GPU implementation of energy minimization for virtual screening

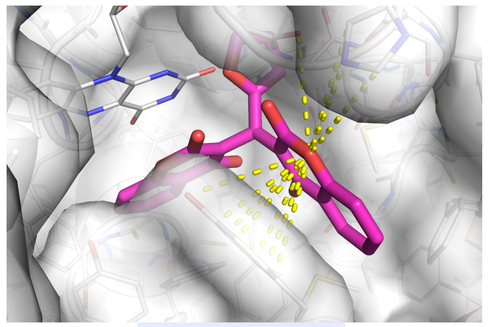
Jocelyn Sunseri, David Ryan Koes

Carnegie Mellon-University of Pittsburgh Joint Program in Computational Biology, University of Pittsburgh

Abstract

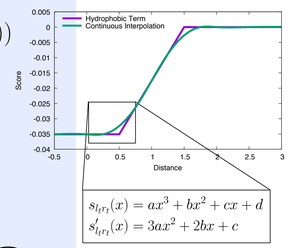
Energy minimization of a protein-ligand pose is an important part of structure-based virtual screening workflows, such as molecular docking and the refinement of pharmacophore aligned poses. Estimates of binding free energy are made via a scoring function, typically a sum of terms describing the interactions between the ligand and nearby receptor atoms, and a virtual screen will involve several thousand ligands. Since the calculations over all the atoms in a ligand as well as the minimization over multiple ligands are independent of one another, this process is highly amenable to parallelization as these calculations may all be performed concurrently. We report our progress in parallelizing the energy minimization algorithm used in smina, our molecular docking program which is a fork of Autodock VINA[1].

Energy Minimization

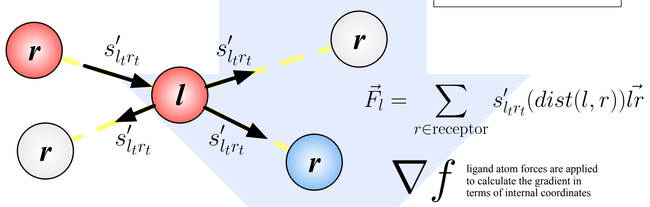


$$f(x_k) = \sum_{l \in \text{ligand}} \sum_{r \in \text{receptor}} s_{l_t r_t}(dist(l,r))$$

$x_k = k$ degrees of freedom that define ligand coordinates



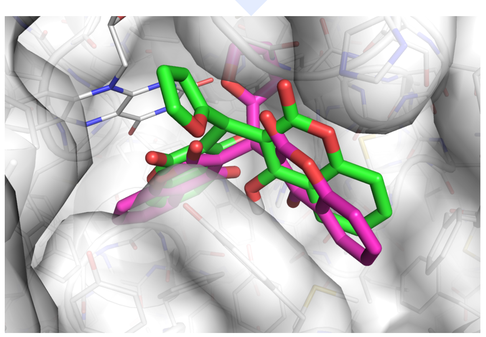
$s_{l_t r_t}$ precalculated piecewise cubic spline of distance dependent scoring term between atom types l_t and r_t



$$\vec{F}_l = \sum_{r \in \text{receptor}} s'_{l_t r_t}(dist(l,r))\vec{lr}$$

∇f ligand atom forces are applied to calculate the gradient in terms of internal coordinates

Broyden–Fletcher–Goldfarb–Shanno



Methods

We focused initial efforts on parallelizing scoring by computing the interaction between a given ligand atom and all relevant receptor atoms concurrently. This involves three reductions in total, and we therefore investigated fast parallel reduction algorithms. Following Mark Harris,[2] we settled on a warp-based reduction utilizing SHUFL intrinsics.

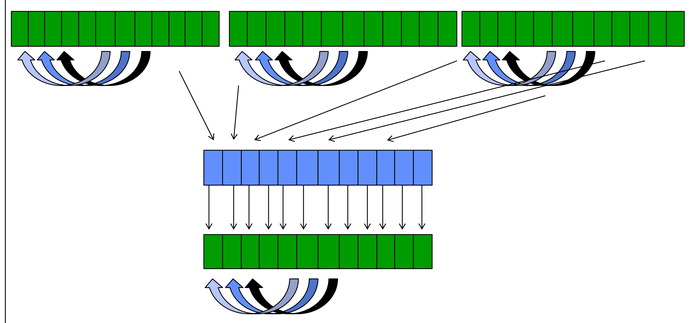


Figure 2. Warp-based SHUFL reduction. By using a warp intrinsic instruction, we minimized shared memory usage and pared down synchronization to a single instruction.

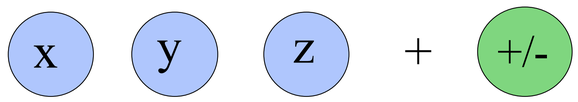


Figure 3. Packing three-dimensional data with a fourth datum in a float4-sized struct and using the __align__ keyword allows for coalesced reads and writes.

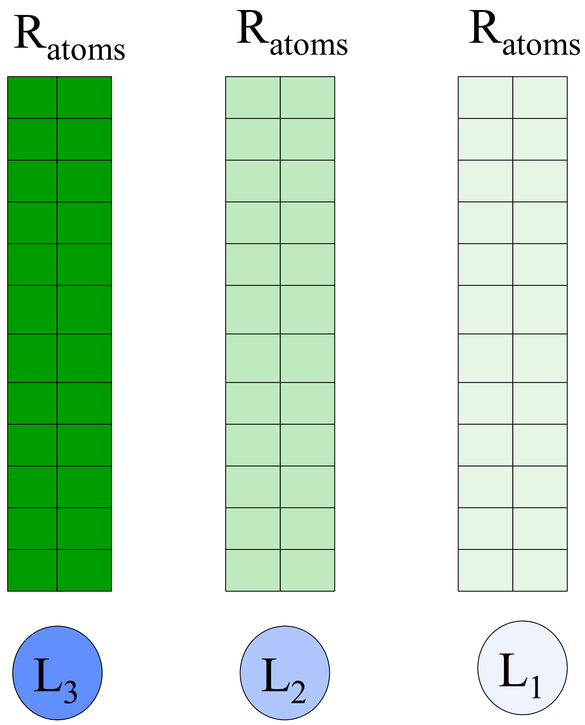
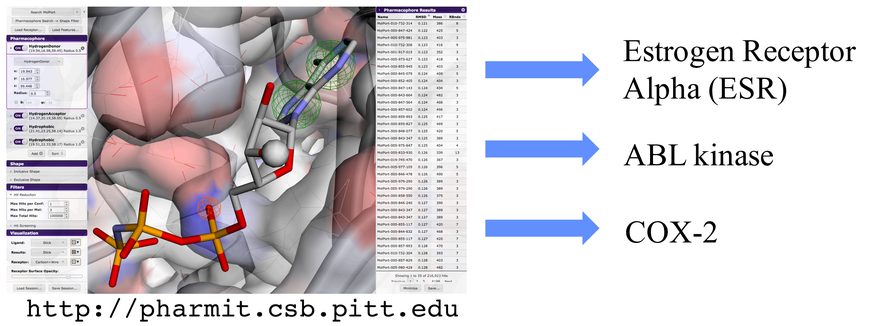


Figure 4. Parallelizing over receptor atoms, which compute the total contribution of each ligand atom. Each ligand atom has its interactions calculated by a **block** of threads, with one thread per receptor atom. Multiple blocks are used if there are more than 1024 receptor atoms.

Results



http://pharmit.csb.pitt.edu

Figure 5. Pharmacophore aligned poses targeting three different targets were generated using the Pharmit webserver, which utilizes smina for minimization. For each target, the top 2500 poses from the pharmacophore search were used for evaluating minimization performance.

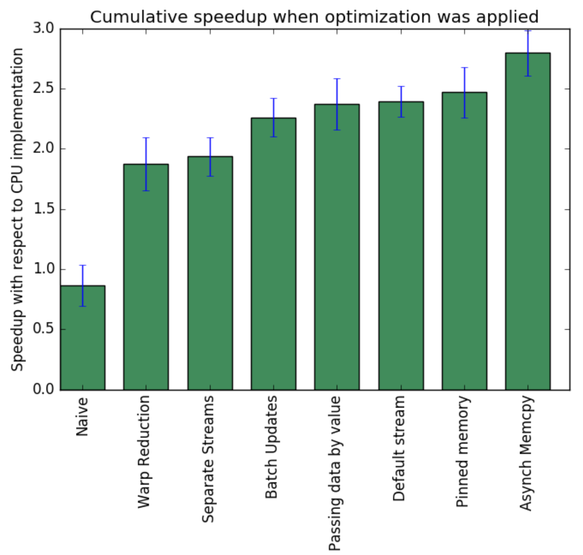
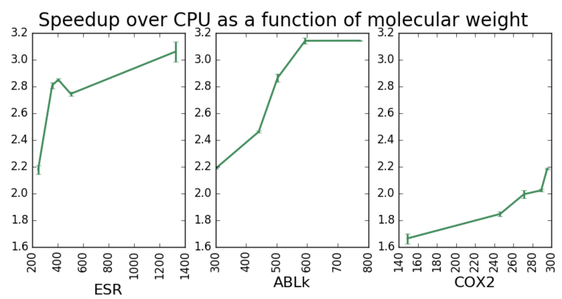


Figure 6. A variety of performance optimizations were applied to increase the scoring performance of our initial implementation by threefold. The largest gain was achieved by replacing existing reductions with a warp-based implementation, which reduced synchronizations as well as shared memory accesses.

Figure 7. The greatest performance improvement is observed with the largest ligands, indicating that the GPU is likely underutilized.



Future Work

- Reduce register usage to maximize SM occupancy
- Move the coordinate updates onto the GPU to minimize data transfer overhead
- Implement a parallel line search algorithm
- Run computations for separate ligands (identical receptor) concurrently

Acknowledgements

This work was supported by the National Institute of Health [R01GM108340]. The content is solely the responsibility of the authors and does not necessarily represent the official views of the National Institutes of Health.

References

[1] Trott, Oleg, and Arthur J. Olson. "AutoDock Vina: Improving the Speed and Accuracy of Docking with a New Scoring Function, Efficient Optimization and Multithreading." *Journal of computational chemistry* 31.2 (2010): 455–461.*PMC*. Web. 15 Mar. 2016.
[2] Harris, Mark. "Optimizing Parallel Reduction in CUDA." NVIDIA Developer Technology. https://docs.nvidia.com/cuda/samples/6_Advanced/reduction/doc/reduction.pdf