17th International Conference
ENGINEERING MECHANICS 2011
Svratka, Czech Republic, 9 – 12 May 2011

PARALLELIZATION OF THE DELAUNAY TRIANGULATION

D. Krybus[*], B. Patzák[*]

Abstract: *This paper deals with the parallelization of Delaunay triangulation. Meshing algorithms are relatively time and memory consuming for large problems. In connection with the spread of multi-core CPUs even among personal computers, the parallelization of the meshing process permits optimal utilization of available computational resources. The triangulation is used for the mesh generation in the context of numerical modeling of fluid flow via Lagrangian finite element method. The possibility of the reuse of the previously developed sequential code in terms of parallelization is discussed and other possible approaches are outlined.*

Keywords: ***Delaunay triangulation, mesh generation, parallelization, shared memory, distributed memory.***

1. Introduction

Typical approach in modeling fluid flow consists in the use of Eulerian description of motion. The computational mesh is fixed and the material points move with the respect to the grid. In Lagrangian description, each node of the computational mesh is associated with the material particle during the motion. Arbitrary Lagrangian – Eulerian (ALE) formulation combines these two classical approaches mentioned in order to gain their best features and avoid their disadvantages (Donea & Huerta, 2004).

The fact that the movement of a domain occurs separately from moving a finite element mesh, is common for both ALE and Eulerian approaches. The relationship between them introduces convective terms of movement in momentum equations. Their treatment is not the only numerical difficulty coming with ALE and Eularian approaches. Also the proper treatment of the incompressibility condition, tracking of a free surface, correct interaction on a phases contact or description of large movements need to be secured.

The use of Lagrangian description has several advantages and some of the mentioned difficulties are not involved. Usually, it is widely applied for modeling large deformations in structural mechanics. Perhaps the main drawback of this approach consists in the frequent demands for re-meshing in order to avoid numerical instability due to large distortions of computational mesh. On the other hand, tracking of free surfaces and material interfaces is possible in a natural way. The Lagrangian formulation of the problem is set by a collection of so-called particles which represents a certain part of continuum. To integrate governing equations in each solution step, a mesh built from particles is needed. The mesh is generated by connecting particles together, actual positions of which are given by initial conditions or are determined by solution procedure. In this work, a triangulation of set of particles is used, as triangular elements are simple to formulate and numerically efficient. The algorithm based on Delaunay triangulation has been chosen.

The requirements for the parallelization of the meshing process are hard mainly due to relatively high time and memory demands of these algorithms. The parallelization allows obtaining results in reasonable time by utilization of potential of modern multi-core processing units. Nowadays, almost every personal computer or laptop comes with a multi-core CPU. It is natural and reasonable to use these available resources.

[*] Ing. David Krybus and prof. Dr. Ing. Bořek Patzák: Department of Mechanics, Faculty of Civil Engineering, Czech Technical University in Prague; Thákurova 7; 166 29 Prague; CZ, e-mails: david.krybus@fsv.cvut.cz, borek.patzak@fsv.cvut.cz

2. Delaunay triangulation

Many algorithms for mesh generation are based on Delaunay triangulation. It consists in such triangulation of a general set of points that no point lies inside circumscribed circle of any triangle except three points forming triangle, that actually lie on circle. Delaunay triangulation maximizes the minimum angle in triangles and it leads to a quite numerically stable mesh. This definition can by simply extended to a three-dimensional space, where the triangulation results in set of tetrahedrons with empty circumscribed spheres.

There are a lot of algorithms constructing the Delaunay triangulation. Incremental algorithms build a large group. In each step, a new point is added to the current triangulation, and then the mesh is rebuilt in order to satisfy the Delaunay condition. Flipping algorithms consist in changing, so called "flipping"' non-Delaunay triangle edges. At the beginning, a general mesh is constructed. In the next step, it is optimized with the flipping to satisfy the Delaunay property. Other methods are based on the "divide and conquer" or the projection algorithms. The different approaches are discussed and compared in the Shewchuk's paper (1996).

Delaunay triangulation using Bowyer/Watson incremental algorithm was implemented by authors and results were presented in the past (Krybus & Patzák, 2010). The meshing code is fully integrated into the OOFEM package (Patzak & Bittnar, 2001). It uses an octree data sorting in order to speed-up the search for non-Delaunay triangles during the mesh build-up phase.

3. Parallelization strategies

In general, several approaches to parallelization of Delaunay triangulation can be found in the literature. A very common way is the divide-and-conquer strategy, first proposed by Guibas & Stolfi (1985). It is based on recursive partitioning of the point sets and local triangulation of the subsets. The global triangulation results from merging in the final phase. Regarding these facts, this algorithm represents a naturally parallelizable approach. The points can be partitioned into certain regions upon their position in space. Next, each partition is assigned to a processor which builds the local triangulation. On the other hand, the merging phase is neither natural nor easy and in fact, it is the bottleneck of the method. In addition, the merging phase must be done only by one processing element with a considerable influence on the overall performance.

Other approaches are based on the concept of incremental insertion. To avoid multiple access to same data structure, the whole set of points is first partitioned into sub-regions. A load balancing technique must be employed in order to obtain reasonable speed up for general data (Chrischoides & Sukup, 1996). The Delaunay triangulation can be constructed incrementally by starting with the first triangle and building-up the triangulation in direction given by the outer normal vector of every edge. The "closest" point giving the "smallest" circumcircle with the edge is searched during the step (Cignoni et al. 1993), see Fig. 1. Finally, the parallelization of incremental approaches can be done without assigning regions to single processors. Kohout and Kolingerova (2003) developed an algorithm working with a shared global list of points, which is accessed by particular CPUs.

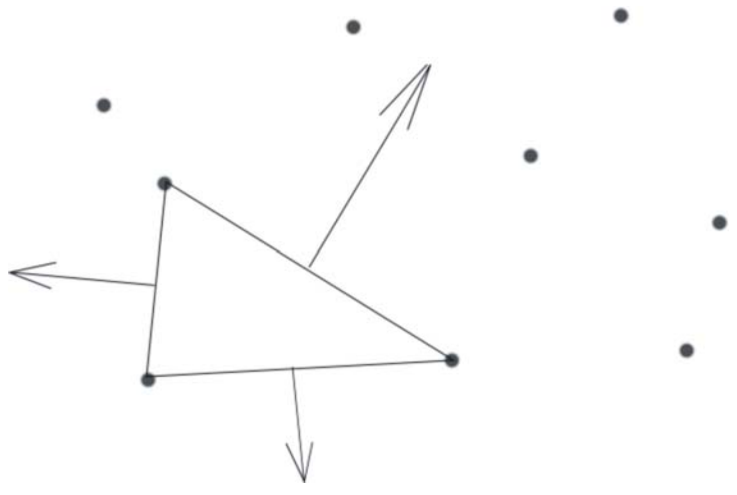
Fig. 1: Incremental construction.

3.1. Shared memory systems

Regarding the spread of multiple core personal computers, the first idea was to reuse the existing code and to improve it in order to allow its parallelization. The incremental algorithm is based on the insertion of the points on by one into the current triangulation. In one step, all triangles violating the Delaunay empty circle property are erased and the cavity is re-triangulated.

Considering shared memory parallel computer, the triangulation itself and its data structure is typically managed in the shared memory. When individual points are concurrently inserted, overlapping of the different cavities must be avoided in order to secure uniqueness of the triangulation or even crash of the triangulating execution. For example, consider the dark grey filled triangle in Fig. 2, the Delaunay property of which is affected by both inserted points, potentially leading to data access collision that must be resolved. Moreover, sophisticated data structures, i.e., spatial containers are usually used to allow fast search for non-Delaunay triangles. The point is that one must optimally ensure concurrent data access without excessive data locking, which is otherwise needed to obtain correct triangulation. Consequently, the scalability of the parallel code is affected in a negative way by these aspects.

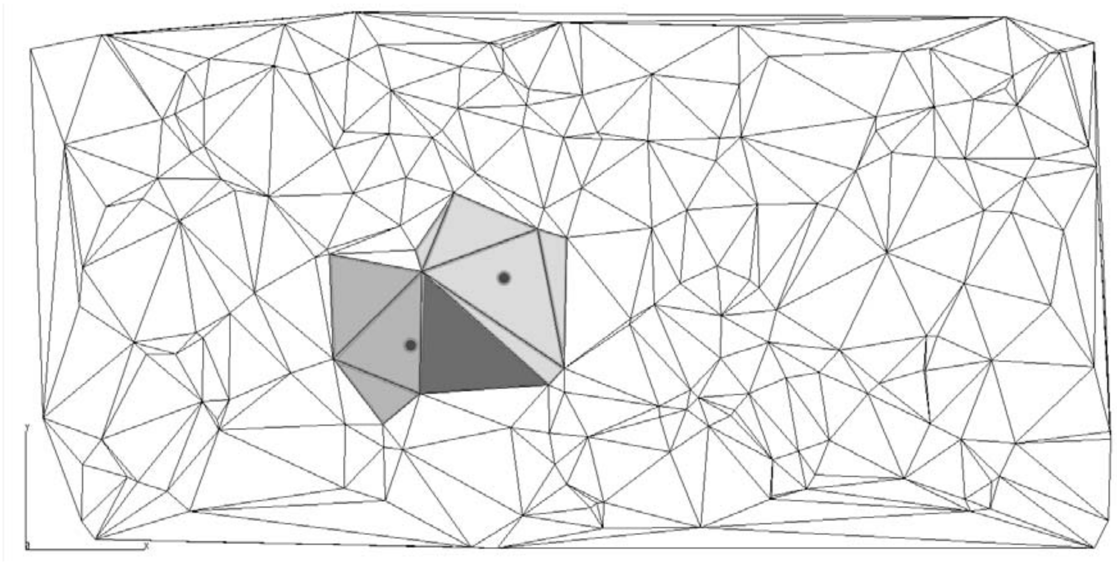

Fig. 2: Triangulation of overlapping cavities.

3.2. Systems with distributed memory

The original Bowyer/Watson algorithm is not very suitable for parallelization on distributed systems. In this case, each computational node manages the data stored in the own memory and any remote data has to be obtained by an explicit communication, contrary to shared memory systems, where data are globally accessible. However, the problem can be decomposed into sub-domains, meshed by single CPUs. This allows performing triangulation in parallel, but there is a significant data transfer between neighbouring domains to secure proper triangulation of the region borders. Every additional communication leads to a lower parallel efficiency of the resulting code.

A smart solution was proposed by Cignoni et al. (1993). Referring to the general classification, it is an incremental constructive algorithm. Contrary to the refining the triangulation by inserting points in Bowyer's /Watson's approach, it starts with one triangle which is final taking into account the resulting triangulation. The other triangles grow from the very first one, until all points are connected. This approach is reused in parallel context, where the idea consists in triangulating the region borders by the constructive algorithm first, so no more synchronization between processors is needed and until merging individual processors can work independently. Every processing unit responsible for triangulation of a certain area obtains a list of points contained and a "boundary" formed by appropriate edges of truly Delaunay triangles on the border. This is illustrated on Fig. 3, showing sub-domain boundary triangulation of rectangular domain, when four CPUs are considered. This step is performed sequentially. Then all processors receive corresponding part of boundary triangulation, list of particles inside each sub-region and start triangulation, which can be performed by any algorithm.

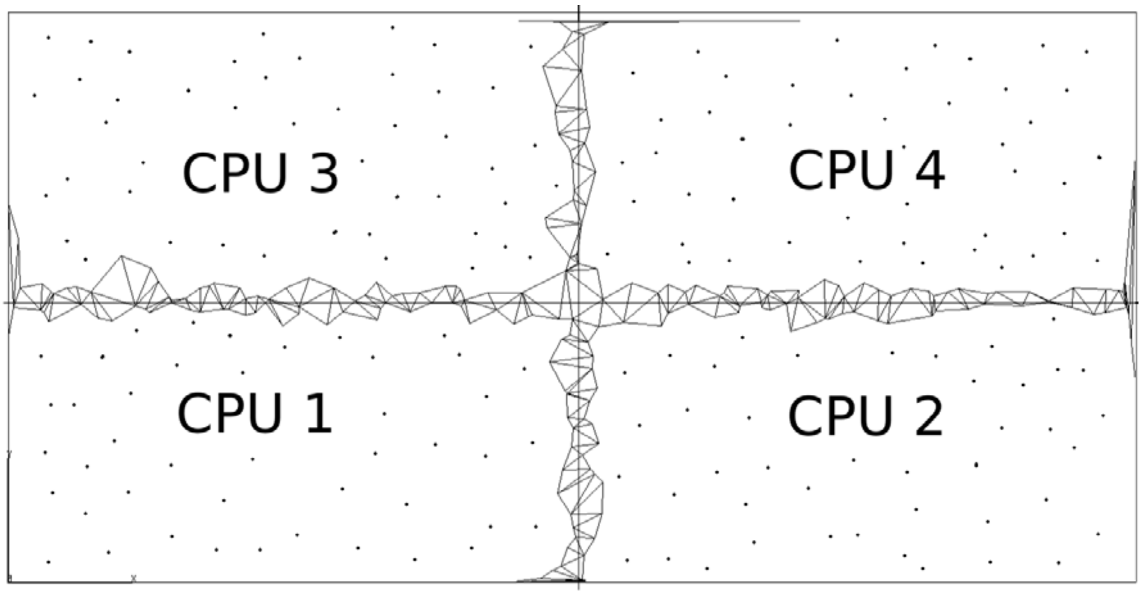

Fig. 3: Triangulation of regions' borders.

4. Conclusions

In this work, Delaunay triangulation is used to construct a finite element mesh from particles representing a Lagrangian formulated fluid problem. Two different approaches to parallelization of Delaunay triangulation were described. The difficulties resulting from parallelization of the basic sequential algorithm were discussed. An approach based on the incremental construction algorithm suitable for the use on systems with distributed memory is proposed. Implementation of parallel algorithms, both the original Bowyer/Watson algorithm as well as the incremental construction, is the objective of present research. The task is not just the scalability of the algorithms but also their overall efficiency in comparison to existing codes. Future work will be focused on examination of practical application of meshing algorithm on fluid problems to answer the question whether a creating of a completely new mesh from scratch is faster than improvement of the distorted one.

Acknowledgement

Financial support of Grant agency of Czech Republic under the project No. 103/09/H078 is gratefully acknowledged.

References

Chrisochoides, N. & Sukup, F. (1996) Task parallel implementation of the Bowyer-Watson algorithm, in: Proceedings of the Fifth International Conference on Numerical Grid Generation in Computational Fluid Dynamic and Related Fields: 773—782.

Cignoni, P., Montani, C., Perego, R. & Scopigno, R. (1993) Parallel 3D Delaunay triangulation, Computer Graphics Forum 12(3): 129-142.

Donea, J. & Huerta, A. (2004) Finite element methods for flow problems, J.Wiley, England.

Guibas, L.J. & Stolfi, J. (1985) Primitives for the manipulation of general subdivisions and the computation of Voronoi diagrams, ACM Transactions on Graphics, Vol.4 (2): 74-123.

Kohout, J. & Kolingerová, I. (2003). Parallel Delaunay triangulation based on circum-circle criterion. In Proceedings of the 19th Spring Conference on Computer Graphics: 73-81.

Krybus, D. & Patzák, B. (2010). On the efficient triangulation of random point sets. Computer and Experimental Analysis of Civil Engineering Materials and their Multilayered Systems. CTU in Prague.

Patzak, B. & Bittnar, Z. (2001) Design of object oriented finite element code. Advances in Engineering Software 32(10-11): 759-767.

Shewchuck, J.R. (1996) Triangle: Engineering a 2D quality mesh generator and Delaunay triangulator. Applied Computational Geometry: 203-222.