Cy P Chan

A Dynamic Programming Approach to Autotuning Multigrid

44 Winter St #1
Cambridge
MA 02141
cychan@csail.mit.edu
Jason Ansel
Massachusetts Institute of Technology, Department of Electrical Engineering and Computer Science, Computer Science and Artificial Intelligence Laboratory
Yee Lok Wong
Massachusetts Institute of Technology, Department of Mathematics
Saman Amarasinghe
Massachusetts Institute of Technology, Department of Electrical Engineering and Computer Science, Computer Science and Artificial Intelligence Laboratory
Alan Edelman
Massachusetts Institute of Technology, Department of Mathematics, Computer Science and Artificial Intelligence Laboratory

Algorithmic choice is essential in any problem domain to realizing optimal computational performance. Multigrid is a prime example: not only is it possible to make choices at the highest level, but a program can switch techniques as the problem is recursively attacked on coarser grid levels to take advantage of algorithms with different scaling behaviors. Additionally, users with different convergence criteria must experiment with parameters to yield a tuned algorithm that meets their accuracy requirements. Even after a tuned algorithm has been found, users often have to start all over when migrating from one machine to another.

We present a programming language and autotuning system that addresses all of these issues in a near-optimal and efficient fashion. The freedom of independently tuning both the algorithm and the number of iterations at each recursion level results in an exponential search space of hybrid algorithms that have different accuracies and performances. To search this space efficiently, our autotuner utilizes a novel dynamic programming method to build near-optimal hybrid algorithms from the bottom up. The result is a tuned hybrid algorithm that invests minimal targeted computational power to yield the accuracy required by the user.

The techniques we describe allow the user to automatically generate tuned "W-

cycles" of different shapes targeted to the user's specific combination of problem, hardware, and accuracy requirements. These cycle shapes dictate the order in which grid coarsening and grid refinement are interleaved with both iterative methods, such as Jacobi or Successive Over-Relaxation, as well as direct methods, which tend to have superior performance for small problem sizes. The need to make choices between all of these methods brings the issue of variable accuracy to the forefront. Not only must the autotuning framework compare different possible W-cycle shapes against each other, but it also needs the ability to compare W-cycles against both direct and (non-multigrid) iterative methods. We solve this problem by using accuracy as a measure for the effectiveness of tuned W-cycle shapes and making comparisons over all algorithmic types based on this common yardstick. In our results, we find that the flexibility to trade performance versus accuracy at all levels of recursive computation enables us to achieve outstanding performance on a variety of platforms compared to algorithmically static implementations of multigrid.

Our implementation uses PetaBricks, an implicitly parallel programming language where algorithmic choices are exposed in the language. The PetaBricks compiler uses these choices to analyze, autotune, and verify the PetaBricks program. These language features, most notably the autotuner, were key in enabling our implementation to be clear, correct, and fast.