Quantum-Based Molecular Dynamics on Graphics Processing Units

Susan M. Mniszewski, CCS-3;
Anders Niklasson,
Ed Sanville,
Marc J. Cawkwell, T-1

The performance of quantum-based molecular dynamics (MD) simulations using the LANL-developed code LATTE is limited by the time required to compute the density matrix at each time step. The density matrix has been computed historically via the diagonalization of the Hamiltonian matrix. We have investigated the performance and accuracy of the second-order spectral projection (SP2) algorithm that calculates the density matrix via a recursive series of generalized matrix-matrix multiplications. Owing to its simplicity, the SP2 algorithm is ideally suited to implementation on graphics processing units (GPU). We demonstrate that optimized implementations of this algorithm on single and multiple GPUs on a single compute node lead to significant speed-ups with respect to central processing unit-based implementations and diagonalization. Furthermore, the SP2 algorithm produced an error rate in the density matrix that are as good or better than traditional methods.

MD is a popular and widely used simulation technique for the study of the evolution of a collection of atoms over time [1]. MD simulations require an interatomic potential that gives the potential energy of the system and the forces acting on each atom as a function of the relative coordinates of all of the atoms [2]. Interatomic potentials that describe explicitly the electronic structure of molecules and solids are the most physically accurate, yet their application in MD simulations has been severely limited by their prohibitive computational expense [3]. As a result, there is a constant demand for the development of better algorithms and computational methods for the acceleration of quantum-based interatomic potentials such that ever more challenging simulations become tractable.

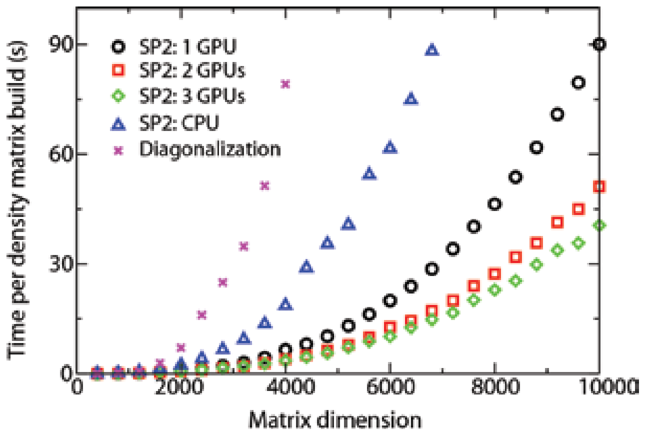

Fig. 1. Time per density matrix construction with LATTE via diagonalization and the SP2 algorithm on two hex-core Xeon processors and on multiple Nvidia C2090 GPUs on the Keeneland cluster at the National Institute for Computational Sciences (NICS).

Modern general purpose GPUs are attractive candidates for the acceleration of many compute-intensive applications on account of their very high memory bandwidth and peak number of floating point operations per second (FLOPs) relative to CPUs. However, GPUs are most efficient only when high levels of thread-level parallelism can be extracted from the algorithm, branching within warps can be avoided, and CPU-GPU communication can be minimized. Thus, not all algorithms are amenable to implementation on GPUs.

The density matrix, ρ, is a key quantity in the computation of the potential energy and interatomic forces in many quantum-based potentials. In dense matrix algebra, the time required for the computation of the M x M density matrix scales with the cube of the matrix dimension, $O(M^3)$ [3]. As a result, the calculation of the density matrix may dominate the total computational time, particularly for large systems. In the quantum-based MD code LATTE we have pursued an algorithm for the computation of the density matrix that avoids many of the intrinsic limitations of GPUs.

The SP2 algorithm [4,5] enables the density matrix to be computed directly from the Hamiltonian via an expansion of the Fermi operator,

$$\rho = \Theta[\mu \mathrm{I} - \mathrm{H}] = \lim_{i\to\infty} f_i[f_{i-l}[\ldots f_0[\mathrm{X}_0]\ldots]]$$

where Θ is the matrix Heaviside step function, **H** the Hamiltonian matrix, μ the chemical potential, I the identity matrix. X_0 is equal to the Hamiltonian matrix rescaled such that its eigenvalues occupy in reverse order the interval [0,1], and

$$f_i[\mathrm{X}_i] = \begin{cases} \mathrm{X}_i^2 & \text{if } 2\mathrm{Tr}[\mathrm{X}_i] \geq N_e \\ 2\mathrm{X}_i - \mathrm{X}_i^2 & \text{if } 2\mathrm{Tr}[\mathrm{X}_i] < N_e \end{cases}$$

where Tr denotes the matrix trace and N_e is the total number of electrons. The value of the chemical potential is adjusted automatically in the SP2 algorithm to obtain the correct occupancy.

The SP2 algorithm was ported to Nvidia GPUs using CUDA version 4.2 with Nvidia's CuBLAS implementation of the level 3 BLAS DGEMM subroutine for performing the generalized matrix-matrix multiplications, $\mathbf{C} \leftarrow \alpha\mathbf{AB} + \beta\mathbf{C}$, where **A**, **B**, and **C** are matrices and α and β are scalars.

For more information contact Marc J. Cawkwell at cawkwell@lanl.gov.

Funding Acknowledgments
DOE, Office of Science, Advanced Scientific Computing Research Program, ExMatEx Project; LANL, Laboratory Directed Research and Development Program

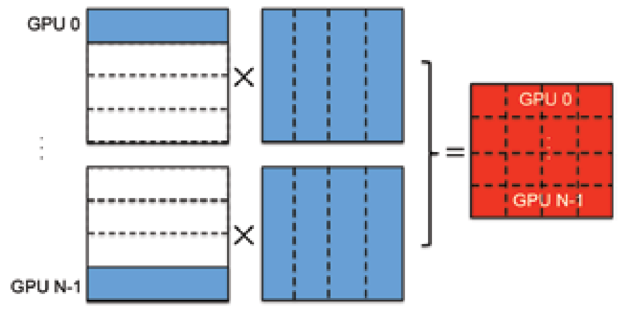

Fig. 2. Schematic illustration of the partitioning of sub-block matrix-matrix multiplications among N GPUs. Each GPU has a copy of the entire symmetric matrix.

We were able to minimize expensive communication between the CPU and GPU by porting the entire subroutine to the GPU such that we transfer only the matrix X_0 to the GPU at the start of the algorithm and pull back ρ once the recursive expansion has converged [5].

In Fig. 1 we present a series of timings for the computation of the density matrix for liquid methane in LATTE [6] using diagonalization and the SP2 algorithm on two hex-core CPUs versus our GPU implementations of the SP2 algorithm. Here it is evident that the performance of the implementation on one GPU exceeds those of an optimized CPU implementation and diagonalization even for relatively small systems [5]. Furthermore, an analysis of the errors in the final density matrices shows that the SP2 algorithm is as accurate or better than traditional diagonalization [5].

Since multiple GPUs are commonly installed on a single compute node, we have investigated a parallel GPU implementation of the SP2 algorithm. We designed and implemented a simple scheme for the parallelization of the generalized matrix-matrix multiplication that is illustrated schematically in Fig. 2. Here, the sub-blocks into which the matrices X_i are divided were purposefully kept as large as possible owing to the problem-size dependence of the FLOP rate of GPUs. The CuBLAS DGEMM was used to perform all of the sub-block matrix-matrix multiplications. The matrix traces and matrix-matrix additions were also performed in parallel across multiple GPUs. The timings presented in Fig. 3 show clearly that the use of multiple GPUs results in improvements in performance as the dimensions of the matrices increase. The overheads encountered for small matrices arise from the problem-size dependence of the FLOP rate of the GPUs and communication between GPUs via the CPU. Nevertheless, our results show a convergence toward ideal speed-ups as the matrix sizes increase with no loss of accuracy with respect to the CPU code or the implementation on one GPU.

In conclusion, hybrid multi-core CPU and multi-GPU computational architectures have been employed to significantly increase the performance of electronic structure and quantum-based MD simulations in LATTE. GPUs are particularly attractive for use as accelerators, especially when one considers their performance on a per Watt or per unit cost basis. However, users will enjoy performance gains only for those algorithms that are well tailored via computational co-design toward the intrinsic strengths of GPUs.

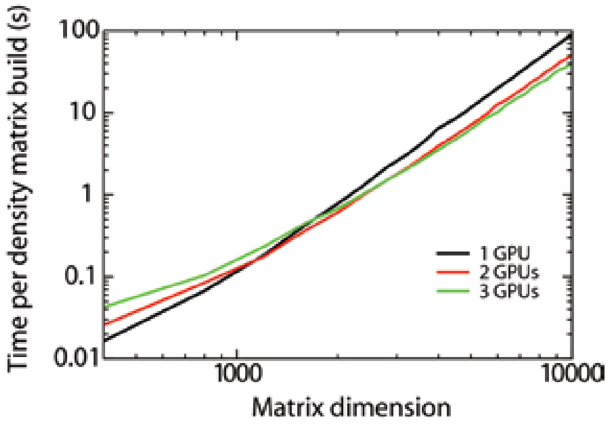

Fig. 3. Relative performance of the SP2 algorithm on one, two, and three Nvidia C2090 GPUs as a function of the problem size. Running on one GPU is fastest for small problems while three GPUs provide best performance when the dimensions of the density matrix exceed about 3000 x 3000 .

[1] Allen, M.P. and D.J. Tildesley, *Computer Simulation of Liquids*, Oxford University Press (1987).

[2] Finnis, M.W., *Interatomic Forces in Condensed Matter*, Oxford University Press (2003).

[3] Goedecker, S., *Rev Mod Phys* **71**, 1085 (1999).

[4] Niklasson, A.M.N., *Phys Rev B* **66**, 155115 (2002).

[5] Cawkwell, M.J. et al., *J Chem.Theory Comput*, **8**, 4094 (2012).

[6] Sanville, E.J. et al., "LATTE," LA-CC 10-004; http://savannah.nongnu.org/projects/latte (2010).