PROJECT ID5130: MESH GENERATION FOR A DOMAIN

OJAS JAIN ME17B050 AND VENU NATHAN ME17B081

Compiled May 30, 2021

CONTENTS

1. INTRODUCTION

A mesh is a discretization of a geometric domain into small simple shapes, such as triangles or quadrilaterals in two dimensions and tetrahedral or hexahedral in three. Meshes are widely used in variety of fields like numerical solutions of physical systems, cartography to get the idea of terrain etc.

A. Quadtree

A Quadtree mesh generator initially encloses the entire domain in axis aligned squares. It splits the root square recursively until the minimal node intersects the domain in simple manner. Later, a triangulation step gives a final unstructured triangular mesh.

Generation of mesh is done via Hexahedral method. OpenMP has been used extensively throughout the process.

2. METHODOLOGY

Consider a grid as shown in figure 1. If the size of the cell is larger than the required mesh size in its domain and the maximum division limit has not been reached, then it is split into 4 smaller square cells in figure 2.
The older cell is not deleted and is marked as parent of the new four cells. Consider that bottom right cell is further split in 4 subparts as shown in figure 3.

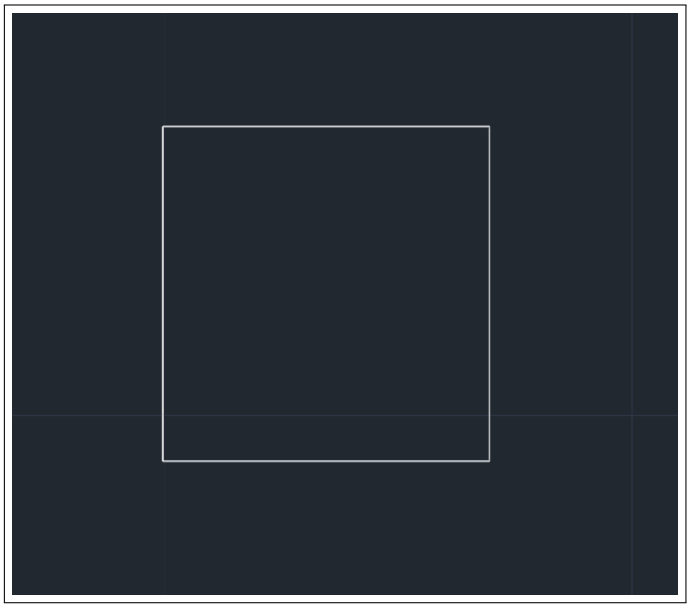

Fig. 1. Consider a square cell

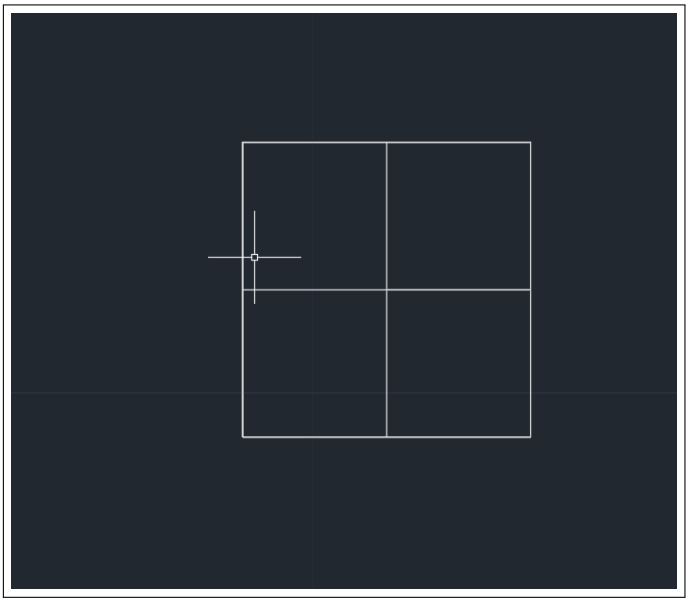

Fig. 2. The cell is further partitioned as it's size is greater than we expected

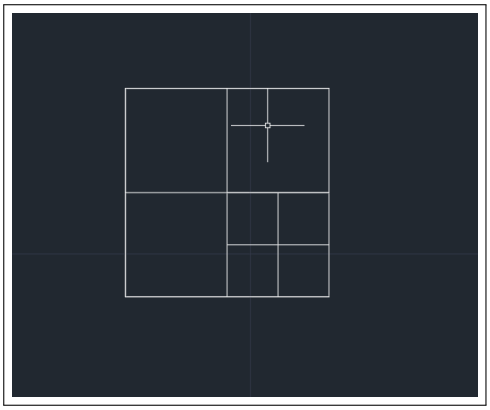

Fig. 3. The cell in bottom right is further split according to requirement

Consider figure 4, each cell has information about its neighbouring cells. The neighbour of a cell is the smallest other cell which shares the complete boundary with a cell.

- Right neighbour of C7 is C8.
- Bottom neighbour of C2 is C4, despite C4 being a parent of 4 cells.
- Top neighbour of C5 is C2.

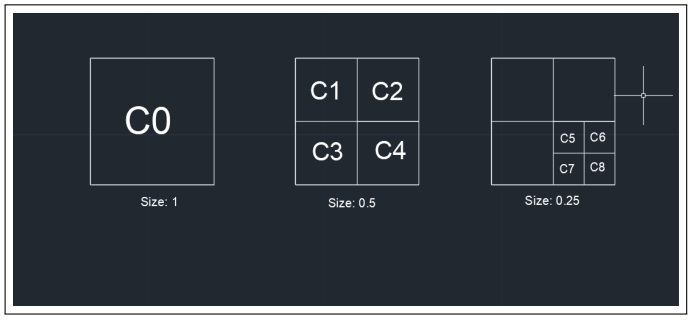

Fig. 4. Parent and Children cell

In figure 5, the process of dividing the cells can be easily **parallelized**. After the cells have been sufficiently divided, the cells which are not parents attempt to give unique numbering to its corners. Each of the cell will have an information about it's neighbouring parent cells. For this purpose, parallelization has been used. Multiple cells will be passed to a thread in OpenMP. This numbering does not need to form a sequence, so the cells can number them however they want.

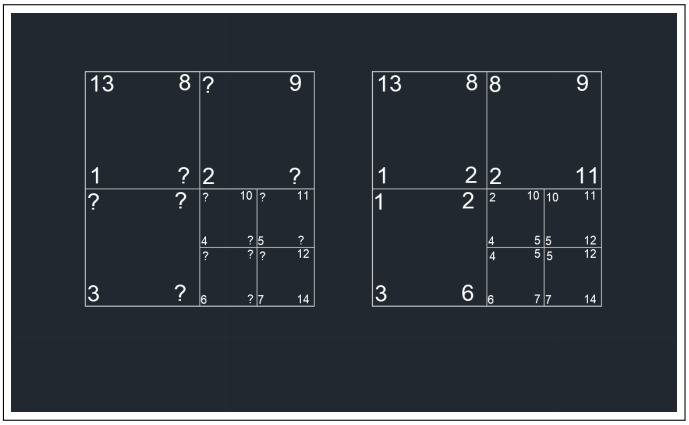

Fig. 5. Consistent numbering across all cells

- A cell always numbers its bottom left corner.
- A cell numbers its top right corner if its not the bottom left corner of some other cell.
- A cell numbers top left and bottom right corners if not the bottom left or top right corner of some other cell.

Cells use the information about neighbours to find which corners to number. After all cells have finished numbering corners, using information about neighbours, threads will collect the numbering of the corners which they themselves did not number. As in figure 6,for each cell, there will be four variables storing the unique id of all four vertices. It is important that for a given vertex, all the threads processing corresponding cells have that unique id stored. Each thread will also maintain an array of co-ordinates and the unique id given to each vertex.

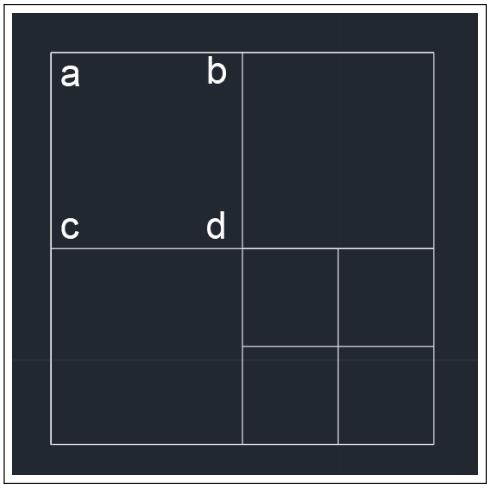

Fig. 6. a,b,c,d are the four variables storing the value of the corners

For any algorithm to work efficiently on the mesh, it is important that all the primary cells have same number of vertices. Hence, we will choose triangle as our primary cell.

Consider figure 7. Now, the cells start forming triangles. Each cell marks a new vertex at their center, then they join segments of their edges to the center to form triangles. For cells which have neighbours on a side (like C2 on the bottom side), they iterate

over the children of the neighbour in binary tree like fashion to obtain the numbering of the vertices forming the segments on their edge (C2 iterates over the top-left and top-right child of C4).

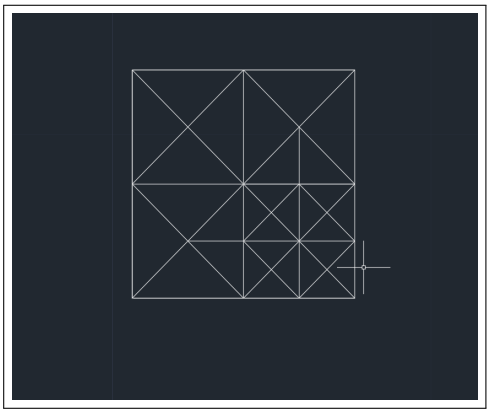

Fig. 7. Formations of triangles to keep number of vertex per cell uniform

3. TIME PLOTS

There are two different times of time mentioned in the tables:

1. C Time: Time required for computation of cells, vertices and triangles
2. E Time: Total execution time, including file I/O

Depth in this context refers to the number of times the base unit square cell was divided. Max Depth is a parameter controlled by the user, which decides the depth at which the program should ignore mesh size requirement and stop refining.
Increasing Max Depth produces finer meshes.

Threads	C Time (μs)	E Time (μs)
1	15283	43901
2	8965	29101
4	5290	31110
8	3532	36472

Table 1. Time for Max Depth: 8

Threads	C Time (μs)	E Time (μs)
1	272698	591341
2	159672	461543
4	91621	430337
8	62498	373718

Table 2. Time for Max Depth: 12

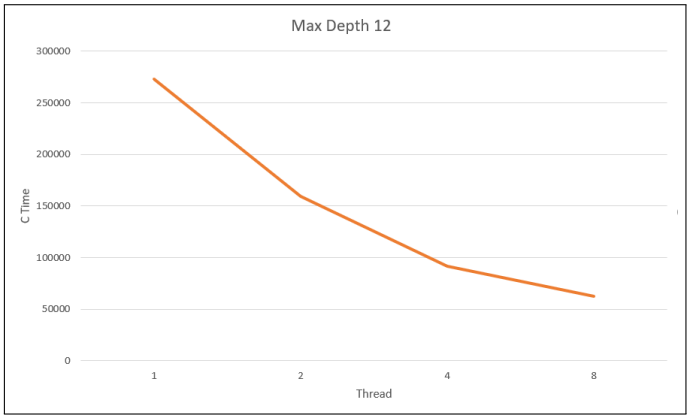

Fig. 8. C Time: Max Depth 12

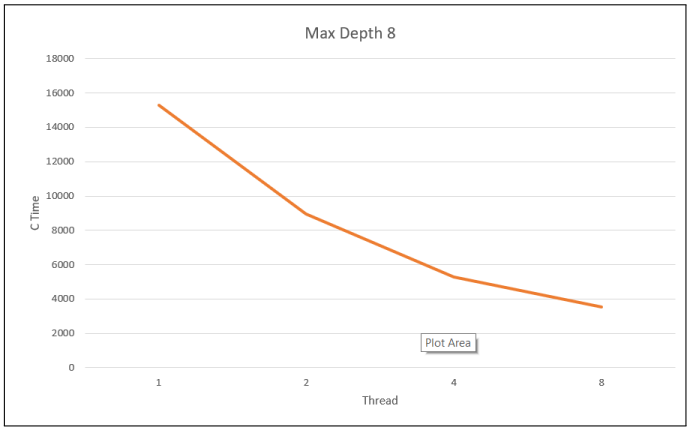

Fig. 9. C Time: Max Depth 8

Threads	Max Depth = 8	Max Depth = 12
1	1.0000	1.0000
2	1.7047	1.7079
4	2.8890	2.9764
8	4.3270	4.3633

Table 3. Speedup Table

4. SAMPLE MESHES : PLOTS

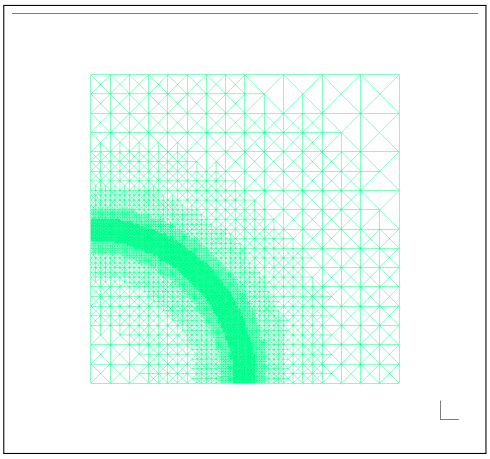

Fig. 10. Example Mesh 1

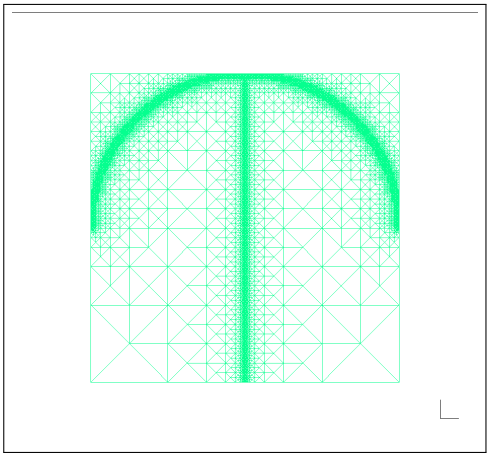

Fig. 11. Example Mesh 2

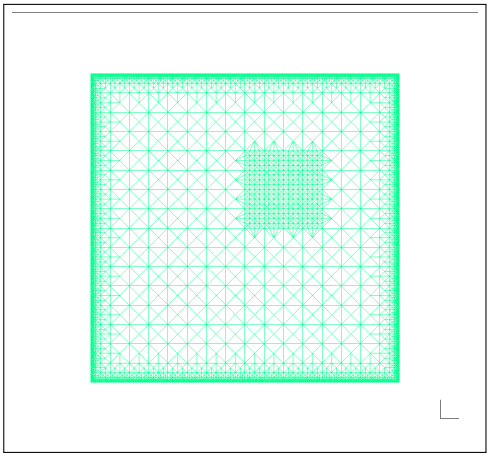

Fig. 12. Example Mesh 3

5. SAMPLE MESHES

For the purpose of demonstrating the types of complex meshes which can be generated by our methods, we have shown three samples. These can be seen in figure 10, 11, and 12.
The instructions for generating these meshes will be included in the zip file submitted along with this report.

6. CONCLUSION

As we can see from figure 8 and figure 9, the major computation of the algorithm, such as division of cells, computation of vertex numbering and triangles, parallelize well. But as a mesh generation algorithm, file I/O is an important part of exporting meshes. This time taken for file I/O is not parallellizable and takes a large amount of time compared to C Time. Hence we conclude that it is advisable to integrate advanced meshing features into solvers, rather than using a separate mesh generation program to produce and export the meshes.

7. FURTHER SCOPE OF IMPROVEMENT

The feature allowing marking of regions outside the domain could not be completed, due to time constraints. We felt it would be better to focus on the consistency of the mesh, due to its importance. We did not implement the code for 3D, but this process of dividing cells, marking vertices and finding triangles can be easily extended to 3D.

8. REFERENCES

1. Research Paper