Cache Aware Algorithms and PDE Libraries

Craig C. Douglas
University of Kentucky
Computer Science Department
&
Yale University
Computer Science Department
http://www.mgnet.org/~douglas

Years ago, knowing how many floating point multiplies were used in a given algorithm (or code) provided a good measure of the running time of a program. This was a good measure for comparing different algorithms to solve the same problem. This is no longer true.

Many current computer designs, including the node architecture of most parallel supercomputers, employ caches and a hierarchical memory architecture. Therefore the speed of a code (e.g., multigrid) depends increasingly on how well the cache structure is exploited. The number of cache misses provides a better measure for comparing algorithms than the number of multiplies. Unfortunately, estimating cache misses is difficult to model a priori and only somewhat easier to do a posteriori.

Typical multigrid applications are running on data sets much too large to fit into the caches. Thus, copies of the data that are once brought into the cache should be reused as often as possible. For multigrid, the possible number of reuses is always at least as great as the number of iterations of the smoother or rougher (plus the residual correction before correction steps).

In this talk, suitable fixed and adaptive blocking strategies for both structured and unstructured grids will be introduced. Both types of cache aware algorithms use a fixed, given percentage of the cache.

Fixed algorithms use fixed blocks of the unknowns (or subdomains) that are determined in a preprocessing step. Adaptive algorithms use a moving active set of unknowns that should be in cache and can be reused. Once an unknown has been completely updated for k iterations, it leaves the active set.

The cache aware algorithms improve the cache usage without changing the underlying algorithm. In particular, bitwise compatibility is guaranteed between the standard and the high performance implementations of the algorithms. This is illustrated by comparisons for various multigrid algorithms on a selection of different computers for problems in two and three dimensions.

Key words: Computer architectures, iterative algorithms, multigrid, high performance computing.