DYNAMIC MESH ADAPTION FOR
TRIANGULAR AND TETRAHEDRAL
GRIDS

RUPAK BISWAS
RIACS-NASA AMES RESEARCH CENTER

ROGER STRAWN
US ARMY AFDD-NASA AMES RESEARCH CENTER
Requirements for Dynamic Mesh Adaption

- Anisotropic refinement capability in order to efficiently resolve directional flow features
- Coarsening required for both steady and unsteady applications
- Algorithm scaling important
- Low memory overhead using dynamic memory allocation
- CPU time comparable to a time step of the flow solver
Linked-List Data Structure

Linked List

![Linked List Diagram]

Static Array

<table>
<thead>
<tr>
<th>Item 1</th>
<th>Item 2</th>
<th>Item 3</th>
<th>...</th>
</tr>
</thead>
</table>

- Facilitates quick insertion and deletion of items
- Dynamically allocates and frees memory
- No need for compaction and garbage collection

Edge-Based Data Structure

- An edge is a line segment that connects two vertices
- A tetrahedron can be uniquely defined by its six edges: \( e_1, e_2, e_3, e_4, e_5, e_6 \)
Adaptive-Grid Data Structure

Edge List

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td>ptr*</td>
<td>array of ptrs</td>
<td>array of floats</td>
<td>int</td>
<td>ptr*</td>
<td>ptr*</td>
<td>ptr*</td>
<td>ptr*</td>
</tr>
</tbody>
</table>

- elem
- next

<table>
<thead>
<tr>
<th>face[1]</th>
<th>face[2]</th>
</tr>
</thead>
<tbody>
<tr>
<td>array of ptrs</td>
<td></td>
</tr>
</tbody>
</table>

Element List

<table>
<thead>
<tr>
<th>id</th>
<th>edge[1]:edge[2]:edge[3]:edge[4]:edge[5]:edge[6]</th>
<th>ipatt</th>
<th>fpat</th>
<th>parent</th>
<th>child</th>
<th>flag</th>
<th>next</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td>array of ptrs</td>
<td>6 bits</td>
<td>6 bits</td>
<td>ptr*</td>
<td>ptr*</td>
<td>1 bit</td>
<td>ptr*</td>
</tr>
</tbody>
</table>

Three Types of Element Subdivision

1:8

1:4

1:2

- The 1:4 and 1:2 elements are the result of anisotropic refinement or act as buffers between the 1:8 elements and the surrounding unrefined mesh
Mesh Refinement

- Individual edges marked for refinement
- Marked edges combined to form binary pattern (ipatt) for each element
- Element patterns upgraded to form valid 1:8, 1:4, or 1:2 subdivisions (fpatt)

![Diagram of mesh refinement]

<table>
<thead>
<tr>
<th>Edge #</th>
<th>ipatt</th>
<th>fpatt</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 1 0 0 1</td>
<td>9</td>
<td>11</td>
</tr>
</tbody>
</table>

Mesh Coarsening

- Elements with edges to be coarsened immediately revert back to their parents
- Parent elements have their ipatt values modified to reflect the fact that some edges have coarsened
- Parent elements then appropriately refined

![Diagram of mesh coarsening]
Additional Constraints for Coarsening

- In general, edges and elements must be coarsened in an order reversed from the one by which they were refined
- An edge can coarsen if and only if its sibling also marked for coarsening
- Edges of non-leaf elements or of their siblings cannot be coarsened

Anisotropic Error Indicator for Edges

- Adaption based on an error indicator computed for every edge of the mesh
- Flow gradients must be aligned with the edges for them to be marked for refinement
- Relative number of edges marked for coarsening and refinement adjusted to maintain a user-specified upper limit on problem size

\[ |E_e| = \| \Delta \vec{x} \cdot \Delta \vec{v} \| \]
Unstructured-Grid Euler Solver

- Basic code written by Barth; rotary-wing version developed by Strawn and Barth
- Finite-volume method with upwind differencing
- Computational control volumes centered at cell vertices
- Edge data structure allows arbitrary polyhedra
- Solution advanced in time using conventional explicit procedures
Example: Inviscid 3-D Wing
SOLUTION - ADAPTED MESH FOR A HOVERING ROTOR

Mtip = 0.90, AR = 13.7, NONLIFTING BLADE

INITIAL MESH: 5,287 POINTS, 28,841 EDGES
FINAL GRID: 27,494 NODES, 172,374 EDGES
3 REFINE LEVELS
2 COARSE LEVELS

MACH CONTOURS FOR THE ROTOR BLADE

Mtip = 0.90, AR = 13.7, NONLIFTING BLADE

INITIAL MESH: 5,287 POINTS, 28,841 EDGES
FINAL GRID: 27,494 NODES, 172,374 EDGES
3 REFINE LEVELS
2 COARSE LEVELS
Current Projects

- Mesh quality for 2-D and 3-D adaptive schemes — Goal is to guarantee that mesh quality does not degrade
- Concurrent operation of flow solver and dynamic mesh adaption on CM-5
- Error estimates/indicators for unstructured-grid solutions

Mesh Quality for Solution-Adaptive Grids

- Elements are checked for quality before they are actually subdivided

```
Bad

Good
```

- Buffer elements with large angles that may result at boundaries between different refinement levels are “corrected” before they are further subdivided

```
2 Buffer Elements

4 Refined Elements
```

- Both techniques can be used in two and three dimensions
MESH ADAPTION FOR A 2-D VISCOUS GRID

ORIGINAL GRID: 27,705 NODES, 54,725 TRIANGLES

3 REFINEMENT LEVELS, 2 COARSENING LEVELS: 73,142 NODES, 144,270 TRIANGLES

MESH ADAPTION FOR A 2-D VISCOUS GRID

CLOSE-UP OF FIRST AIRFOIL ELEMENT

ORIGINAL GRID: 27,705 NODES, 54,725 TRIANGLES

3 REFINEMENT LEVELS, 2 COARSENING LEVELS: 73,142 NODES, 144,270 TRIANGLES
Summary and Conclusions

- A new procedure has been developed for dynamic adaption of two- and three-dimensional unstructured grids
- An innovative new data structure combined with dynamic memory allocation results in fast coarsening and refinement
- Mesh quality can be "controlled" for arbitrary refinement levels
- Computed results using the solution-adaptive algorithm show excellent agreement with results for conventional structured-grid solvers