Practical Diamond Tiling for Stencil Computations Using Chapel Iterators

Michelle Mills Strout (presenter), Ian J. Bertolacci, and Catherine Olschanowsky
from Colorado State University
Ben Harshbarger and Bradford L. Chamberlain from Cray Inc.
David G. Wonnacott from Haverford College

1 CHIUW Talk Abstract

Stencil computations make up the core of many scientific computations, often as partial differential equation solvers used in science and engineering simulations. However, typical data-parallel schedules of stencil computations parallelize each time-step in isolation; this ignores the potential for leveraging data locality between time steps. Advanced tiling techniques, such as diamond tiling, strike a balance between parallelism and data locality, effectively improving parallel scaling on multicore processors.

However, advanced tiling techniques are difficult to include in general purpose optimizing compilers because of the need for interprocedural pointer and array data-flow analysis plus the need to tune scheduling strategies and tile size parameters for each pairing of stencil computation and machine. Such techniques are also difficult to incorporate into code by hand.

In this talk, we present an advanced, parameterized tiling approach that was implemented using Chapel iterators. This work will be presented the week before CHIUW at ICS [1]. We show how such iterators can be used by programmers in stencil computations with multiple spatial dimensions, and that these new iterators provide better scaling than a traditional data parallel schedule.

Currently, performance programmers perform these kinds of transformations manually for performance tuning and prototyping purposes [2]; however, this approach causes significant code maintenance issues. Moreover, the tiled loop structures obfuscate the original stencil computation being performed to the point of making them virtually unrecognizable. We have developed a systematic method for creating diamond tiling code that is stencil/-data dependency specific, and parameterized for tile size. Of most interest to CHIUW will be that we show how to implement diamond tiling code using Chapel iterators including a comparison of the programmability benefits of using Chapel parallel iterators as compared to the equivalent OpenMP code.

References

[1] I. Bertolacci, C. Olschanowsky, B. Harshbarger, D. Wannacott, B. Chamberlain, and M. Strout. Parameterized diamond tiling for stencil computations with chapel iterators. In *To appear in ACM International Conference on Supercomputing (ICS)*, 2015.

[2] C. Olschanowsky, M. M. Strout, S. Guzik, J. Loffeld, and J. Hittinger. A study on balancing parallelism, data locality, and recomputation in existing pde solvers. In *To be published in The IEEE/ACM International Conference for High Performance Computing, Networking, Storage and Analysis (SC)*, November 2014.