Quadtrees e Aplicações à Geração de Malhas

Pontifícia Universidade Católica do Rio de Janeiro

IMPA - Instituto de Matemática Pura e Aplicada

Departamento de Engenharia Civil

Disciplina: **Geometria Computacional**

Professor: **Paulo Cézar Carvalho**

Antonio Carlos de Oliveira Miranda (amiranda@tecgraf.puc-rio.br)

Luciano Falcão da Silva(falcao@openlink.com.br)

1. INTRODUÇÃO

O método do estudo de problemas em engenharia tem se desenvolvido nas últimas décadas aliado ao maior acesso aos recursos computacionais. Houve tempo em que determinada estrutura, necessitava da construção de um protótipo para ser estudada, ou então eram utilizados fatores de ponderação que levavam a um custo desnecessário.

As técnicas de computação, quando aplicadas a problemas de engenharia, proporcionam uma análise muito mais próxima do comportamento real de modelos, pois possibilitam o uso de processos matemáticos complexos, que para o ser humano seria impossível de se fazer.

Nestes processos matemáticos, a técnica de elementos finitos se destaca como uma solução prática e de fácil implementação, devido à simplicidade e repetição de seus passos de solução.

O aprimoramento destas técnicas leva a estudos de eficiência e de velocidade na geração de malhas. Vários algoritmos vêm sendo desenvolvidos com a finalidade de tornar os programas de processamento de elementos finitos ao mesmo tempo rápidos e cada vez mais confiáveis. Estudos têm sido feitos no sentido da utilização de etapas prévias à geração de malhas. Uma delas é a representação da estrutura em uma quadtree.

Neste trabalho são apresentados os conceitos de malhas de elementos finitos e da representação de uma estrutura em uma quadtree. Alguns algoritmos utilizando estes recursos são mostrados e uma análise comparativa da qualidade da malha gerada em cada um deles, assim como da velocidade de geração, é feita. Também é feita uma análise da complexidade de cada um destes algoritmos, em cada uma de suas etapas.

2. QUADTREE

Quadtree é uma forma de representação bidimensional, criada no final dos anos 60. A idéia fundamental é a capacidade de aproveitamento do conceito de dividir para conquistar de uma subdivisão binária. Uma quadtree é criada pela subdivisão sucessiva de um plano bidimensional, para que sejam formados quadrantes, como mostrado na Figura 1. Quando uma quadtree é utilizada para representar uma área no plano, cada quadrante pode estar inteiramente contido nesta área, parcialmente contido, ou estar vazio, de acordo com a interseção do quadrante com a área considerada. Um quadrante parcialmente contido é subdividido recursivamente em subquadrantes. Estas subdivisões continuam até que todos os quadrantes são homogêneos (inteiramente contidos ou não) ou até que determinado nível de

subdivisão predeterminado seja alcançado. As subdivisões sucessivas podem ser representadas com uma árvore onde os quadrantes parcialmente contidos são os nós internos e os quadrantes vazios ou completos são as folhas, como ilustra a Figura 2.

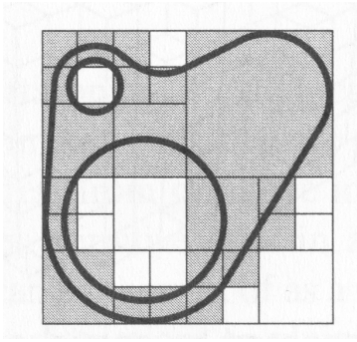


Figura 1 - Representação de um objeto usando quadtree.

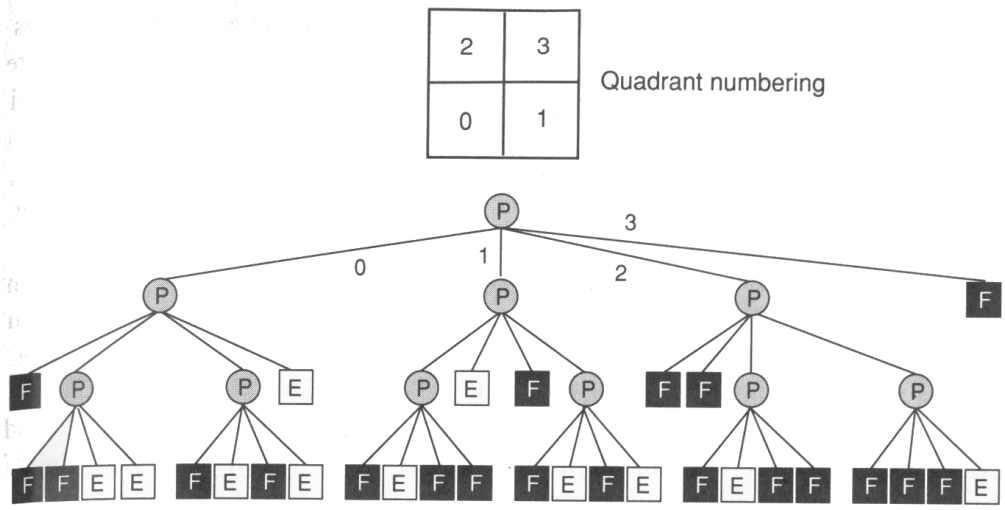


Figura 2 - Estrutura da quadtree para a Figura 1.

Os quadrantes são normalmente referidos pelos números de 0 a 3, e quando um sistema numérico não é utilizado, estes são nomeados de acordo com sua direção relativa ao centro do nó pai: NW, NE, SW, e SE.

Exceto por alguns piores casos, pode-se mostrar que o número de nós em uma quadtree que representa um objeto é proporcional ao perímetro ou superfície deste objeto. Esta relação se mantém porque a subdivisão de um nó ocorre pela necessidade de representação do contorno do objeto que se quer representar. Os únicos nós internos que são divididos são aqueles por onde passa o contorno. Portanto, qualquer operação nesta estrutura de dados que é linear no número de nós que contém também é executada em tempo proporcional ao seu perímetro ou área.

2.1. Operações e Transformações

Atualmente há muitos trabalhos que desenvolvem algoritmos eficientes de armazenagem e processamento de quadtrees. Por exemplo, operações boleanas podem ser executadas diretamente para quadtrees. Para que se faça uma união ou interseção de duas árvores, S e T, Figura 3, percorre-se ambas as árvores em paralelo. Cada par de nós é analisado. Considere o caso da união. Se ambos os nós do par é preto, então este nó é acrescentado à árvore resultante, U. Se um dos nós do para é branco, então o nó correspondente é criado em U, com o valor do outro nó do para. Se ambos os nós são cinza, então um nó cinza é adicionado a U, e o algoritmo é aplicado recursivamente aos filhos deste par. No último caso, os filhos do novo nó em U devem ser inspecionados após receberem a aplicação do algoritmo. Se todos eles são pretos, estes são removidos e o seu pai em U é alterado de cinza para preto. O algoritmo para interseção é semelhante, exceto pelas regras do branco e preto, que são alteradas. Se ambos os nós do par é branco então um nó correspondente é acrescentado em U. Se um dos nós do par é preto, então o nó correspondente é criado em U com o valor do outro nó do par. Se ambos os nós são cinza, então um nó cinza é adicionado a U, e o algoritmo é aplicado recursivamente aos filhos deste par. Como no algoritmo de união, se ambos os nós são cinza, então após a aplicação do algoritmo nos filhos do novo nó em U, os filhos devem ser inspecionados. Neste caso, se todos eles são brancos, serão eliminados e seus pais em U devem ser alterados de cinza para branco.

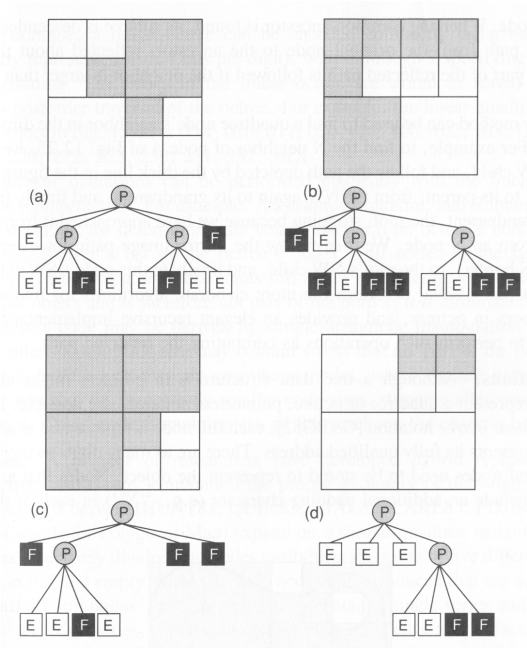


Figura 3 - (a) Objeto S. (b) Objeto T. (c) S ∪ T. (d) S ∩ T.

As transformações em quadtrees são simples de se executar. Por exemplo, a rotação em torno de um eixo de múltiplos de 90^o são feitas pela rotação recursiva dos filhos em cada nível. A escala e o espelhamento também são diretos.

2.2. Vizinhança

Uma operação importante em uma quadtree é a procura da vizinhança de um nó, ou seja, encontrar um segundo nó que seja adjacente ao primeiro e de tamanho igual ou maior (compartilhamento de uma face, aresta ou vértice). O nó de uma quadtree tem vizinhos em oito direções possíveis. Os seus vizinhos em relação às direções N, S, E e W compartilham uma mesma aresta, enquanto os vizinhos nas direções NW, NE, SW e SW têm um vértice em comum.

Para que se encontre a vizinhança de um nó em uma direção especificada, o procedimento é o seguinte: o método se inicia no nó inicial e ascende a quadtree até que o primeiro ancestral comum do primeiro nó e seu vizinho seja encontrado. A árvore é então percorrida na direção das folhas para que se encontre o vizinho desejado. Dois problemas têm que ser resolvidos eficientemente: encontrar o ancestral comum e determinar qual de seus descendentes é o vizinho. Por exemplo, deseja-se encontrar a vizinhança N do nó A, Figura 4. Inicia-se em A, que é um filho do tipo NW, e percorre-se a árvore para cima. De A, passa-se para o seu pai, um NW, depois para o seu avô, um SW e após, para o seu bisavô, o nó raiz. Finalmente percorre-se agora, para baixo, a partir da raiz, e do seu filho NW (direção refletida pelo eixo N-S) e a seguir, para o seu filho SW que é uma folha.

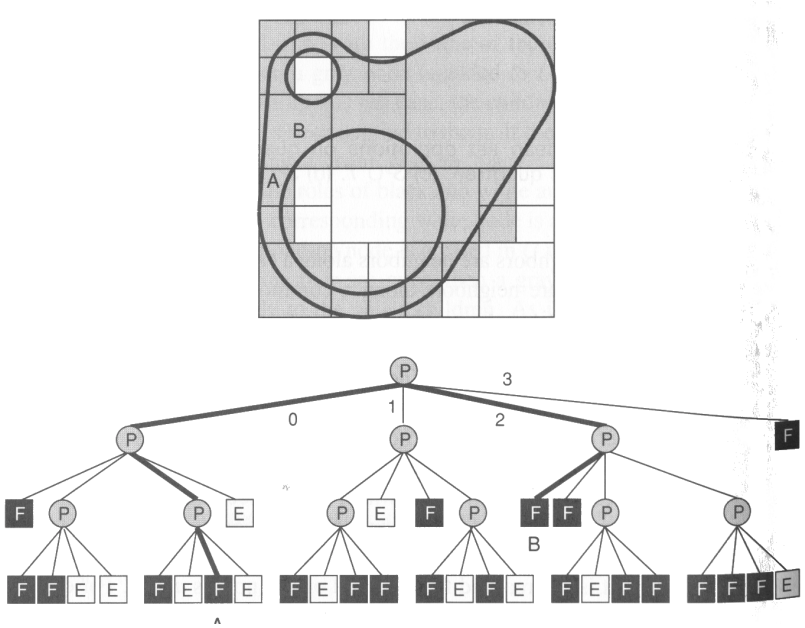


Figura 4 - Busca de vizinhança em uma quadtree.

3. GERAÇÃO DE MALHAS

O estudo de peças e materiais na engenharia, por exemplo, sempre envolvia a construção de protótipos, onde se efetuavam os estudos para os vários tipos de carregamentos a serem solicitados. A cada melhoramento no protótipo, seja no seu material ou na sua forma, havia a necessidade de sua reconstrução, envolvendo desperdício de tempo e de capital. Por outro lado, não havia outra forma de previsão do comportamento destas peças.

Desenvolveu-se, então o Método dos Elementos Finitos, onde é feita uma subdivisão da peça em vários elementos de comportamento conhecido, bem como de sua interação. Este elementos, geralmente triângulos ou quadriláteros, formam um malha, que deve possuir determinadas características, como:

- Deve ser conforme, onde não há um vértice de um elemento situado no interior da aresta do elemento vizinho;
- Os elementos de bordo devem possuir arestas coincidentes com o contorno da peça estudada;
- Deve ser constituída de elementos com boa qualidade de forma, sem ângulos internos muito pequenos, que levariam a uma convergência mais lenta do processo numérico;

Desta forma, criou-se a possibilidade de um projeto automatizado, estudando-se apenas o modelo computacional, sem a necessidade de construção de protótipos. Os elementos do modelo herdam as propriedades do material na região a que pertencem e os nós destes elementos herdam atributos aplicados nas arestas do modelo geométrico, seja de temperatura, deslocamento, ou força.

Os dados extraídos destes elementos são processados em um grande sistema de equações, que é resolvido numericamente. A precisão do método depende muito da malha gerada. Quanto mais fina, ou seja, com número maior de elementos, melhor. Porém isto acarreta um tempo de processamento numérico, computacional, muito grande. A solução é a utilização de malhas não uniformes, sendo mais fina somente onde necessários.

Considere o exemplo mostrado na Figura 5. A superfície é um quadrado com um furo, também quadrado, no canto superior esquerdo, na região onde há maior interesse de estudo. Como é pequena a distância entre a borda do furo e a borda lateral esquerda da superfície, não há outra solução melhor para esta região, utilizando malhas triangulares, senão o uso de triângulos pequenos, de aresta igual à distância referida acima. Quando se repete este triângulo para toda a superfície, observa-se que não há o refinamento da malha somente na região de interesse, levando a uma quantidade muito grande de elementos e aumentando o custo de processamento dos dados provenientes destes elementos. Uma solução seria substituir os quadrantes superior direito e inferiores direito e esquerdo por 6 triângulos maiores,

de lado igual à metade do lado do quadrado. Porém, esta malha não satisfaz a condição de conformidade. A solução é mostrada na Figura 5b , onde há um aumento gradual do tamanho do elemento triangular a medida em que este esteja afastado da região de estudo.

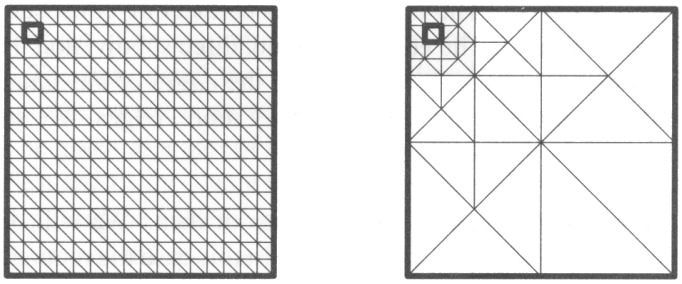


Figura 5 - Malha estruturada e não-estruturada.

Neste trabalho será visto três algoritmos que geram malhas de modo distinto, mas que usam a quadtree como informação para geração dos elementos da malha. Esses trabalhos são decorrentes da linha de pesquisa da Computação Gráfica Aplicada, do Departamento de Engenharia Civil da PUC-Rio (Vianna, 1992; Cavalcante Neto,1994; Miranda ,1998) e estão implementados como uma biblioteca 2D de geração de malha. O algoritmo de Vianna (1992) gera pontos no centro de cada célula da *quadtree* e depois faz uma contração do contorno, construindo os elementos. O algoritmo de Cavalcante Neto (1994) constrói os elementos do interior do domínio utilizando-se de padrões de elementos para cada configuração apresentada pela célula da *quadtree;* para a faixa do contorno gera os elementos por uma técnica de contração de contorno. O algoritmos de Miranda (1999) constrói os elementos por contração de contorno, porém diferente do algoritmo de Vianna os pontos são gerados juntamente com a contração do contorno.

Nas seções seguintes serão vistas os passos necessários para a construção da quadtree e também é feita a descrição dos algoritmos de geração de malhas. Os trabalhos de Vianna(1992), Cavalcante Neto (1994) e Miranda (1999) serão referenciados como Algoritmo 01, Algoritmo 02 e Algoritmo 03, respectivamente.

Os dados de entrada dos algoritmos são descritos por uma lista de nós, definidos por suas coordenadas, e uma lista com o número de arestas de cada circuito do modelo (circuito externo e os circuitos internos se existirem). Esse tipo de estrutura de dados tem alguns aspectos a serem considerados: pode representar geometria de qualquer forma de uma maneira simples, incluindo furos e cavidades, e pode ser facilmente incorporado em qualquer sistema que necessite de malhas triangulares. Tem-se então triangulação com restrição, em que as arestas são as restrições do problema.

3.1. Geração da Quadtree

A geração da *quadtree* envolve três passos. Em um primeiro passo, a *quadtree* é iniciada baseada nos dados de entrada. Em outros dois passos, a *quadtree* é posteriormente refinada. Um exemplo hipotético bidimensional, mostrado na Figura 6, é usado aqui para ilustrar o processo de geração da *quadtree*.

- **Geração da Quadtree Baseada na Discretização do Contorno**

Inicialmente, um quadrado envolvente é criada baseada nas coordenadas cartesianas máximas e mínimas dos nós dos dados de entrada. Esse quadrado é a célula raiz da *quadtree*, conforme ilustrado na Figura 7. No primeiro passo da geração da *quadtree*, cada aresta de contorno é usada para determinar a profundidade local da subdivisão. A célula da *quadtree* contendo o ponto médio de cada aresta de contorno é determinada. Se o tamanho da célula é maior que um fator do comprimento da aresta de contorno, então esta célula é subdividida em quatro células menores. Este processo é repetido recursivamente e acaba quando o tamanho da célula é menor do que o fator do comprimento da aresta de contorno. O procedimento é repetido para todas as arestas do contorno do modelo fornecidas. Com relação ao fator, é recomendado um valor entre 0,7 e 1,4 (Potyondy, 1993). Neste algoritmo adotou-se um fator de 1,0 por recomendação do trabalho de Cavalcante Neto (1994). O resultado está ilustrado na Figura 7.

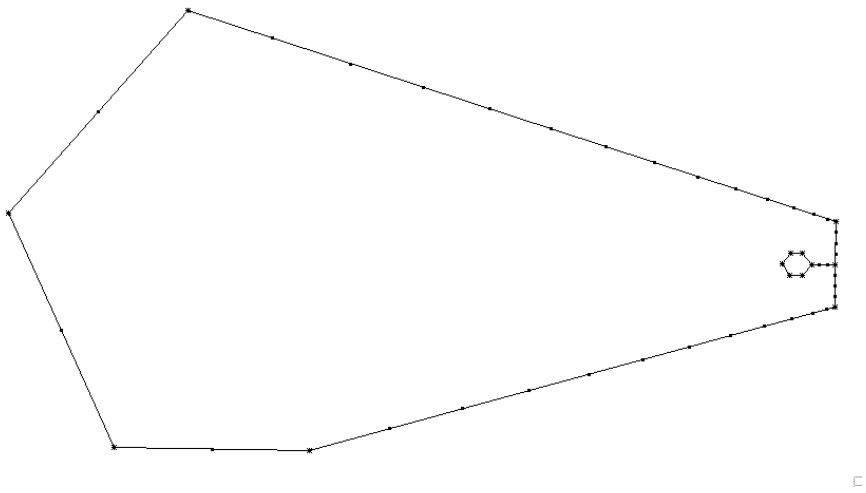


Figura 6 - Modelo bidimensional e seu refinamento no contorno.

- **Refinamento da Quadtree para Forçar um Tamanho de Célula Máximo**

A construção inicial da *quadtree* pode deixar células no interior do domínio com tamanho exageradamente maior do que das células que se situam no contorno. Neste passo, a

quadtree é refinada para garantir que nenhuma célula no interior do domínio seja maior do que a maior célula no contorno. Isto evitará a geração de elementos excessivamente grandes no interior do domínio. A Figura 8 mostra a *quadtree* resultante após esta operação.

- **Refinamento da Quadtree para Forçar Disparidade de Tamanho Mínima**

Neste passo, a *quadtree* é processada para forçar um único nível de diferença de profundidade entre células vizinhas. Isto força uma transição natural entre regiões com diferentes graus de refinamento. Esta operação é realizada percorrendo-se a *quadtree* e examinando o nível de refinamento entre células adjacentes. Se a diferença é maior do que um nível, as células apropriadas são refinadas até que o critério seja satisfeito. A Figura 9 mostra a *quadtree* gerada após este procedimento. Pode-se notar que as células próximas à ponta da trinca foram afetadas por esta fase, se comparada com a *quadtree* da fase anterior.

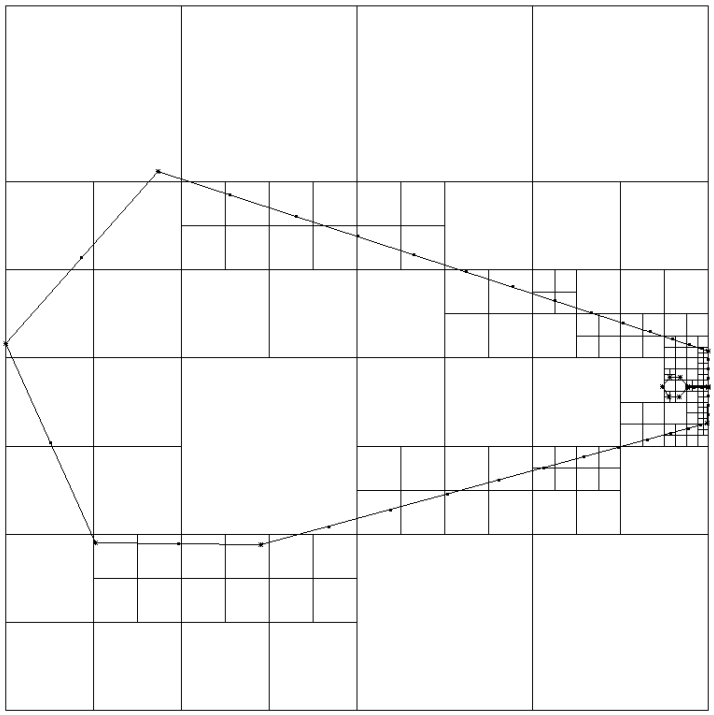


Figura 7 - *Quadtree* inicial do modelo.

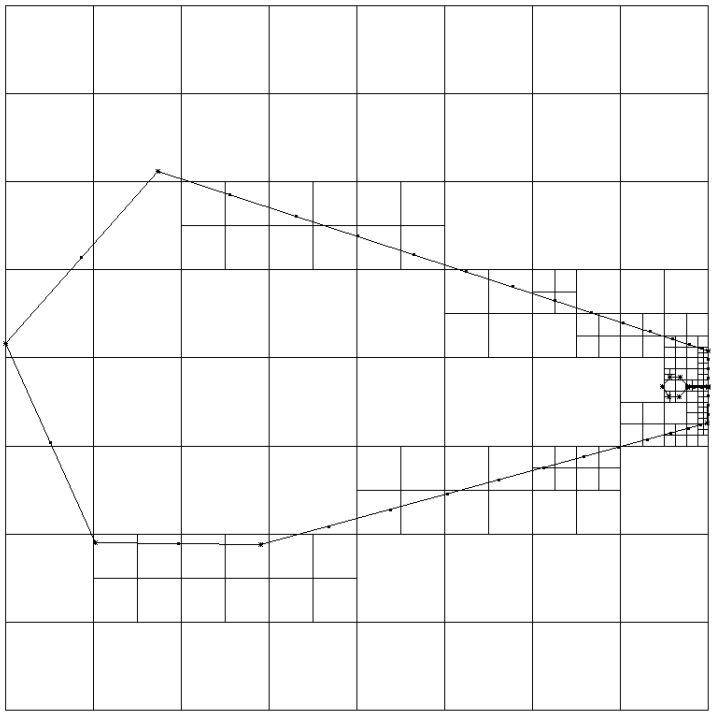


Figura 8 - *Quadtree* após forçar maior tamanho de célula do contorno.

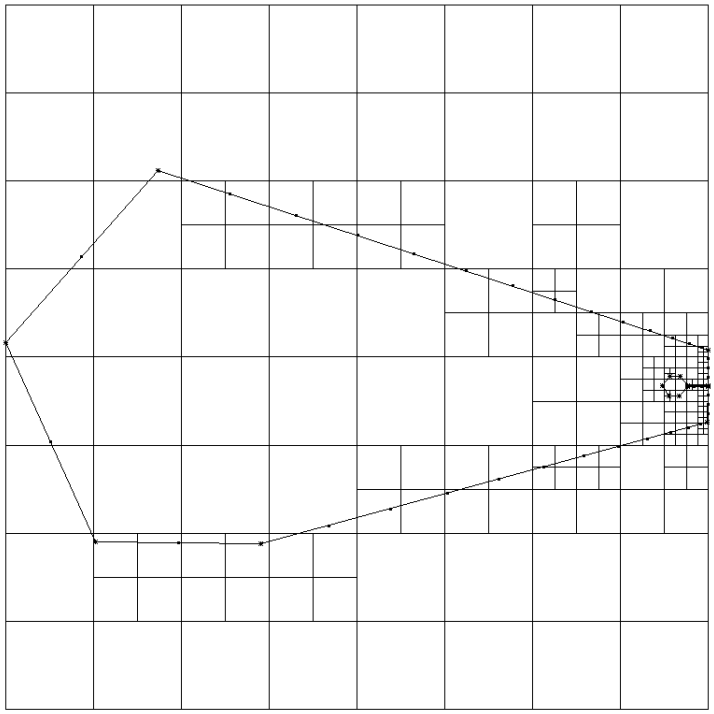


Figura 9 - *Quadtree* após forçar disparidade de tamanho mínimo entre células.

3.2. Algoritmos de Triangulação

A) Algoritmo 01 – Vianna (1992)

Este algoritmo gera *a priori* os pontos no centro de cada célula da *quadtree* e depois faz uma contração do contorno, construindo os elementos. É um algoritmo bem simples no qual a partir dos ponto interiores os elementos são gerados seguindo a propriedade de triangulação de Delaunay. O algoritmo pode ser resumido como a seguir:

- A partir dos passos anteriores da geração da quadtree, analisa-se a possibilidade de se gerar um ponto no interior de cada célula. Isto ocorre com base nas seguintes condições: (a) o ponto a ser gerado deve estar dentro da região de interesse; (b) o ponto a ser gerado não deve estar muito próximo do contorno da região. O critério de proximidade adotado estabelelece que a distância do ponto (a ser gerado) ao contorno não deve ser superior ao tamanho da célula que contém multiplicada por um fator, que é estabelecido impiricamente.
- Inicia-se uma lista com todas as arestas do contorno.
- Seleciona-se uma aresta do contorno contida na lista.
- Para todos os nós do contorno ou internos seleciona-se aquele que forma com a aresta selecionada (aresta corrente) o melhor triângulo possível. Define-se o melhor triângulo , como sendo aquele que apresenta o maior ângulo de inclusão como apresendo na Figura 10.
- Atualiza-se a lista de arestas do contorno, retirando-se a aresta selecionada e introduzindo-se as outras arestas que formaram o triângulo, caso não estejam na lista.
- Prossegue-se com o processo até que a lista fique vazia.

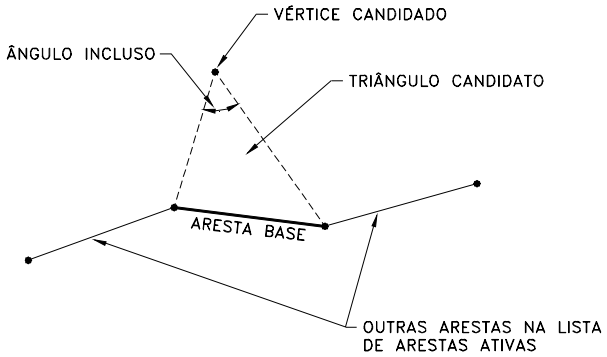


Figura 10 - Definição do ângulo para um vértice.

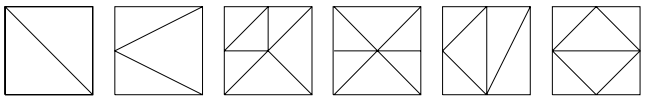
B) Algoritmo 02 – Cavalcante Neto (1994)

Antes de proceder com a geração da malha no interior do domínio após geração da quadtree, este algoritmo classifica as células. Todas as células que forem interior ao domínio irão gerar elementos através de padrões. Na região entre as células interiores e o contorno externo os elementos serão gerados utilizando-se a triangulação de Delaunay. Precisa-se então classificar bem o que são células interiores, pois células que tiverem muito perto do contorno podem ocasionar elementos de forma ruim (ângulos muito agudos, por exemplo) ao se processar a triangulação de Delaunay. A solução para isso é não considerar como células interiores as células que estejam a uma distância de uma aresta do contorno menor que uma percentagem do comprimento. No caso deste algoritmo adotou-se um fator de 0.2 (20 %).

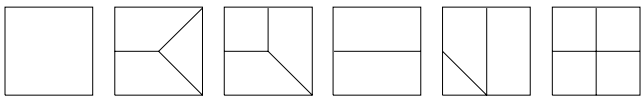
B.1) Geração de malhas em células interiores por padrões

Nesta fase são gerados os elementos em cada célula interior ao domínio, seguindo os padrões mostrados na Figura 11. Esses padrões são definidos de acordo com o número de células adjacentes em cada célula. É considerado que existe somente um nível de diferença entre células adjacentes.

Com uso destes padrões, gera-se então a primeira parte da malha, parte esta que ocupa normalmente a maior área do domínio. Na Figura 12 é mostrado o estágio do algoritmo para um exemplo após a etapa de geração da malha interna utilizando a *quadtree.*



padrões triangulares



padroes preferencialmente quadrilaterais

Figura 11 - Padrões de células interiores.

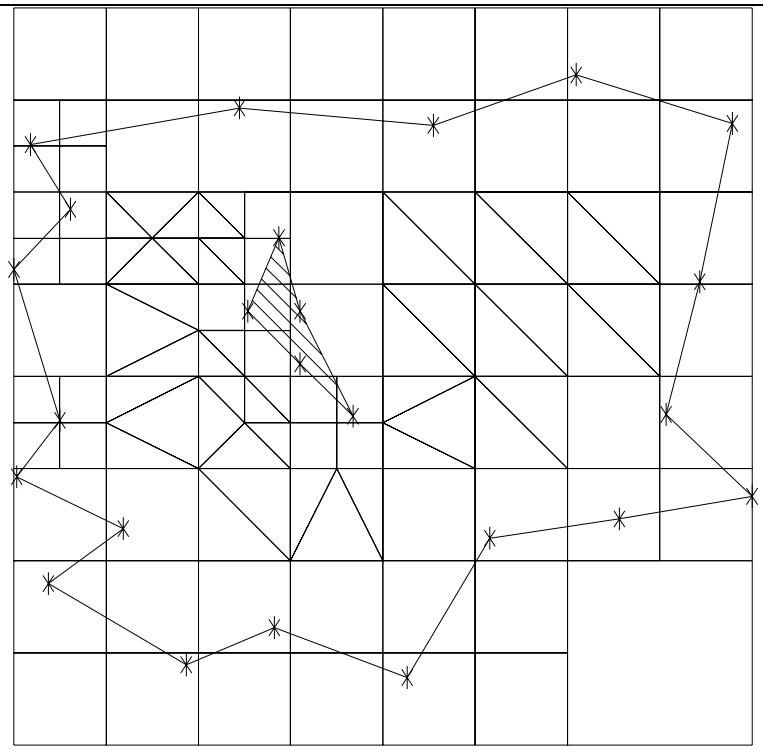


Figura 12 – Geração de elementos interiores por células para um exemplo.

B.1) Geração da malha no contorno

Esta seção define como a malha é criada na região entre a quadtree interna e o contorno. Isto é feito através da técnica de triangulação de Delaunay por contração de contorno como descrito no algoritmo de Vianna. Isto completa o método de geração de malhas.

Nesta fase é feita a contração de contorno considerando apenas os pontos criados no contorno dos elementos interiores e os pontos passados como dado de entrada do problema. Para obter as arestas do contorno da malha interior é utilizada uma árvore de ordenação B-Tree e a partir das arestas do contorno, os pontos que serão testados para contração do contorno. Desta forma tem-se um número reduzido de vértices para os quais o ângulo que deve ser computado, aumentando o a eficiência do algoritmo. Isto torna-se mais significativo se for considerado que são feitos também testes geométricos para validação do vértice candidato, que deve estar dentro do domínio no qual está sendo gerada a malha e também não podem formar arestas que interceptem alguma aresta existente.

C) Algoritmo 03 – Miranda (1999)

O procedimento de avanço da fronteira, descrito nesta seção, começa pelo contorno que limita o domínio a ser preenchido com uma malha triangular (termo aqui utilizado para definir uma triangulação). Elementos triangulares são “extraídos” ou “cortados” do domínio um por vez. Quando um elemento é extraído, o contorno limitante é atualizado e o processo é repetido. O procedimento termina quando o domínio inteiro contiver uma malha triangular.

Neste algoritmo, o processo de avanço da fronteira é dividido em duas fases. Em uma primeira fase, uma geração de elementos baseada em geometria é tentada para gerar elementos de formas ótimas. Após esta fase ideal ser exaurida, e elementos ótimos não poderem ser mais criados, passa-se para geração de elementos baseada em topologia, criando elementos válidos na região restante, mas não necessariamente de boa forma.

C.1) Geração de Elementos baseada em Geometria

Idealmente, a malha no domínio seria inteiramente gerada na fase baseada em geometria. Isto depende da geometria e topologia do contorno do modelo fornecido e está fortemente relacionado com a discretização de contorno fornecida.

C.1.1) Listas de Contração de Contorno

O processo começa com a criação de uma fronteira de avanço inicial, que é formada pelos segmentos discretizados do contorno fornecido como dado de entrada. Os segmentos do contorno corrente são armazenados em duas listas duplamente encadeadas. A primeira lista é uma lista de arestas ativas, que contém as arestas de contorno que não foram ainda usadas em uma tentativa de gerar triângulos válidos. A outra é uma lista de arestas rejeitadas, isto é, que contém as arestas que falharam na geração de elementos para a fase corrente. Inicialmente, as arestas de contorno fornecidas são inseridas na primeira lista e a segunda lista é vazia.

A lista de arestas ativas é ordenada pelo comprimento das arestas. Isto garante que a primeira aresta selecionada será sempre a aresta com o menor comprimento. Isto tem sido recomendado por outros autores (Peraire et al., 1988) para prevenir que elementos grandes penetrem em regiões com arestas de comprimento pequenos.

Também é conveniente para alguns passos do algoritmo ter uma estrutura de dados adicional que contém as duas arestas do contorno adjacentes a cada nó da fronteira de avanço

corrente. Esta estrutura de dados é inicializada para todos os nós do contorno fornecido. A estrutura de dados é atualizada quando o procedimento de contração de contorno progride.

C.1.2) Geração de Elementos Ótimos

Na fase de geração de elementos baseada em geometria, o contorno corrente avança tentando formar triângulos baseados principalmente em considerações geométricas. A cada passo, uma aresta do contorno, referenciada com *aresta base,* é escolhida da lista de arestas ativas. Esta possui o menor comprimento na fronteira corrente e sua normal aponta para o interior da região a ser triangulada.

Os critérios para gerar um triângulo nesta fase estão explicados com auxílio da Figura 13. O procedimento é dividido nos seguintes passos:

- A localização ótima *N1* para o vértice do triângulo a ser formado é determinada com a ajuda da *quadtree*. A célula da *quadtree* contendo o ponto médio da aresta base é determinada. O ponto ótimo *N1* se localiza na linha perpendicular a aresta base passando pelo seu ponto médio, *M*. A distância do ponto ótimo para o ponto médio da aresta base é igual ao tamanho da célula da *quadtree*.
- O ponto ótimo define uma região ótima onde o vértice do triângulo a ser formado está localizado. Essa região é um setor de círculo cujo centro é o ponto ótimo e cujo raio é proporcional ao tamanho da célula da *quadtree*. Na implementação corrente, um fator igual a 0,85 foi adotado para essa proporção. Esse círculo define um limite superior para a distância entre o vértice alvo do triângulo e o ponto médio da aresta base. Um limite inferior é definido para garantir que o triângulo a ser gerado tenha uma área maior que a menor área aceitável. Nesta implementação este limite inferior é definido pelo triângulo com altura igual a 1/10 do comprimento da base. A região ótima é usada por duas razões. Primeiro, para assegurar qualidade na forma de todos os elementos gerados e, segundo, para assegurar que novos nós internos somente sejam criados quando é estritamente necessário e sempre em boas posições.

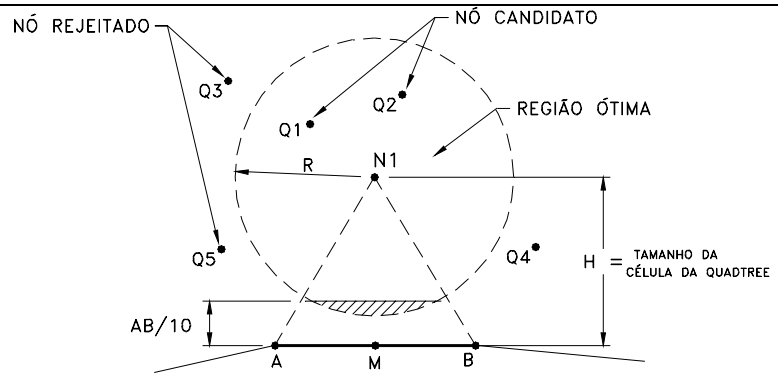


Figura 13 - Região ótima de vértice candidato de um triângulo.

- Se nenhum nó existente está dentro da região ótima, um novo nó é inserido na localização ótima *N1* e um elemento é gerado usando este nó. Se somente um nó existe nesta região, este nó é usado para formar o elemento. Se mais de um nó são encontrados na região ótima, eles são classificados de acordo com o ângulo que eles formarão com a aresta base. Uma lista auxiliar é usada para selecionar eficientemente esses nós. O nó que formará o maior ângulo incluso (veja Figura 10) é usado para gerar o elemento.
- Uma vez que um triângulo válido é formado para a aresta base corrente, a lista de arestas ativas é atualizada. Isto é feito através dos seguintes passos. Primeiro, a aresta base é removida da lista de arestas ativas. Então, para as outras arestas do elemento: cada aresta é eliminada se coincide com uma aresta já existente na lista, ou é inserida na lista como uma nova aresta.
- Devido aos limites geométricos impostos pela fase corrente do avanço da fronteira, existem situações em que o algoritmo falha em formar um triângulo válido para a aresta base corrente do contorno. Nestes casos, a aresta base corrente é removida da lista de arestas ativas da contração de contorno e é armazenada na lista separada de faces rejeitadas. Pode acontecer que esta aresta seja removida subseqüentemente desta última lista se ela for usada na criação de um triângulo válido para uma aresta base adjacente.
- Quando não existem mais arestas na lista de arestas ativas da contração de contorno, o algoritmo tenta gerar elementos usando as arestas que foram rejeitadas previamente. Pode acontecer de arestas que falharam previamente na tentativa de formar um elemento possam ser capazes de funcionar agora, uma vez que a fronteira muda com a adição de novas arestas de novos elementos gerados. A fase de geração de elementos baseada em geometria acaba quando não existem mais arestas em ambas as listas da contração de contorno (em cujo caso uma malha ótima foi gerada) ou quando uma aresta rejeitada falha uma segunda vez.

C.2) Geração de Elementos Baseada em Topologia

O objetivo desta fase do algoritmo é forçar a geração de triângulos válidos, mesmo que os novos elementos não satisfaçam os limites usados na fase anterior para forma dos elementos.

A geração de elementos baseada em topologia começa quando uma aresta de contração do contorno falha duas vezes na tentativa de gerar um elemento ótimo. A lista de arestas de contorno rejeitadas da fase anterior é transformada na lista de arestas ativas de contorno. É sempre possível gerar uma triangulação válida do ponto de vista topológico para qualquer polígono simples (Figueiredo e Carvalho, 1991).

Na fase de geração de elementos baseada em topologia, qualquer nó próximo da face base corrente é selecionado e armazenado em uma lista local de nós candidatos. O nó que tem o maior ângulo com respeito à aresta base é escolhido para a geração do novo triângulo. Se as arestas deste triângulo incluso não interceptarem qualquer outra aresta da fronteira de avanço corrente, o elemento é criado e o contorno é contraído. A fase baseada em topologia acaba quando a lista de arestas ativas do contorno fica vazia.

3.3. Melhoria Local da Malha

Uma técnica de suavização é usada para melhorar a qualidade da malha através do reposicionamento de nós dentro de um *patch* (grupo de elementos adjacentes). Uma formulação desta técnica é dada através da equação (3.3), que é a forma genérica de uma função Laplaciana ponderada (Foley *et al*., 1996):

$$X_0^{n+1} = X_0^n + \phi \frac{\sum_{i=1}^{m} w_{i0}(X_i^n - X_0^n)}{\sum_{i=1}^{m} w_{i0}}. \qquad (1)$$

Na equação (1), *m* é o número de nós conectados ao nó *0*, X_0^{n+1} é a posição do nó *0* com *n+1* interações de suavização, w_{i0} é a função ponderada entre o nó *i* e *0*, e ϕ é o parâmetro de relaxação que é normalmente entre 0 e 1. Neste trabalho, adotou-se $\phi = 1$ e $w_{i0} = 1$ resultando assim em uma média simples das coordenadas dos nós adjacentes. Essa média é repetida quatro ou cinco vezes para dar resultado final satisfatório e afeta somente os vértices interiores, já que não se pode alterar os coordenadas dos vértices originais do contorno.

4. ANÁLISE DE COMPLEXIDADE

A forma recursiva da definição de uma quadtree nos leva imediatamente a um algoritmo recursivo. De acordo com o problema estudado, construi-se quadrantes sucessivos até à finalização desejada. O único detalhe que não segue a definição recursiva é encontrar o quadrado inicial, de onde se iniciará a construção. Algumas vezes este passo inicial é dado do problema, outras vezes a sua definição pode ser feita em tempo linear, a partir dos pontos de ordenadas e abscissas mínimas e máximas, como é o caso deste trabalho, de geração de malhas.

Fazendo-se analogia ao problema proposto por De Berg (1997), onde se deseja armazenar um grupo de pontos do plano, faremos uma análise da complexidade da criação de quadtrees para auxílio na geração de malhas.

No problema dos pontos, a cada passo, o quadrado contendo os pontos é subdivido em quatro menores, até que não haja dois ou mais pontos em uma mesma célula. Com isso, dependendo do quadrado inicial e da disposição dos pontos, a quadtree pode ficar desbalanceada, e a sua profundidade não poderia ser expressa em função da quantidade de pontos que ela armazena. Portanto, a profundidade desta quadtree está relacionada à distância entre os pontos e o tamanho de quadrado inicial. Então, De Berg estabelece o seguinte lema: a profundidade de uma quadtree para um grupo P de pontos no plano é pelo menos log(s/c) +3/2, onde c é a menor distância entre quaisquer dois pontos em P, e s é o comprimento do lado do quadrado inicial que contém P.

Fazendo-se analogia ao nosso caso, onde a subdivisão só termina quando o comprimento do lado do quadrado formado é menor que o comprimento da aresta cujo ponto médio esteja no interior deste quadrado, o mesmo lema pode ser seguido, onde c passa a ser o comprimento da aresta.

A análise acima foi feita porque o tamanho de uma quadtree, e seu custo de construção, está diretamente relacionado à sua profundidade e ao número de dados que ela armazena. Cada nó interno de uma quadtree tem quatro filhos, então, o número total de folhas é três vezes o número de nós internos mais um. Qualquer nó interno possui no interior de seu quadrado associado, dados cuja relação com o tamanho deste quadrado, ainda não satisfazem à condição de finalização da subdivisão. E ainda, somando-se todos os quadrados relacionados às células de um mesmo nível da quadtree, tem-se o quadrado inicial. Portanto, isto significa que o número máximo de nós internos em determinada profundidade é n. Se para cada nó interno o tempo gasto é linear ao número de dados do problema contidos no quadrado associado, sendo o número máximo em uma mesma profundidade igual a n, uma quadtree pode ser construída em tempo O((d+1)n) no pior caso, onde d é a sua profundidade.

Como no algoritmo apresentado neste trabalho, a medida em que se constrói a quadtree dados relativos à vizinhança já são armazenados em quatro ponteiros em cada célula, o tempo gasto para busca na árvore é linear ao número de nós, no pior caso.

A análise de complexidade dos algoritmos de triangulação será feita em cima de seus pontos críticos, pois apesar de se terem descrições relativamente simples, apresentam complexidade internas quando é feita a implementação.

No algoritmo de Vianna cada aresta deve-se procurar entre todos os pontos qual é o melhor candidato, o que representa uma complexidade de ordem quadrática O(n^2) na análise do pior caso.

O algoritmo de Cavalcante Neto usa a quadtree para gerar a malha somente no interior do modelo, simplificando um pouco o processo de gerar todos os elemento baseados na quadtree. Esta operação de gerar os elementos interiores é feita em tempo proporcional ao número de folhas da quadtree pelo número de células padrões existentes. A decisão de qual é o padrão de elemento é feita consultando as células padrões. A operação total resulta em tempo O(kn). Na fase de contração de contorno é evitado testes exaustivos contra vértices gerados no interior do domínio em que utiliza uma estrutura *B-Tree* para determinar o contorno do elementos interiores. Tem-se complexidade O($n\ log(n)$) devido a estrutura *B-Tree*. Porém, se os elementos no interior do modelo forem poucos, ainda assim a fase de contração de contorno testará todos os vértices do modelo, levando a uma complexidade perto de O(n^2) na análise do pior caso.

No algoritmo de Miranda os pontos interiores são gerados a medida que o contorno vai se contraindo. No início do algoritmo a aresta testa o melhor triângulo com apenas os pontos do contorno passado. Na fase final do algoritmo o teste é feito com todos os pontos gerados no domínio, logo o teste do melhor triângulo cresce de proporcional com o avanço da fronteira. Seja a e n o número de pontos do contorno e total de pontos, respectivamente. Uma primeira análise de complexidade é O($n\ (a+log\ n)$). No caso do algoritmo não conseguir incluir nenhum ponto no domínio, a análise de complexidade torna-se O(n^2) para o pior caso.

5. Comparação De Performance E Qualidade Dos Elementos Entre Os Algoritmos De Triangulação

Nesta seção são feitas comparações entre os algoritmos de triangulação apresentados neste trabalho. Os algoritmos propostos foram implementados em um modelador geométrico de malha, o Mtool (1992), para se ter as mesmas condições para avaliação do tempo de processamento.

O tempo medido para cada algoritmo foi feito quando da chamada da função que gera os elementos triangulares e logo após a sua saída. O tempo medido é estritamente do algoritmo, sem computar o processamento que o modelador geométrico gasta para construir a estrutura de dados de entrada e o tempo de restauração dos dados de saída. O computador utilizado nesta operação foi um computador da linha PC (*Personal Computer*) com processador Intel de 200 Megahertz, memória RAM de 32 Megabytes, sistema Windows 95 mono-usuário.

Para avaliar a qualidade das malhas triangulares geradas pelo algoritmo descrito, uma análise pode ser feita baseada na métrica normalizada γ/γ^* adotada. Esta métrica foi detalhadamente descrita em Miranda (1999). Esta métrica tem um intervalo de valores válidos que variam de 1,0 a infinito ($[1, \infty)$) e o valor ótimo para o triângulo (equilátero) é igual a 1,0. É desejável ter a maioria dos elementos da malha com valores de γ/γ^* próximos a 1,0.

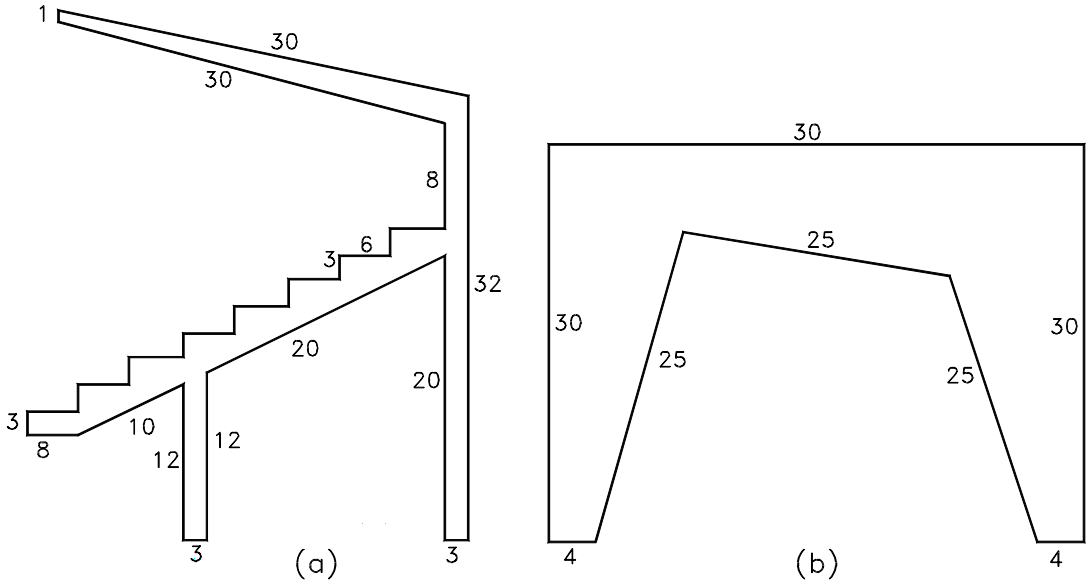


Figura 15 - Modelos: (a) arquibancada e (b) pórtico.

As comparações de forma e de tempo foram feitas para os seguintes modelos: um quadrado discretizado em 60 arestas com igual comprimento em cada lado, totalizando 240

arestas; um círculo discretizado uniformemente em 120 arestas; uma arquibancada discretizada conforme ilustra a Figura 15(a), em que os números mostrados são as divisões do contorno; e um pórtico discretizado uniformemente conforme ilustra a Figura 15(b). O número de elementos gerados para cada modelo está apresentado na Tabela 3.1.

A Tabela 3.2 apresenta os tempos de processamento e a Tabela 3.3 apresenta o número de elementos gerados por segundo em cada modelo. Pode-se verificar através dessas tabelas que o algoritmo de Miranda é mais rápido que os outros algoritmos.

Os resultados da análise de forma dos elementos da malha de elementos são apresentados na forma de histogramas através das Figuras 16 a 19. Nos histogramas, o eixo horizontal representa a métrica normalizada em intervalos de 0,1. O eixo vertical representa o percentual de elementos dentro dos intervalos da métrica.

Em todos os modelos, o algoritmo de Miranda consegue gerar elementos triangulares com maior qualidade em relação aos algoritmos de Vianna e Cavalcante Neto.

Tabela 1 - Número de elementos gerados para os modelos comparados.

Exemplo	Algoritmo		
	Vianna	Cavalcante Neto	Miranda
Quadrado	8070	1560	6330
Círculo	1670	600	4994
Arquibancada	499	788	751
Pórtico	979	1207	1061

Tabela 2 - Tempo de processamento para os modelos comparados (segundos).

Exemplo	Algoritmo		
	Vianna	Cavalcante Neto	Miranda
Quadrado	67,61	4,83	13,62
Círculo	4,34	2,64	8,29
Arquibancada	2,75	13,73	1,70
Pórtico	2,58	9,40	1,59

Tabela 3 – Número de elementos gerados por segundo.

Exemplo	Algoritmo		
	Vianna	Cavalcante Neto	Miranda
Quadrado	119.3	323.0	464.7
Círculo	384.8	227.3	602.4
Arquibancada	181.5	57.4	441.8
Pórtico	379.5	128.4	667.3

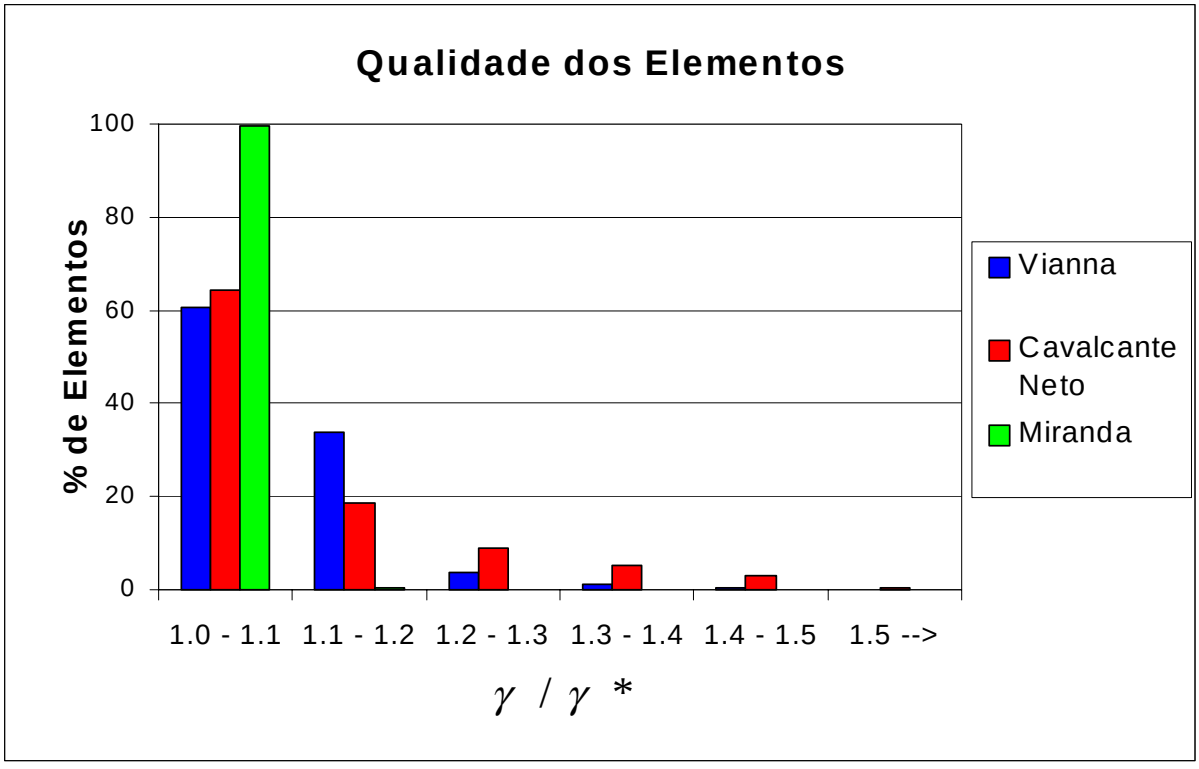


Figura 16 - Histograma de análise de forma para o exemplo de um quadrado.

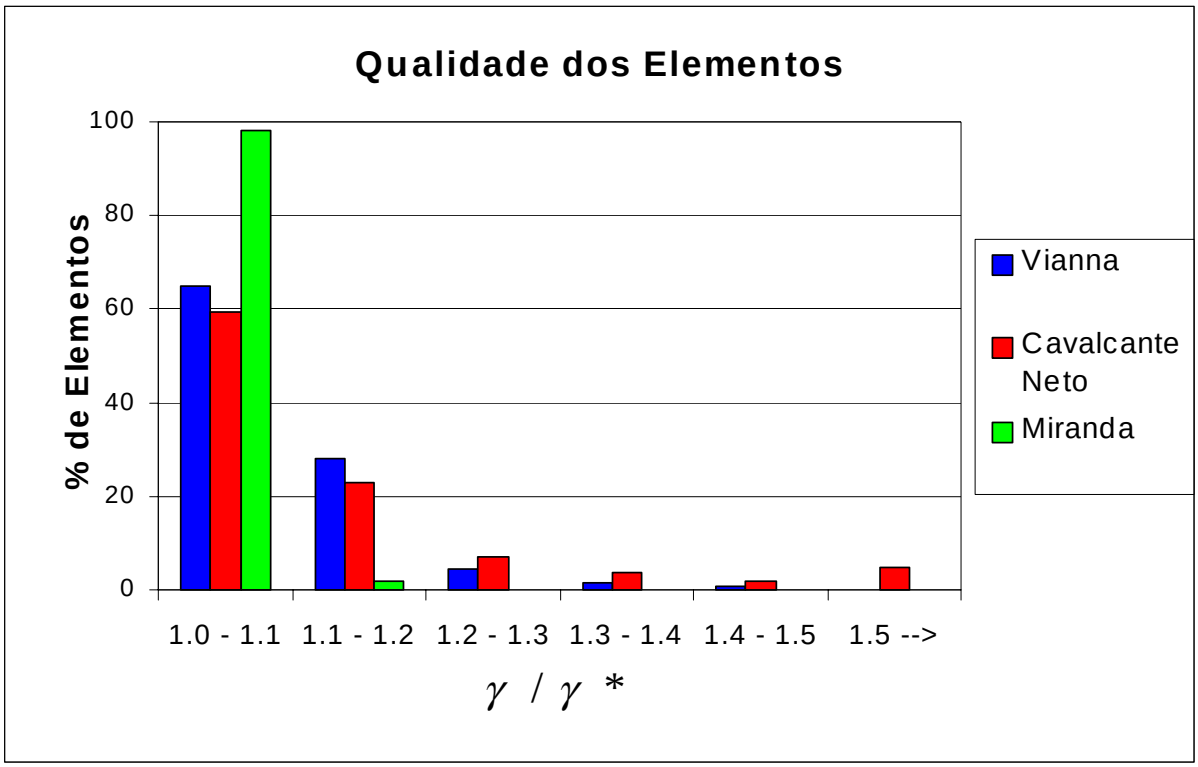


Figura 17 - Histograma de análise de forma para o exemplo de um círculo.

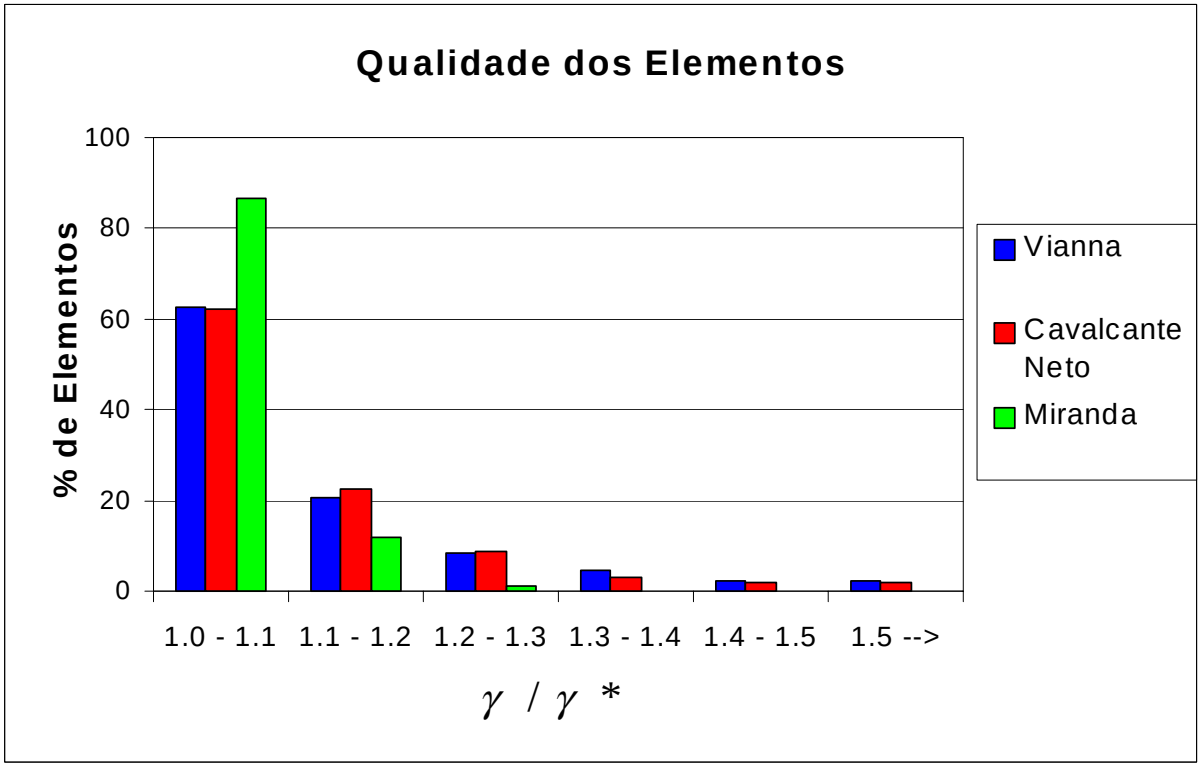


Figura 18 - Histograma de análise de forma para o exemplo de uma arquibancada.

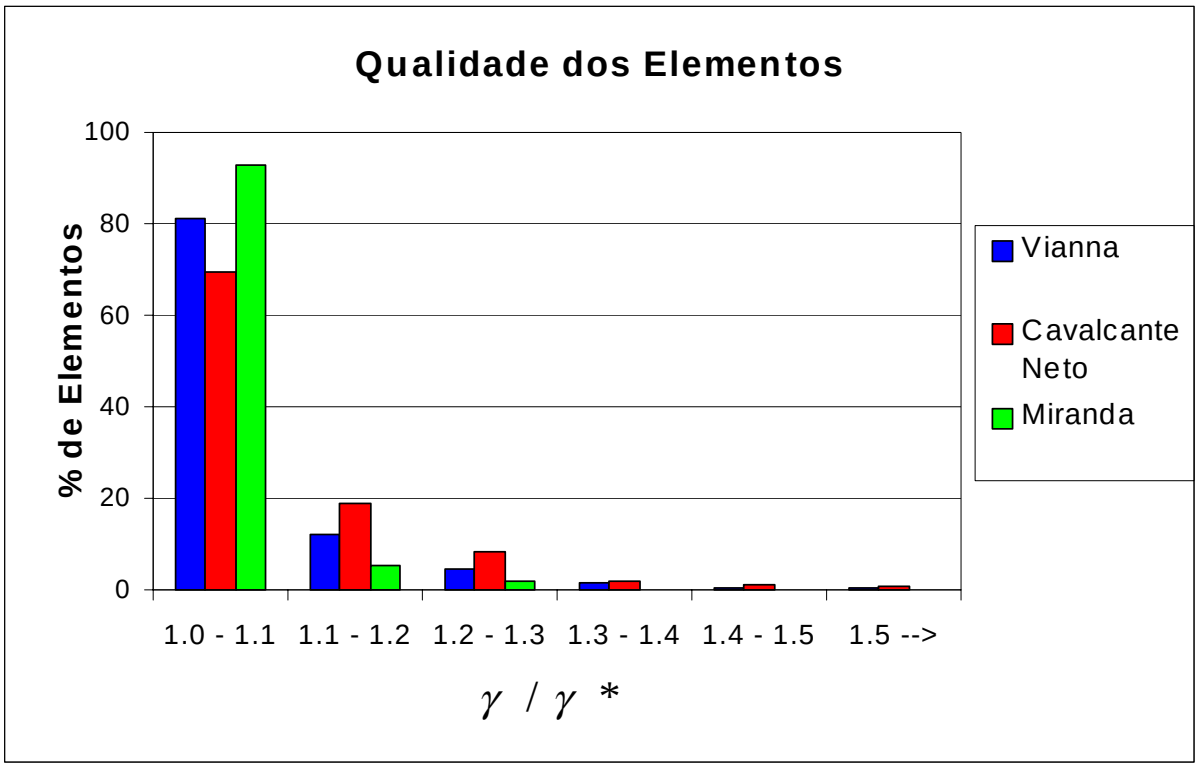


Figura 19 - Histograma de análise de forma para o exemplo de um pórtico.

6. CONCLUSÃO

A quadtree é uma estrutura de dados que tem aplicações na computação gráfica, análise de imagens, sistemas de informação geográficas e muitas outras áreas. Em específico é usada aqui para geração de malhas não-estruturas triangulares. Os seguintes passos são seguidos para geração da quadtree a partir de um modelo representado por uma lista de nós e arestas:

- Geração baseada na discretização do contorno fornecida;
- Refinamento para forçar um tamanho de célula máximo;
- Refinamento para forçar disparidade de tamanho mínima entre células adjacentes.

Os algoritmos de geração de malhas bidimensionais por triangulação apresentados utilizam-se da técnica de *Enumeração Espacial Recursiva*, através do uso de árvores quaternárias (*quadtree*), para gerar os elementos triangulares. O algoritmo de Vianna (1992) gera pontos no centro de cada célula da *quadtree* e depois faz uma contração do contorno, construindo os elementos. O algoritmo de Cavalcante Neto (1994) constrói os elementos do

interior do domínio utilizando-se de padrões de elementos para cada configuração apresentada pela célula da *quadtree;* para a faixa do contorno gera os elementos por uma técnica de contração de contorno. Os elementos gerados por esses algoritmos demostram ter uma grande dependência da árvore *quadtree*. No caso do algoritmo de Miranda(1999), os elementos não estão necessariamente atrelados à árvore *quadtree*, pois esta é utilizada apenas para dar uma gradação de transição na geração de elementos. Isto permite uma maior flexibilidade e controle na forma dos elementos.

7. REFERÊNCIAS

(Cavalcante Neto, 1994) Cavalcante Neto, J. B. - "Simulação Auto-Adaptativa Baseada em Enumeração Espacial Recursiva de Modelos Bidimensionais de Elementos Finitos", Dissertação de Mestrado, Departamento de Engenharia Civil, PUC/Rio,1994.

(De Berg *et al*., 1997) de Berg, M.; van Kreveld, M.; Overmars, M.; Schwarzkopf, O. – *Computacional Geometry (Algorithms an Applications)*, Springer-Verlag Berlin Heidelberg – Germany, 1997.

(Figueiredo e Carvalho, 1991) de Figueiredo, L. H. e. de Carvalho, P. C. P - *Introdução à Geometria Computacional*, 18º Colóquio Brasileiro de Matemática - RJ, 1990.

(Foley *et al*., 1996) Foley, J. D.; van Dam, A.; Feiner, S. K. e Hughes, J. F. - *Computer Graphics (Principles and Practice)*, 2nd. Edition, Wesley Publishing Company, Reading - MA, 1996.

(Miranda, 1999) Miranda, A. C. de O. - "Integração de Algoritmos de Geração de Malhas de Elementos Finitos", Dissertação de Mestrado, Departamento de Engenharia Civil, PUC/Rio,1999.

(Mtool, 1992) Bidimensional Mesh Tool - Manual do Usuário, PUC-Rio, 1992.

(Peraire *et al.*, 1988) Peraire, J.; Peiro, J.; Formaggia, L.; Morgan, K. e Zienkiewicz, O. C. - "Finite Euler Computation in Three-Dimensions", *International Jornal For Numerical Methods in Enginnering*, vol. 26, pp. 2135-2159, 1988.

(Potyondy, 1993) Potyondy, D. O. - "A Software Framework for Simulating Curvilinear Crack Growth in Pressurized Thin Shells", Ph.D. Thesis, School of Civil Engineering, Cornell University, 1993.

(Vianna, 1992) Vianna, A. C. - "Modelagem Geométrica Extendida para Modelos Bidimensionais de Elementos Finitos", Dissertação de Mestrado, Departamento de Engenharia Civil, 1992.