Grid Evolution for Oxidation Simulation Using a Quadtree Based Grid Generator

Zakir H. Sahul, Eugene W. McKenna, Robert W. Dutton
AEL 231, Stanford University, Stanford, CA 94305-4055, USA

Abstract

Grid and geometry movement algorithms for oxidation simulation are presented using the grid/geometry server *Forest*. Initial grid generation is performed using a quadtree region decimation algorithm. An oxidation solver is used to compute node velocities and the maximum time step is determined from geometry and grid considerations. The boundary nodes are then moved and geometry singularities like loop formation and region crashes are detected and removed. All the mesh nodes are moved and retriangulation is performed if necessary. The quadtree is rehashed and new triangles are allocated to conform to the new geometry. This algorithm guarantees a high grid quality at all times with minimal grid changes between time steps.

1 Introduction

Grid operations for oxidation simulation present many difficulties due to moving boundaries. In addition to grid quality and adaptation requirements, oxidation simulation requires controlled grid movement. Triangles should not diminish to zero areas, nodes should not overtake one another, and the geometry after grid movement should be valid [1] [2]. Boundaries should not self-loop and isolated regions should not move into one another. A grid movement algorithm that addresses all these issues has been developed and incorporated into *Forest*, a 2D geometry and grid server. The flowchart for the entire algorithm is shown in Figure 1.

Grid generation in *Forest* is performed using a quadtree procedure [3] by enclosing the geometry in a root square quadrilateral and recursively dividing it into a terminated lines square mesh. Boundary quadrilaterals are preprocessed by a technique called warping and final triangulation is performed using templates. Note that the geometry, terminated lines mesh and triangular mesh are stored and manipulated independently of one another for simplified grid and geometry operations.

2 Grid Movement

Node velocities are computed using SUPREM-IV's oxidation solver [2] on the triangular mesh. After an appropriate time step is determined, all the nodes in the triangular mesh are moved and a new geometry is computed. The triangular mesh is regenerated if elements

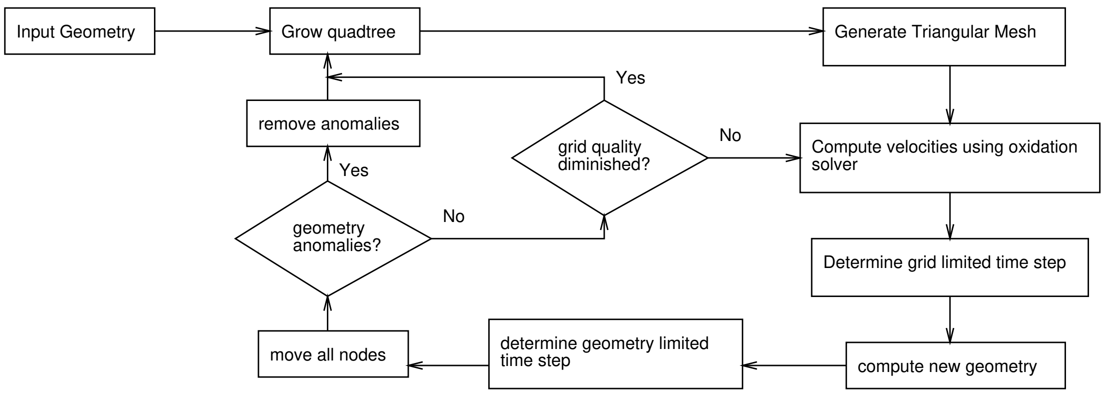

Figure 1: Flowchart for grid evolution during oxidation simulation

become highly deformed. Grid regeneration proceeds by unwarping the quadrilaterals and rehashing the quadtree to reflect the new geometry. If the geometry extends outside of the old root quadrilateral, a new root quadrilateral is formed by recursively doubling the old root's size. All of the branches in the quadtree are then visited and pruned or grown. If a branch that previously enclosed an area of the geometry now does not enclose a material area, it is pruned. Conversely, if an old branch encloses an area when it did not previously, it is grown. The new quadrilaterals are then warped and triangulated. This procedure is illustrated in Figure 2. This strategy guarantees a high quality grid at all times.

Solutions on the new nodes are interpolated from the old triangle mesh and the old triangular mesh is destroyed. The advantage of this approach is that the quadtree structure changes *only* along the moving oxide interface. The triangulation nodes in areas of no grid movement are the same - resulting in fewer interpolations between meshes. Moreover, the level of recursive division of the quadtree can be controlled to make boundary quadrilaterals or quadrilaterals in areas of high solution variation finer, allowing simultaneous grid movement and adaptation.

The maximum time step is determined using the grid and geometry criteria illustrated in Figure 3. In each case, the nodal velocity vectors are shown along with node movement with too large a time step and with the optimal time step. Node overtake conditions are detected by checking for negative or zero triangle areas after the entire triangular mesh is moved.

3 Geometric Singularities

The geometry functions of *Forest* are used to detect and remove region crashing and boundary self-looping conditions. The structure geometry is independent of the quadtree and triangular meshes and consists of boundaries with ordered and directed edges. Nodes along the boundary are first moved to determine a new geometry. Geometry anomalies are then detected. As shown in Figure 3, region crashing is detected by checking for intersections of edges of different region boundaries. Boundary self-looping is detected by checking for intersections between non-adjacent edges. The time step is adjusted using a time inter-

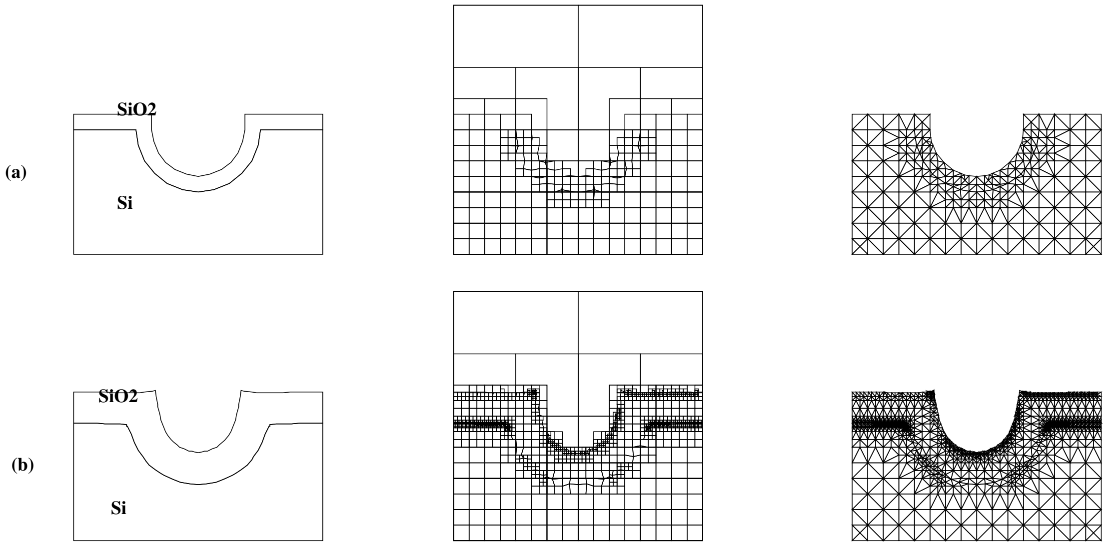

Figure 2: Use of Forest in oxidation simulation. Geometry, quadtree mesh and triangular meshes before (a) and after several oxidation steps (b).

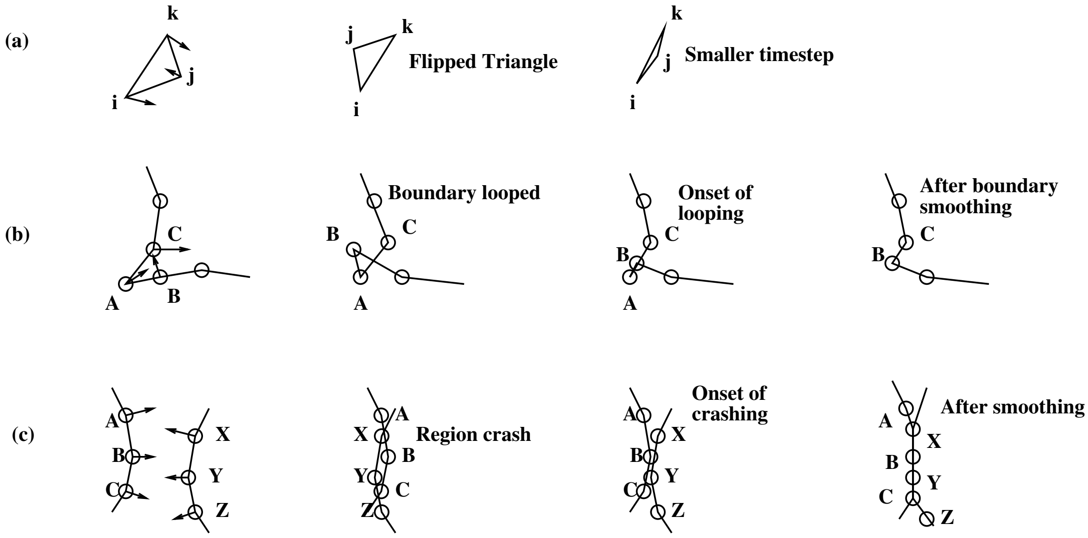

Figure 3: Grid and geometry criteria used to limit time step. (a) Nodes should not overtake one another. (b) Boundaries should not form self loops. (c) Regions should not crash into one another.

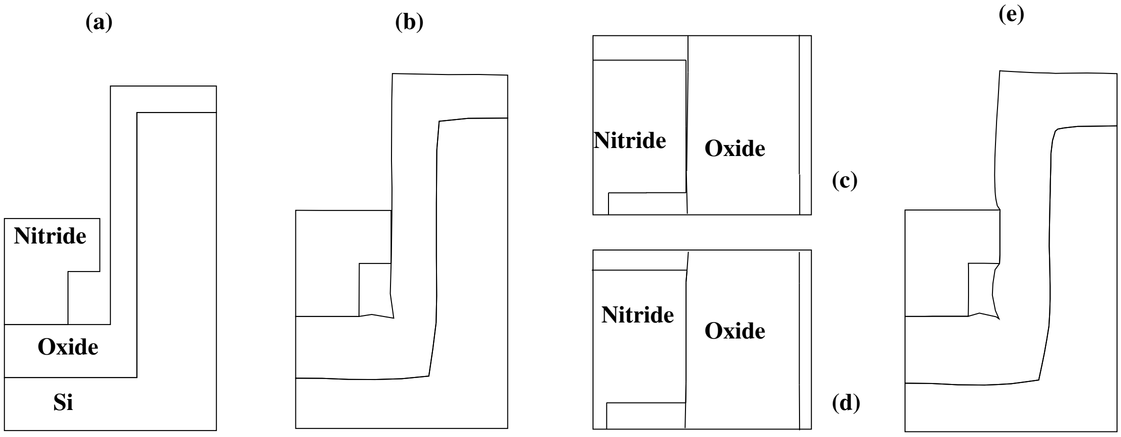

Figure 4: Oxidation simulation with colliding regions. (a) Initial geometry. (b) Geometry at the onset of crashing during oxidation. (c) Close-up view of regions before crashing. (d) Boundary smoothing to allow proper geometric coupling. (e) Subsequent oxidation

val bisection approach until the onset of an anomaly as shown in Figure 3. User-defined closeness threshold criteria are used to determine the onset of anomalies.

To continue the simulation after the onset of region crashing or boundary self-looping, the boundary is smoothed by collapsing nearby edges as illustrated in Figure 3. This removes geometry anomalies and provides proper coupling between different regions. The quadtree grid is updated to conform to the new geometry as discussed above.

Figure 4 illustrates the utility of this approach in oxidizing complicated structures. The sidewall oxide crashes into the abutting nitride region. Time steps are chosen until the onset of the crash and the geometry is smoothed as shown in Figures 4c and 4d. The grid is updated and simulation continued (Figure 4e).

Acknowledgement: This work was supported by the Semiconductor Research Corporation under contract # 92-SP-101.

References

[1] P. Sutardja. "Finite Element Methods for Process Simulation Application To Silicon Oxidation." Ph.D Thesis. Dept. of Elect. Eng. UC-Berkeley. 1988.

[2] C. Rafferty. "Stress Effects in Silicon Oxidation - Simulation and Experiments." Ph.D. Thesis. Dept. of Elect. Eng. Stanford University. 1989.

[3] Z. Sahul, R. Dutton, M. Noell. "Grid and Geometry Techniques for Multi-Layer Process Simulation". Proc. SISDEP '93. Sept. 1993.