High Performance Parallel Delaunay Mesh Generation and Adaption

Franz Stimpfl, René Heinzl, Philipp Schwaha, and Siegfried Selberherr

Institute for Microelectronics, TU Wien, Gußhausstraße 27-29, Vienna, Austria

Abstract. To overcome the limitations of current meshing algorithms, a new parallel Delaunay mesh generation approach has been developed. With the steady evolution of software tools and languages, conventional programming paradigms are not sufficient anymore. Therefore, the generic programming paradigm is incorporated.

1 Introduction

Due to increasing computational power with vast performance even in desktopsystems, new possibilities for multiphysics simulations arise. However, simulations with huge meshes result in additional constraints on mesh generation, while at the same time element count and complexity of the desired structures increase. The quality of the mesh is not only critically important for the quality of the calculated results, but failure to properly control the meshing process can jeopardize or even completely prevent simulation. Since meshing is the first initial step of the simulation flow, all subsequent results depend on this fundamental step. It is of high importance to spend more effort on generating a high quality mesh and thereby reducing the executing time of subsequent modules, e.g., a linear solver. Also remeshing and adaptation steps, which have to be performed in the course of a simulation, e.g., process or device simulation in the area of Technology Computer Aided Design (TCAD), are very critical to the accuracy of the result of a simulation. The availability of high quality and robust high performance mesh generation tools is therefore of utmost importance.

We present a parallel meshing and adaptation approach combining Delaunay and advancing front algorithms suitable for finite volume and finite element discretization schemes. The introduced approach is based on a surface treatment algorithm to satisfy the constrained Delaunay property for the hull first. The subsequent task of volume meshing is performed using an advancing front algorithm, which is specially adapted to carefully, if at all necessary, insert new vertices and to comply consistent Delaunay property.

Due to the incorporation of a generic programming paradigm with external libraries, such as the parallel STL of the new GCC 4.3, can be straightforwardly integrated. Also the full utilization of upcoming many-core CPUs can be achieved, by choosing a multi paradigm programming approach.

2 Parallel Meshing Techniques

Parallel mesh generation requires a consistent interface between the segments. If a new vertex is introduced to one subdivided surface part, this vertex has to be communicated to all other surfaces which are adjacent to this one. This property can be used to categorize parallel meshing with regard to the interaction between the parallel segments - tightly coupled, partially coupled, and decoupled [1].

Due to the mathematical formulation, our approach can be classified as partially coupled. If the surface remains consistent, there is no need to exchange any data between the individual threads of the volume meshing process. This also enables the inherent utilization of many-core CPUs.

3 The Meshing Approach

A major drawback found in many Delaunay algorithms is the requirement to enclose the inital input in a convex hull from which the final mesh has to be extracted by recreating the given boundary of the modeled structure. This issue may not only result in overhead, due to the construction of convex hull parts, which can be of substantial size and also have to be meshed just to be removed at the end of the mesh generation, but also due to the numerical problems which arise from the creation of the convex hull. This issue unnecessarily complicates and slows down the whole Delaunay mesh generation process.

Our approach is based on the formal statement that a constrained Delaunay surface produces a constrained Delaunay volume mesh, where using an advancing front which does not insert new vertices.

Lemma 1. *Given a domain D containing the vertices V and the boundary B, then* $\forall\, b \in B$ *there is no vertex* $v \in V$*, which encroaches b, if b is Delaunay.*

In case an encroaching element exists, an orthogonal, azimuthal projection or a rotation of the encroaching element onto the boundary element is applied [2].

Lemma 2. *If* $b \in B$ *is Delaunay, the* $v \in V$ *closest to b, which does not encroach b due to Lemma 1, is used to create a Delaunay volume element [2].*

Lemma 3. *Let T be the set of volume elements of a tessellation of D. If* $\forall\, t \in T$ *is locally Delaunay* $\Rightarrow$ *T is globally Delauany [2].*

Our parallel meshing approach is based on this essential formal statement and the guarantee that the surface remains constant during the volume meshing procedures. It produces a Delaunay conform surface which ensures its consistency during all following meshing steps. This so prepared surface is divided into segments which are then meshed in parallel.

Basically, the advancing front algorithm does not distinguish between the actual mesh generation and adaptation steps, therefore mesh adaptation can be treated the same way as mesh generation. The well known Delaunay refinement step is another basic mechanism to improve the mesh quality. Thus, the whole Delaunay refinement mesh generation algorithm is basically a mesh adaptation with the additional property of surface incorporation.

Our parallel mesh adaptation algorithm can utilize the introduced mechanism to adapt a given mesh. Due to the constrained boundary representation and the advancing front, all subdomains can be adapted in parallel.

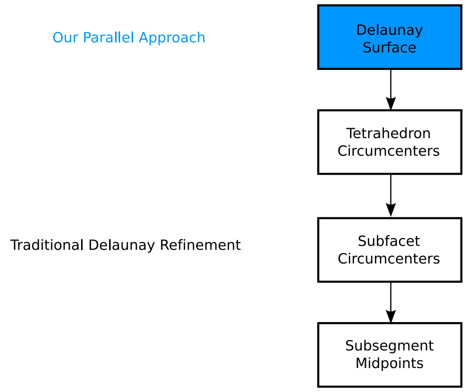

Fig. 1: Comparison of the traditional and our parallel mesh generation approach

4 Programming Paradigms

The combination of geometrical and topological issues places algorithms related to advancing front mesh generation into complex fields of programming. Problems with the robustness of geometrical algorithms for instance can yield topological inconsistencies, while topological problems, such as crossing elements, can adversely effect successful termination of the meshing algorithm. The matter of consistency is even more pronounced in a parallel environment, where consistency between the concurrent parts has to be accounted for explicitly.

To deal with these issues, we have separated the geometrical and topological areas into different programming parts. Geometrical issues are treated by using generic programming and the outsourcing of this treatment into numerical libraries, e.g., interval arithmetic or exact numerical kernels like CGAL[3] or Mauch [4]. If the geometrical predicates are correct, the element consistency is given by a simple advancing front algorithm.

This design decision permits an orthogonal optimization approach. Related to the generic approach, robustness issues can be transferred to several kernels and libraries, e.g., the CGAL numerical kernel. Our parallel meshing approach is based on the paradigms and concepts of the Generic Scientific Simulation Environment (GSSE) [5, 6].

5 Examples and Benchmarks

The presented approach is demonstrated using examples from different fields of TCAD. It can be observed that the speed of the parallel approach reduces meshing time, thus enabling the whole simulation process to quickly get a result or giving the possibility to refine the mesh more often to achieve enhanced accuracy. Execution time can be decreased with increasing segment size and complexity. The following example shows device structures which have been meshed in parallel. The various segments are colorized differently to show the partition of the mesh.

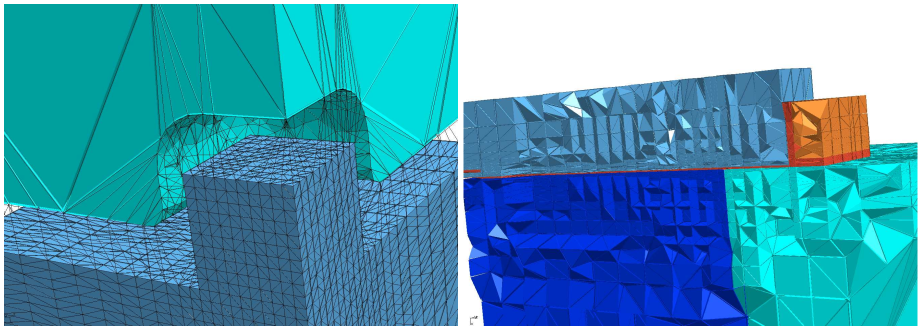

Fig. 2: Left: The mesh resulting from the extraction of an implicit surface, as used simulating moving surfaces in TCAD, Right: a structure used in the branch of TCAD dealing with device simulation.

Example	Sequential Meshing	Parallel Meshing	Num. points	Num. segments
Levelset	31sec	19sec	1.9e4	3
MOSFET	74sec	46sec	3.6e5	7

Table 1: Comparisons of the mesh generation and included mesh adaptation times (in seconds) on AMD's X2 5600.

6 Conclusion

The highly complex tasks of modeling, mesh generation, and adaption can greatly benefit from modern programming and multi-paradigm approaches. The application of modern programming paradigms and the use of a multi-paradigm development enables not only the incorporation of modern compiler technology, but also eases an orthogonal optimization approach. Because of the utilization of this programming approach the further enhancement to parallel mesh generation was reduced to a minimum.

References

1. Chrisochoides, N.: Parallel mesh generation. In: Numerical Solution of Partial Differential Equations on Parallel Computers
2. Shewchuk, J.R.: Delaunay Refinement Mesh Generation. Dissertation, School of Computer Science, Carnegie Mellon University, Pittsburgh, Pennsylvania (May 1997)
3. Fabri, A.: CGAL - The Computational Geometry Algorithm Library (2001)
4. Mauch, S.: Efficient Algorithms for Solving Static Hamilton-Jacobi Equations. Dissertation, California Institute of Technology, Purdue, CA (2003)
5. Heinzl, R., Spevak, M., Schwaha, P., Grasser, T.: A High Performance Generic Scientific Simulation Environment. In: Proc. of the PARA Conf., Umea, Sweden (June 2006) 61
6. Heinzl, R., Schwaha, P.: GSSE. (2007) http://www.gsse.at.