Triangle Graphs*

Messaoud Benantar[1], Uğur Doğrusöz[2]
Joseph E. Flaherty[2], and Mukkai S. Krishnamoorthy[2]

[1] Large Scale Computing Division, IBM Corporation,
MS 105, Neighborhood Road, Kingston, NY 12401
[2] Department of Computer Science,
Rensselaer Polytechnic Institute, Troy, NY 12180

April 10, 1995

Abstract

We introduce a new class of planar graphs called triangle graphs. First, we present a formal way of constructing and characterizing triangle graphs, and then show that they characterize the adjacencies of arbitrary triangulations and they are three-colorable for a certain subclass of triangulations. Subsequently, we discuss an application of triangle graphs to the parallel finite element solution of elliptic partial differential equations on triangulated domains.

1. Introduction

The recent growth and availability of parallel computers has spurred a corresponding interest in the development of parallel procedures for the computational solution of large-scale scientific problems. The popular finite element method [12,13] for solving partial differential equations, for example, generally is a computationally intensive task that involves, among other things, the solution of a large sparse linear algebraic system. Although irregular structures abound, we show in this study that effective parallel solution strategies can be obtained for two-dimensional elliptic and parabolic systems by introducing a new planar graph called a *triangle graph* to suggest a decomposition of the problem domain into independent segments that may be processed in parallel by cooperating units.

Triangular, quadrilateral or mixed triangular and quadrilateral meshes may be constructed on an arbitrarily complex two-dimensional problem domain by initially embedding it in a square universe and recursively bisecting those edges of squares that intersect

*This research was partially supported in part by the National Science Foundation under Grant Number CDA-8805910, by the U.S. Air Force Office of Scientific Research, Air Force Systems Command, USAF, under Grant Number AFOSR F49620-93-1-0218, and by the U.S. Army Research Office under Grant Number DAAL-03-91-G-0215.

the domain's boundary [7]. This process leads to a *quadtree* dissection of the domain with nodes of the tree representing square regions called *quadrants*. Quadrants at terminal tree nodes may be further divided or distorted to produce the triangular, quadrilateral or mixed mesh. This procedure extends to three-dimensional domains using an octree decomposition of the domain [14]. Benantar et al. [11] developed a "coloring" procedure that separated contiguous terminal quadrants in order to process quadrants having the same color in parallel without conflict on shared-memory computers. Algorithms using a maximum of six and eight colors were described with the six-color procedure providing superior performance [11] due to the finer data granularity. When used with finite element computations, data associated with all elements in a quadrant was generated and assembled into the global algebraic system, which was solved in parallel by colors using a preconditioned conjugate gradient method [11].

Performance of the six-color procedure, as measured by parallel speed-up, was good when piecewise linear finite element approximations were used [11]; however, performance degraded with increasing polynomial degree. A finite element approximation typically converges to the exact solution of the partial differential system as either the number of elements or the degree of the polynomial tends to infinity. Recent experience suggests that the latter approach is the more efficient [15] and that techniques that simultaneously refine elements of the mesh while increasing the degree of the polynomials often have exponential convergence rates, even with singular solutions [16].

In summary, as the degree of the piecewise polynomial approximation increases, (i) serial efficiency increases, (ii) parallel performance of quadrant-coloring procedures decrease and (iii) the number of edge and face nodes increases; hence, it is natural to consider algorithms that separate contiguous element edges or faces by color. Concentrating in this study on two-dimensional meshes of triangular elements, we introduce and characterize a new planar graph (cf. Section 2) called a *triangle graph*, prove that a restricted type of triangle graphs is three-colorable (cf. Section 3) and describe an algorithm to perform the coloring. Using this algorithm, independent sets of triangular element edges may be processed in parallel with a minimum number of colors; thus, the procedure has the finest data granularity. An example (cf. Section 4) indicates the superior performance of the edge-coloring procedure relative to quadrant coloring [11]. We close with some concluding remarks (cf. Section 5).

2. Triangle Graphs

We introduce some basic material relevant to triangle graphs.

Definition 1: *Merging* two nodes v and w, consists of collapsing them to form a new node at the same position as v which is adjacent to all nodes x that were adjacent to either v or w.

Definition 2: An *element* is a complete graph of three nodes with an embedding (cf. Figure 1a).

Definition 3: A *component* is either a single element or a chain of elements where two and only two elements called *end elements* have two degree-two nodes. The remaining elements have two degree-four and one degree-two nodes (cf. Figure 2a). The smallest component is a single element.

Definition 4: A *triangle graph* is an embedded planar graph G constructed according to the following rules:

i. A component (basic building block) is a triangle graph.

ii. If G is a triangle graph and H is a component, and x and y are degree-two nodes of G and H, respectively, then the graph obtained by merging nodes x and y is a triangle graph, if and only if the resulting graph is planar (cf. Figure 1b).

iii. If G is a triangle graph and H is a component, and x_1 and x_2 are two neighboring degree-two nodes of G, and y_1 and y_2 are two neighboring (cf. Definition 5) degree-two nodes of H, then the graph obtained by merging nodes x_1 and y_1, and nodes x_2 and y_2 is a triangle graph if and only if the resulting graph is planar, no node of an interior face is of degree two, every interior face contains at least three edges, and the resulting graph has at least three nodes of degree two (cf. Figure 1c).

Definition 5: Since triangle graphs are embedded in a plane, the exterior face consists of all nodes in the exterior region. Two degree-two nodes n_1 and n_2 of a triangle graph are *neighbors* if and only if there is a path along the exterior face from n_1 to n_2 that does not contain any degree-two nodes. For example, in the graph of Figure 3, nodes n_1 and n_2 are neighbors while n_1 and n_3 are not.

Remark: The triangle graph in Figure 1c has three elements, not four as apparent from Definition 2. Thus, the graph in Figure 1c can be constructed from a component of a single element by adding a second component of a single element using Rule *ii* and a third one using Rule *iii*.

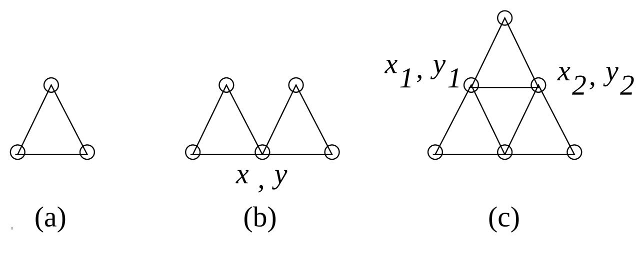


Figure 1: An element of a triangle graph (a) and the construction of other triangle graphs by merging one (b) and two (c) degree-two node(s).

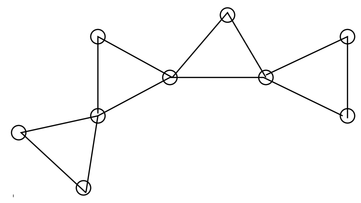


Figure 2: An example of a component of a triangle graph.

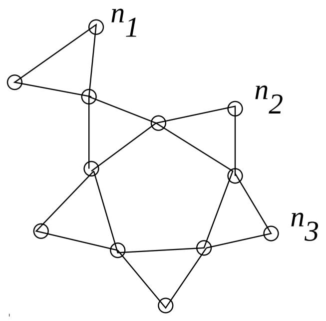


Figure 3: Neighboring nodes n_1 and n_2 and non-neighboring nodes n_1 and n_3 of a triangle graph.

3. Coloring Triangle Graphs

Having characterized triangle graphs, we now show that triangle graphs with certain properties can be colored with three colors. Prior to that, let us first derive some properties of triangle graphs.

Definition 6: A path from u to v is said to be a *maximal alternating path*, or simply an *alternating path*, of two colors, if it cannot be extended and the nodes defining it alternate between two colors.

Lemma 1: In a three-colored triangle graph, an alternating path of two colors starting from a node of degree two can only be a simple path ending at a node of degree two.

Proof: We proceed by contradiction. Assume there is an alternating path of two colors emanating from a node of degree two that is not a simple path. Then, there is at least one node x that is visited more than once and it must be of degree four. The four neighbors of x must have the same color. However, this implies the existence of an element having only two colors, which is impossible for a three-colorable graph. Therefore, the alternating path is simple. Moreover, an alternating path terminating at a degree-four node cannot be maximal because it is extendable there. Hence, a maximal path must end at a node of degree two. ■

Lemma 2: Alternating paths of the same two colors, say $r - g$, do not intersect in a three-colored triangle graph.

Proof: The arguments follow those of Lemma 1. ■

Lemma 3: In a three-colored triangle graph, an alternating path of two colors starting from a degree-four node has to return to the starting node forming a cycle.

Proof: According to Lemmas 1 and 2, an alternating two-color path starting at a node of degree two node cannot contain a cycle at a degree-four node and must end at a degree-two node. Thus, any alternating two-color path starting from a node of degree four must contain only degree-four nodes and for the path to be maximal, it must be a cycle. ■

Lemma 4: Interchanging colors along a maximal alternating two-color path in a three-colored triangle graph preserves the three-coloring of the graph.

Remark: *Interchanging* refers to the interchange of successive pairs of colors along an alternating path.

Proof: A maximal alternating path of two colors, say r and g, in a three-colored triangle graph is connected to the rest of the graph at nodes having color b. Hence, interchanging colors along the path does not affect the remaining graph and, therefore, three-coloring is preserved. ■

Definition 7: Consider the class of triangle graphs for which it is possible to decompose the graph into components such that the composition of the components back together uses only Rule *iii* for components with more than one triangle element. We will call these graphs *restricted triangle graphs.*

With these preliminaries established, we prove the following.

Theorem 1: Restricted triangle graphs are three-colorable. Furthermore, any two neighboring degree-two nodes of these graphs can be colored distinctly.

Proof by Induction: The necessity of three colors is trivially established from the basic building block of a triangle graph which is a single element. We prove sufficiency using an induction argument on the number of components of the graph.

Base: As the basis, we take a triangle graph consisting of a single component. A component can be three colored by using two alternating colors, say r and b, at degree-four nodes and the third color g at nodes of degree two (cf. the left portion of Figure 4).

Coloring two neighboring degree-two nodes distinctly utilizes the interchanging technique of Lemma 4. Consider a component having neighboring degree-two nodes, n_1 and n_2, of the same color (cf. the left portion of Figure 4). Suppose they are connected by an alternating path of colors g and r, then by interchanging colors along a $g - b$ path starting at either n_1 or n_2, the two nodes will have distinct colors g and b. (cf. the right portion of Figure 4).

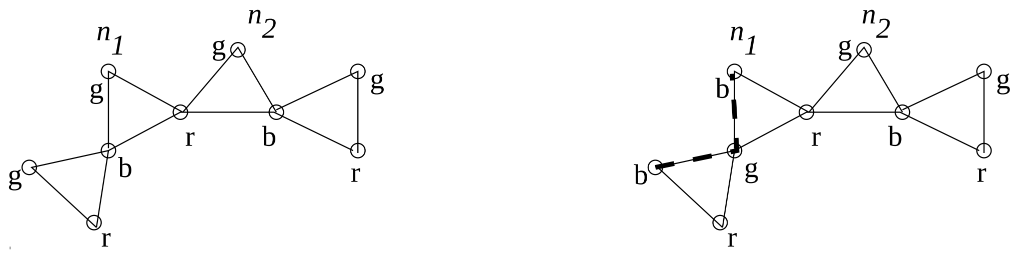


Figure 4: Coloring neighboring nodes distinctly in a component of a triangle graph. The path along which colors are interchanged is marked with dashed lines.

Induction Hypothesis: Assume all restricted triangle graphs of $n-1$ components are three-colorable and any of their neighboring degree-two nodes can be colored distinctly. Now, consider the addition of an nth component.

1. We first consider the addition of a component using Rule *ii.* In this case, a three-coloring and the distinct coloring of neighboring nodes of the new component follow the arguments used for the base case. Now, suppose that neighboring nodes n_1 of

the new component and n_2 of the existing $(n-1)$-component graph have the same color (cf. Figure 5a). From Definition 4, there is only one node n_3 where the new component connects to the existing $n-1$ components. Since nodes n_1, n_2 and n_3 are colored using at most two colors, there is an alternating path of two colors starting from n_2 that uses a color different than the color of n_3 and does not lead to n_1. We use this path to interchange the colors, and, thus, change the color of n_2.

2. When the addition of an n-th component is done using Rule *iii*, a three-coloring and distinct coloring of neighboring nodes of the same component follow arguments similar to those used for adding a new component using Rule *ii*. When neighboring nodes n_1 and n_2 are of two different components and they have the same color, we have two cases depending on whether or not n_3 and n_4 have the same color. Let n_3 and n_4 be the nodes where the new component is added to the $(n-1)$-component graph and further let n_3 be adjacent to n_1 (cf. Figure 5b).

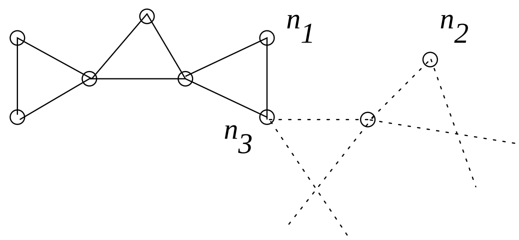
   (a) If n_3 and n_4 have the same color, then nodes n_2, n_3 and n_4 have at most two colors. Therefore, interchanging colors along the alternating path from n_2 using a color different than that of n_3 and n_4 (in order to make sure n_3 is not on this alternating path and does not end up having the same color as n_1) will change the color of n_2.

   (b) If n_3 and n_4 have distinct colors, say r and b, respectively, then assume, without loss of generality, that nodes n_1 and n_2 have color g (cf. Lemma 4). If the alternating path p_1 of colors g and r starting from node n_2 does not reach node n_1, then interchanging colors along p_1 will change the color of n_2. A similar procedure can be used if the path p_2 of colors g and b starting from n_2 does not reach n_1. If both p_1 and p_2 reach node n_1, then the degree-two node n_5 in the n-th component that is adjacent to node n_4 must have color r (since node n_6, which is adjacent to both n_4 and n_5, must be on path p_2 and have color g). Since path p_1 and a $g-r$ path p_3 starting at node n_5 do not intersect, interchanging colors g and r along p_3 will not change the colors of n_1 and n_2. However, path p_2 now ends at node n_5. Hence, interchanging colors along p_2 will change the color of n_2.

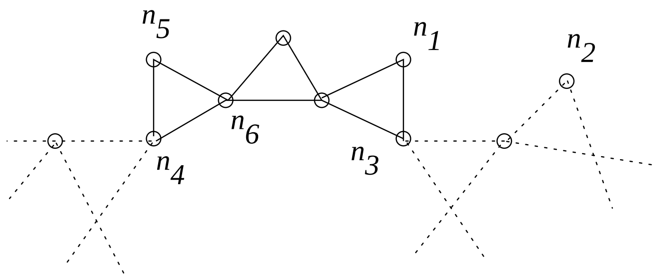
Corollary 1: All degree-two nodes of a three-colored restricted triangle graph having exactly three degree-two nodes, must have distinct colors.

Proof: Let's denote the three degree-two nodes of a three-colored restricted triangle graph by n_1, n_2 and n_3. Without loss of generality, suppose that n_1 and n_2 have color r, and n_3 has color b. By Lemma 1, the path of colors $b-g$ starting at n_3 should end at either n_1 or n_2; hence, a contradiction. ■

Following Theorem 1, we devise a recursive procedure to three-color restricted triangle graphs. We proceed by coloring one component at a time starting from a node n_1 of degree four. The nodes n_1, n_2 and n_3 of an element el_1 are colored and n_2 is enqueued to recursively spread the process to neighboring components. The coloring procedure is shown in Figure 6.



(a)



(b)

Figure 5: Distinct coloring of neighboring nodes n_1 and n_2 across components of a triangle graph when n_1 belongs to a component added using Rule *ii* (a) and when n_1 belongs to a component added using Rule *iii* (b).

```
procedure color_triangle_graph(n1:node)
begin
  retrieve element el1 containing node n1;
  if (el1 not colored) then
    if (n3 not colored) then
      color n2 and n3 distinct from n1;
      label el1 colored;
      enqueue(n2);
      color_triangle_graph(n3);
    else if (color of n1 ≠ color of n3) then
          color n2 distinct from n1 and n3;
          label el1 colored;
          enqueue(n2);
          color_triangle_graph(n3);
        else interchange the color of n3 with a color distinct from that of n1;
          repeat this color interchanging until reaching a degree two node;
          color n2 with a color distinct from n1 and n3;
          label el1 colored;
          enqueue(n2);
          color_triangle_graph(n3);
        end if
    end if
  else if (queue not empty) then
      color_triangle_graph(dequeue(queue))
      else return;
      end if
  end if
end color_triangle_graph
```

Figure 6: Algorithm for three-coloring of triangle graphs.

4. An Application of Triangle Graphs

We consider an application of triangle graphs to the parallel solution of two-dimensional finite element problems on grids of triangular elements. Parallel techniques for finite element problems frequently employ decomposition methods [1,3,4] where elements and nodal data are separated into independent noncontiguous regions on which computational tasks are scheduled in parallel. The performance of these techniques is generally based on the granularity of the decomposition with finer decompositions being more likely to produce higher parallel performance.

We show that grids of triangular elements can be represented using triangle graphs and that the three-coloring algorithm (cf. Section 3) can be used to separate nodal data that is associated with finite element edges of triangular grids.

Definition 8: Two edges of a triangular grid are *adjacent* if and only if they belong to the same finite element.

Definition 9: The *edge adjacency graph of a triangular grid* is a graph $G_2 = (V, E)$ where every edge e of the grid is mapped onto a vertex n of G_2 and two nodes of G_2 are adjacent if and only if their corresponding grid edges in the grid are adjacent.

Remark: This notion of edge adjacency differs from its normal definition for line graphs [8].

Theorem 2: The edge adjacency graph of an arbitrary triangular grid is a triangle graph.

Proof: We use induction on the number of the triangular elements of the grid. A single triangular finite element maps onto a single element of its triangle graph representation. A grid of two triangular finite elements maps onto a triangle graph consisting of a component of two elements. Two grids of three triangular finite elements corresponding to two triangle graphs are shown in Figures 7a and b respectively. A slightly more complex triangular grid with its corresponding triangle graph superimposed is shown in Figure 8.

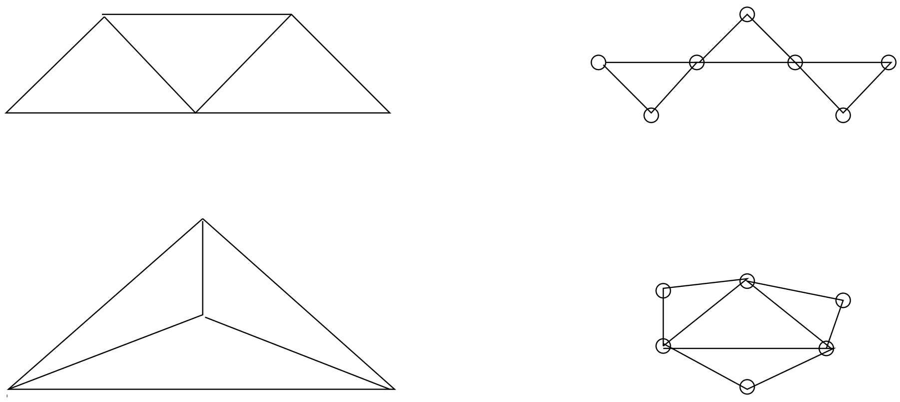


Figure 7: Two grids of three triangular finite elements (left) and their corresponding triangle graphs (right).

Assuming the induction hypothesis is true for triangular grids of $n-1$ elements, consider an arbitrary grid of n triangular finite elements. Removing a boundary element results in a grid having $n-1$ finite elements which can be mapped onto a triangle graph. Returning the removed element to the triangular grid results in two situations where (i) the triangular element adds two edges to the grid to extend the corresponding triangle graph according to Rule *ii* of Definition 4, or (ii) the new element adds one edge to close a triangle graph according to Rule *iii* of Definition 4. Both cases result in a triangle graph since they are built using the rules of Definition 4. ■

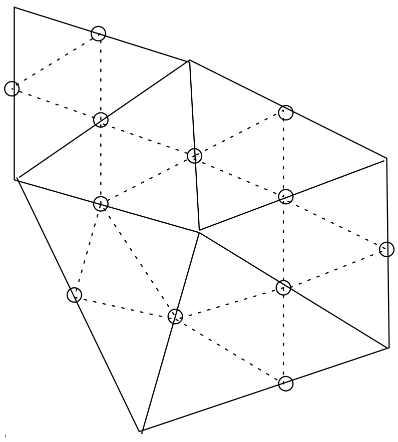


Figure 8: A grid of triangular finite elements (solid lines) corresponding to triangle graph (dotted lines) representing adjacency.

As noted (cf. Section 1), our interest in triangle graphs was motivated by a need to construct parallel techniques for high-order finite element approximations which have the largest portion of their solution nodes distributed along element edges. Any interior nodes on elements can be associated with particular edges. Each vertex unknown is associated with one of the edges incident upon the vertex. This process is equivalent to coloring finite element vertices so that no two vertices incident upon the same edge have the same color. Since this generally requires four colors, two vertices can be associated with the same edge when a conflict occurs. Therefore, parallel computations can be scheduled over element edges rather than on subregions as customary with domain decomposition methods [1-3]. Elemental computations and assembly of the global linear system can be done in parallel by traversal of one set of edges having the same color, say r, since this set spans the entire triangular grid. The resulting linear algebraic system is solved using a preconditioned conjugate gradient method with a symmetric successive over relaxation (SSOR) preconditioning [11]. SSOR iterations are performed in parallel by sweeping the edges corresponding to each color [5,6]. Process synchronization is only needed at the end of each queue of colors; thus, any load imbalance is extremely minor. The minimal (three) colors ensure a fine granularity that improves with increasing polynomial degree [6].

Example: Consider the Dirichlet problem

$$u_{xx} + u_{yy} = f(x,y), \quad (x,y) \in \Omega, \tag{1}$$

$$u = 0, \quad (x,y) \in \partial\Omega, \tag{2}$$

with $\Omega = \{(x,y)| -1 \le x,y \le 1\}$ and the function $f(x,y)$ defined so the exact solution is

$$u(x,y) = \sin 3\pi x \ sin 3\pi y. \tag{3}$$

We solved this problem on a grid of 238 elements with 1989 nodes using piecewise polynomial approximations of degree four. Parallel speedups,

$$S_p = \frac{execution\ time\ using\ one\ processor}{execution\ time\ using\ p\ processors}, \tag{4}$$

reported in Figure 9, correspond to a solution obtained on a 16-processor Sequent Symmetry S-81 shared-memory multiprocessor. The performance of the solution using edge coloring is compared to one using quadrant coloring (cf. Section 1, [5,6,11]) to separate the contiguous regions. Speedups using edge coloring of the triangular grid are clearly superior to those using quadrant processing. With 15 processors, the speedup using edge processing is 87% of ideal while the quadrant processing result is only 66% of ideal.

5. Conclusion

We presented a new class of graphs, called triangle graphs, which arise from the discretization of partial differential equations on two-dimensional domains of triangular elements. We prove that three colors are necessary and sufficient to color the vertices of restricted triangle graphs and we discuss an application of this coloring to the parallel high-order finite element solution of partial differential equations.

It is also natural to ask whether or not the coloring concepts and procedures developed herein extend to three dimensions. One immediate extension that comes to mind is to consider *tetrahedra graphs* describing the facial adjacency of grids of tetrahedral finite elements. However, an alternate possibility would separate contiguous edges of tetrahedral finite elements. Graphs based on the latest approach would have a more complex structure than the former, but recent results indicate computational advantages of three-dimensional edge-based procedures [17].

References

[1] C. Farhat and F. Roux, "An Unconventional Domain Decomposition Method for Efficient Parallel Solution of Large-Scale Finite Element Systems," SIAM J. Sci. Stat. Comput., Vol. 13, pp. 379-396, 1992.

[2] S. Hutchinson, E. Hensel and S. Castillo, "The Finite Element Solution of Elliptical Systems on a Data Parallel Computer," Int. J. Num. Meth. Engng., Vol. 32, pp. 347-362, 1991.

[3] G. Yagawa, N. Soneda and S. Yoshimura, "A Large Scale Finite Element Analysis Using Domain Decomposition Method on a Parallel Computer," Computers and Structures, Vol. 38, pp. 615-625, 1991.

Figure 9: Parallel speedups for edge and quadrant-processing procedures when solving the Dirichlet Problem (1,2).

[4] S. Succi and M. Benassi, "A Four-Color Parallel Algorithm for the Solution of a Two-Dimensional Advection-Diffusion Equation with the Finite Element Method," J. Sci. Comput., Vol. 4, pp. 61-70, 1989.

[5] M. Benantar, Parallel and Adaptive Algorithms for Elliptic Partial Differential Systems, Ph.D. Dissertation, Department of Computer Science, Rensselaer Polytechnic Institute, Troy, 1992.

[6] M. Benantar, J. E. Flaherty and M. S. Krishnamoorthy, "Coloring Procedures for Finite Element Computation on Shared-Memory Parallel Computers," AMD-Vol. 157, Adaptive, Multilevel, and Hierarchical Computational Strategies, A. K. Noor Ed., ASME, New York, pp. 435-449, 1992.

[7] P. L. Baehmann, S. L. Wittchen, M. S. Shephard, K. R. Grice and M. A. Yerry, "Robust Geometrically Based, Automatic Two-Dimensional Mesh Generation," Int. J. Num. Meths. Engng., Vol. 24, pp. 1043-1078, 1987.

[8] F. Harary, Graph Theory, Addison Wesley, Reading, 1969.

[9] T. Nishizeki and N. Chiba, Planar Graphs: Theory and Algorithms, Elsevier Science Publishers B. V., North-Holland, 1988.

[10] H. Whitney, "Planar Graphs," Fund. Math. ,Vol. 20, pp. 73-84, 1933.

[11] M. Benantar, R. Biswas, J.E. Flaherty and M.S. Shephard, "Parallel Computation with Adaptive Methods for Elliptic and Hyperbolic Systems," Comput. Meths. Appl. Mech. Engrg., Vol. 82, pp. 73-93, 1990.

[12] E. B. Becker, G. F. Carey, and J. T. Oden, Finite Elements: An Introduction, Prentice-Hall, Englewood Cliffs, 1981.

[13] B. Szabo and I. Babuska, Finite Element Analysis, John Wiley and Sons, New York, 1991.

[14] M. S. Shephard and M. K. Georges, "Automatic Three-Dimensional Mesh Generationby the Finite Octree Technique," Int. J. Num. Meth. Engng., Vol. 32, pp. 709-739, 1991.

[15] S. Adjerid, J. E. Flaherty, P. K. Moore, and Y. J. Wang, "High-Order Adaptive Methods for Parabolic Systems," Physica D, Vol. 60, pp. 94-111, 1992.

[16] E. Rank and I. Babuska, "An Expert System for the Optimal Mesh Design in the hp-Version of the Finite Element Method," Int. J. Num. Meth. Engng., Vol. 24, pp. 2087-2106, 1987.

[17] R. Biswas and R. Strawn, "A New Procedure for Dynamic Adaption of Three-Dimensional Unstructured Grids, Paper AIAA-93-0672, 31th Aerosciences Meeting and Exhibit, Reno, 1993.