Orthogonal Scheduling of Stencil Computations with Chapel Iterators

Ian Bertolacci
Computer Science Department
Colorado Statue University
Ft. Collins, Colorado 80523
Email: ibertola@cs.colostate.edu

Michelle Mills Strout
Computer Science Department
Colorado Statue University
Ft. Collins, Colorado 80523
Email: mstrout@cs.colostate.edu

I. Summary

Stencil computations are common in scientific computing, often as part of partial differential equation solvers. These computations involve iterating across all points in a space and applying a kernel to update to the next time step. While straight-forward parallel implementations are easy to develop, they show poor multicore scaling. There have been a number of tiling techniques developed that show much better scaling; however, they are not available in general purpose compilers, and implementing these techniques by hand is difficult due to the high complexity of the loop bound expressions.

A more preferable solution might be the development of a library that provides programmers with optimized schedules in the form of some tiling construct. The Chapel language, developed at Cray Inc., includes a language feature called an iterator, where a loop structure can iterate over the values yielded by the iterator. Several Chapel iterators were written that yield indices of a tiling schedule to the loop performing the computation, thereby decoupling the implementation of the schedule from the computation. It was found that no performance penalty was incurred by using the iterator, and that the multicore scaling was substantially better than that of a straight-forward implementation.

In my poster, I demonstrate the power of this schedule-by-iterator idiom by comparing the scale-up performance and source code complexity of various implementations against each other, as well as comparing against OpenMP C implementations as a performance baseline.