Accelerating MST using the GPGPU

Andrew Zigerelli

University of Pittsburgh

Motivation

In materials science, we study the structure of solid state materials, as the arrangment of atoms determines its mechanical and physical properties, which is important in fields such as chemistry and material engineering. For example, an engineer may want to produce an alloy with certain properties. Instead of physically making alloys with different combinations and amounts of metals, which is expensive, the engineer could simulate the alloy on a compuer and determine its properties. This simulation is achieved by using computational methods based on quantum mechanics.

The Physics

We are interested in a material's structure at its *ground state*, which is the lowest energy state. The structure is determined by the interactions between the particles in our physical system. There are different methods to investigate these interactions. We use an *ab initio method*, which is based on quantum mechanic principles, and no empirical data assumptions. Thus, ab initio methods rely on only the particle interactions, captured by the time independent **Shrödinger's Equation**, which is

$$\mathcal{H}\Psi = E\Psi$$

where $\mathcal{H}$ is the Hamiltonian operator, Ψ is the wave function, and E is the energy. The Hamilitonian is $\Delta + V$, where Δ is the Laplacian, and V is the potential field. Ψ is a function of all the particles in the system. Because the electrons are approximately fixed relative to the motion of the electrons, we can approximate the wave function via the **Born-Oppenheimer approximation**.

$$\Psi \approx \Psi(\text{nuclei})\Psi(\text{electrons})$$

Thus, we calculate Ψ(electrons). Unfortunately, the many electron Shrödinger's equation cannot be solved, as the resulting mathematical problem has too many variables (corresponding to the number of electrons in the system!). Fortunately, due to the work of Kohn and Sham in their development of density functional theory, we can transform the many elecron Shrödinger's equation into a single electron Shrödinger's equation, or Kohn-Sham equation with no loss of accuracy.

The **Kohn-Sham equation** is

$$(\Delta + v_{\text{eff}})\psi = \varepsilon\psi$$

where Δ is the Laplacian, v_{eff} is the *effective* one-electron potential, ψ is the single electron wave function (block wave), and ε is the energy parameter.

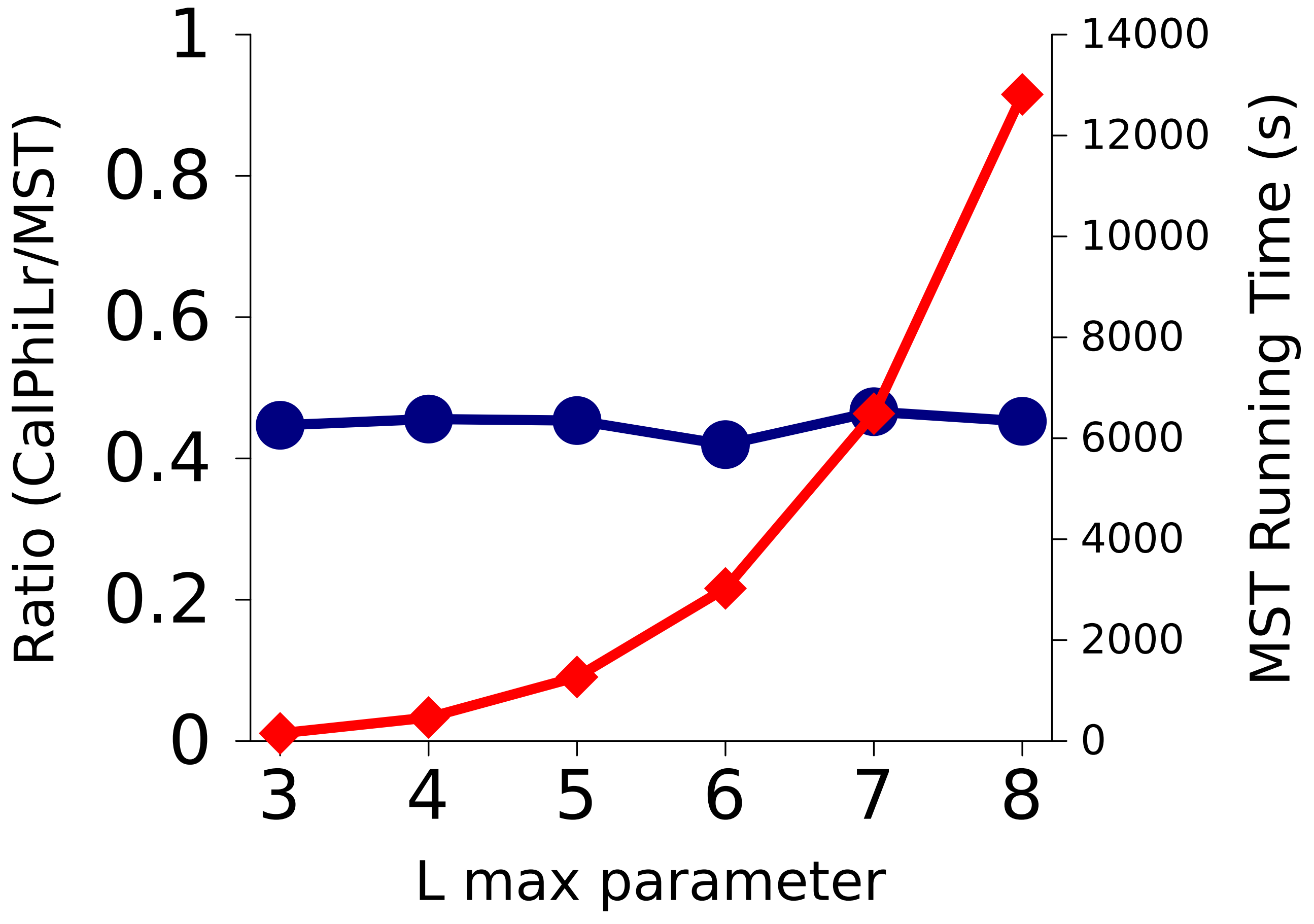
Solving the Kohn-Sham equation

There exists different methods to solve the Kohn-Sham equation. We use a method based on **Multiple Scattering Theory**. Multiple scattering theory invovles the idea that interactions between the nuclei and electrons "scatter" the electrons. In our method, we divide the physical space into cells. Each cell contains one scattering center, the nucleus. The method calculates the equations for each single scattering site independently and combines them together. This is advantageos because we may only need to calculate a few sites for materials with periodicity. For example, in metals, each site is essentially the same, so we only need to do the calculation at a single site once for a set of energy parameters.

The Current Implementation

Currently, there is a working implementation of multiple scattering method in a Fortran software package called **MST**. MST takes as input a physical system of atoms and the corresponding parameters, numerically calculates the Kohn-Sham equation, and determines the structure of the material which yields the lowest resulting energy (the ground state). The CalPhiLr subroutine in the single site scattering solver is currently the bottleneck of the package, which is something we would ultimately like to resolve.

The Bottleneck



CalPhiLr is 45% of the total running time!

CalPhiLr $= \Theta(L^3_{max})$

Parallelizing the Bottleneck

The multiple scattering theory method allows us to write the block wave function ψ as a linear combination of regular solutions ϕ_L local to each single scattering site. Thus, $\psi = \sum_L c_L\phi_L$. Each ϕ_L is a solution to the Shrödinger equation at a specific single scattering site. These regular solutions ϕ_L can be solved independently of each other by the CalPhiLr routine.

In each calculation of ϕ_L, there is opportunity for more parallelization. During the calculation, there are sums in which the terms are independent of each other, so they can be calculated at the same time.

GPU Implementation

The **Graphic Processing Unit**, or GPU, has traditionally been used for the sole purpose of displaying graphics. All other processing, including computation, has been handled by the **Central Processing Unit**, or CPU. Thus, most scientific computing packages are written to be run on the CPU. Recently, GPUs have been engineered not for the purpose of displaying graphics, but for the purpose of being used in scientific computing. GPUs allow for greater parallelism for arithmetic operations, resulting in a potential huge performance benefit. NVIDIA is the current leader in manufacturing GPUs for general purpose calculations. CUDA is NVIDIA's parallel computing platform and programming model for calculating on the GPU. CUDA supports a subset of the C programming language. Recently, the calculation of ϕ_L in CalPhiLr was ported to C. Currently, it is being implemented on the GPU using CUDA extensions to allow the single site wavefunctions to be computed independently. Future work includes optimizing the CUDA code to take advantage of the hardware through techniques such as memory coalescing.

Acknowledgements

This work is supported in part by the Center for Defect Physics, An Energy Frontier Research Center of the Department of Energy and by Pittsburgh Supercomputing Center under the direction of Yang Wang.

PITTSBURGH
SUPERCOMPUTING
CENTER