-cell code represents a class of algorithms at odds with data-parallelism. Vectorizing this application greatly increased the memory footprint, preventing loop-level parallelism and limiting the potential for high scalability.

Acknowledgements

The authors would like to sincerely thank: the staff of the Earth Simulator Center, especially Dr. T. Sato, S. Kitawaki and Y. Tsuda, for their assistance during our visit; D. Parks and J. Snyder of NEC America for their help in porting applications to the ES.

All authors from LLBNL were supported by Director, Office of Computational and Technology Research, Division of Mathematical, Information, and Computational Sciences of the U.S. Department of Energy under contract number DE-AC03-76SF00098. S.E. was supported by the U.S. Department of Energy under contract number DE-AC020-76-CH03073. This research used resources of the National Energy Research Scientific Computing Center, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC03-76SF00098.

References

[1] P. Pavlo, G. Vahala, and L. Vahala, “Higher Order Isotropic Velocity Grids in Lattice Methods”, Phys. Rev. Lett., vol.80, pp.3960, (1998).

[2] G. Allen, T. Goodale, G. Lanfermann, T. Radke, E. Seidel, W. Benger, H.-C. Hege, A. Merzky, J. Masso, and J. Shalf, “Solving Einstein’s equations on supercomputers”, IEEE Computer, vol.32, pp.52–58 (1999).

[3] M. C. Payne, M. P. Peter, D. C. Allan, T. A. Arias, and J. D. Joannopoulos, “Iterative minimization techniques for ab initio total-energy calculations: Molecular dynamics and conjugate gradients”, Rev. Mod. Phys., vol.64, pp.1045–1098, (1993).

[4] Z. Lin, S. Ethier, T. S. Hamh, and W. M. Tang, “Size Scaling of turbulent transport in magnetically confined plasmas”, Phys. Rev. Lett., vol.88, pp.195004 (2002).

[5] W. W. Lee, “Gyrokinetic particle simulation model”, J. Comp. Phys., vol.72, pp.243 (1987).

[6] A. Nishiguchi, S. Orii, and T. Yabe, “Vector Calculation of Particle Code”, J. Comp. Phys., vol.61, pp.519 (1985).

[7] A. MacNab, G. Vahala, P. Pavlo, L. Vahala and M. Soe, “Lattice Boltzmann Model for Dissipative Incompressible MHD”, 28th EPS Conference on Contr. Fusion and Plasma Phys., vol.25A, pp.853–856, Funchal, Czech Republic, June 2001.

[8] National Energy Scientific Computing Center. http://www.nersc.gov

[9] Cactus Code Server. http://www.cactuscode.org

[10] PARAllel Total Energy Code. http://www.nersc.gov/projects/paratec