Application of loop order optimization techniques to increase performance of grid-characteristic method*

A.M. Ivanov, N.I. Khokhlov

Moscow Institute of Physics and Technology (MIPT), Russia

We consider a variety of techniques to take advantage of the modern CPU memory system and to increase performance of elastic wave modeling code based on the solution of elastic wave equations with the grid-characteristic method. We use model of isotropic linear elastic material. This approximation of a more general non-linear model is used for numerical modeling in tasks connected with dynamic wave disturbances in seismic exploration. More thorough description is given in [1]. This method is implemented as iterations over the computational grid in space and time. We use an explicit scheme and a five-point stencil (fig. 1), which enables a set of loop optimizations.

The idea of performance improvement in considered code is to change order of loop iteration through space-time computational domain. Due to the specifics of our grid-characteristic method [2] implementation, we split each time step in separate substeps over every space dimension. Therefore it is difficult to change the order of iterations. Our approach of joining nodes to apply the following optimizations is shown on fig. 2.

Loop tiling with a square blocks of a fixed size is considered. The performance of this implementation is compared to diamond tiling [3] technique (fig. 3). Both methods require constant size of tile as a parameter, which should be adjusted according to the size of different cache levels on the specific CPUs. In addition, we demonstate the performance of recursive tiling method based on hierarchy of tiles [4] which automatically adapts to specific cache size without user interaction (fig. 4).

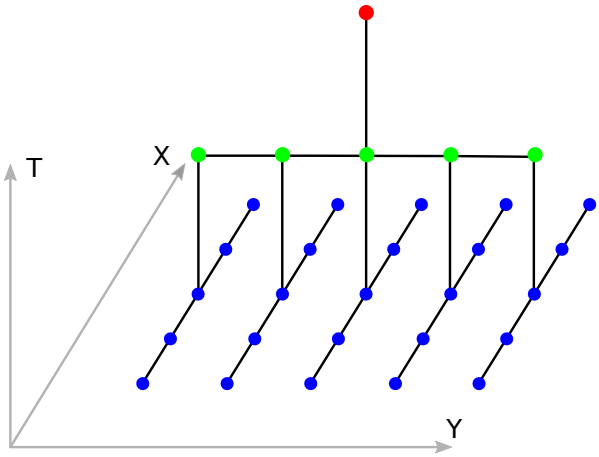

Figure 1. A stencil for 4th order accuracy. So the point at $n+1$ X layer (red) depends on 5 points at n Y layer (green) and each Y layer point depends on 5 points at n X layer (blue).

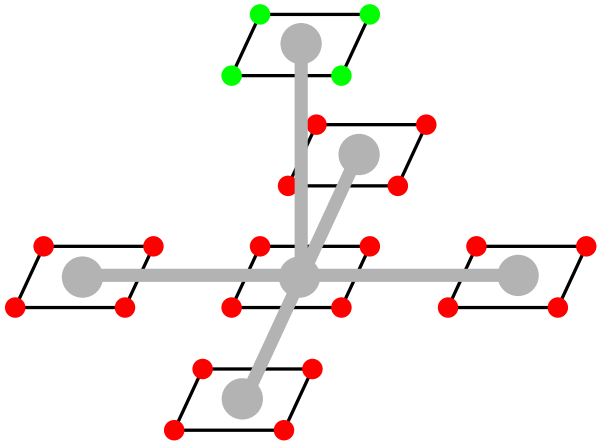

Figure 2. Five point stencil for diamond tiling. Each node (grey) represents 4 nodes (red and green) on initial grid. Red nodes - current substep (X or Y), green nodes - next substep

As shown by practice (table 1), there is no benefit of using diamond or hierarchical tilings in this kind of task (however it can improve parallelism, which is not considered in this work), because it generates an array traversal which is less friendlier to cache, possibly because it is

*The work was supported by the Ministry of Education and Science of the Russian Federation (no. 2.9901.2017/8.9).

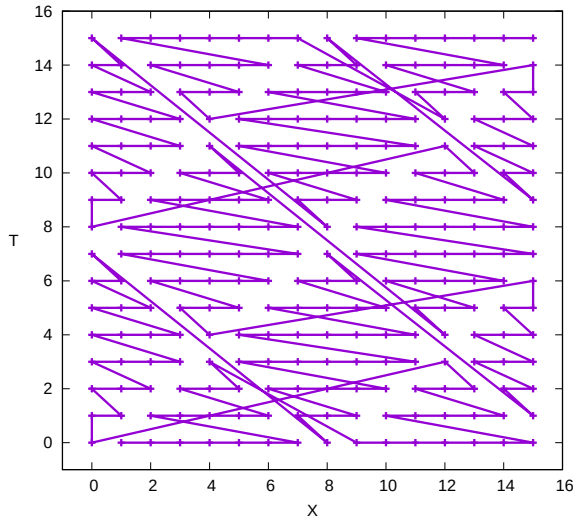

Figure 3. Example of diamond tiling on 1-dimensional grid with $h = 8$

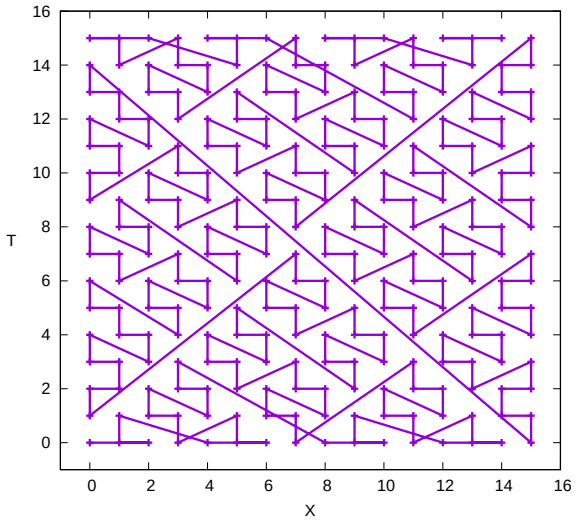

Figure 4. Example of hierarchical tiling on 1-dimensional grid

more difficult for hardware prefetcher to predict access patterns defined by tiles. It is important to notice that in our problem statement we have to use large amount of memory for each grid node, therefore too few nodes can be placed in cache simultaneously.

In conclusion, only the first approach with manual tiling by blocks of fixed width gives the highest performance benefits. Probably we should consider tiles not as cubes in diamond tiling, but as some other shapes, which will be axis aligned. Presumably it will help to store more adjacent nodes in cache.

Table 1. Comparison of performance of different tiling methods for grid size 2000×2000 and 500 time steps (Intel Core i7-4820K, 3.70GHz; gcc compiler, "-O2" optimization, single thread)

Tiling type	Time without tiling, s	Time with tiling, s	Speedup, %
Block	189.70	161.76	15
Diamond		201.81	-6
Hierarchical		215.18	-13

References

1. LeVeque, R.J.: Finite volume methods for hyperbolic problems, vol. 31. Cambridge university press. (2002)

2. Favorskaya, A.V., Petrov, I.B. (eds.): Grid-Characteristic Method, pp. 117–160. Springer International Publishing, Cham (2018).

3. Bertolacci, I.J., Olschanowsky, C., Harshbarger, B., Chamberlain, B.L., Wonnacott, D.G., Strout, M.M.: Parameterized diamond tiling for stencil computations with chapel parallel iterators. // ICS '15, ACM. pp. 197–206. (2015)

4. Carter, L., Ferrante, J., Hummel, S.F.: Hierarchical tiling for improved superscalar performance. In: Proceedings of 9th International Parallel Processing Symposium. pp. 239–245 (1995)