11th World Congress on Computational Mechanics (WCCM XI)
5th European Conference on Computational Mechanics (ECCM V)
6th European Conference on Computational Fluid Dynamics (ECFD VI)
July 20–25, 2014, Barcelona, Spain

THREAD-PARALLEL MESH IMPROVEMENT USING FACE AND EDGE SWAPPING

Reza Zangeneh[1] **and Carl Ollivier-Gooch**[2]

[1] MASc Student, Mechanical Engineering Department, University of British Columbia, r.zangeneh87@gmail.com

[2] Professor, Mechanical Engineering Department, University of British Columbia, cfog@mech.ubc.ca

Key words: *Mesh improvement, Face and edge swapping, Thread-parallel, OpenMP.*

Mesh generation and adaptation form the basis of finite volume and finite element analysis. Automatic unstructured mesh generation tools are used extensively for complex three-dimensional domains and parallel adaptive solution algorithms are maturing as well. However their performance is limited by the difficulty of parallelizing mesh adaptation. There has been significant effort to optimize parallel mesh adaptation codes for distributed memory architectures[1, 2], but little or no work on scalable algorithms for multi-core shared memory architectures despite the current trend towards ever higher core exists.

In this paper, we present a new thread-parallel edge and face swapping algorithm for three dimensional meshes using OpenMP. Serial swapping algorithms act on one face at a time, deciding whether to swap it and then performing the reconfiguration. In our thread-parallel implementation, we first identify all faces that should be swapped, then do all the reconfiguration. Identification is a read-only operation, and so threads. Reconfiguration writes to the mesh data base, and so problems with mesh consistency and race condition arise. To avoid this problem, thread-local containers are introduced so that the pool of unused entities will be distributed between threads. Hence, each thread has its own pool of unused mesh entities, while still being able to read from the whole shared mesh. This approach bypasses the need for locks while creating and deleting mesh entities, which can compromise the overall speedup. Instead, updates to global unused entity pool are deferred until the end of the parallel section. Experimental results showed that the cost of synchronization at entry and exit from the parallel sections is about 2-3% of the overall runtime.

Each face selected for swapping has a neighborhood of tetrahedra that will be replaced by new ones. To avoid mesh inconsistency the faces selected must be far enough from each other that their tetrahedra do not have any vertices in common. We enforce this during an intermediate stage that applies a vertex locking strategy, rejecting some faces for reconfiguration until a later pass. After this, the actual reconfiguration is guaranteed to be conflict-free regardless of order of operations. Finally a dynamic load balancing method[3] is implemented so that faces from highest load threads migrate to the threads with lower load prior to each parallel pass. This approach requires few transfers, yet guarantees load balance.

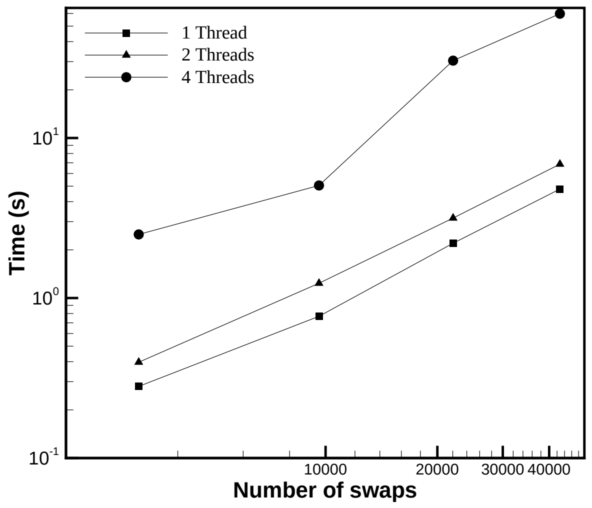

Figure 1: Speedup results in an quad-core machine

The parallel algorithm has been implemented in an automatic mesh generation library for unstructured meshes with mixed element types called Generation and Refinement of Unstructured, Mixed-Element Meshes in Parallel (GRUMMP). Figure 1 shows the preliminary speedup results on an up-to-date machine, featuring Intel core i7 quad-core at 3.60 GHz and 16 GB of RAM. The results presented here are the outcome of an experiment on a cubic box with different resolutions. The detailed algorithm and thorough results will be presented in the final paper.

REFERENCES

[1] L. Oliker, R. Biswas, and H.N. Gabow. Parallel tetrahedral mesh adaptation with dynamic load balancing. *Parallel Computing*, 26:1583–1608, 2000.

[2] R. Said, N. P. Weatherill, K. Morgan, , and N. A. Verhoeven. Distributed parallel Delaunay mesh generation. *Comp. Meth. Appl. Mech. Eng.*, 177:109–125, 1999.

[3] A. Vidwans, Y. Kallinderis, and V. Venkatakrishnan. Parallel dynamic load-balancing algorithm for three-dimensional adaptive unstructured grids. *AIAA J.*, 32:497–505, 1994.