PR-QuadTree

Camila Pereira Sales

June 6, 2019

1 Introdução

Este trabalho tem como objetivo apresentar a estrutura de dados QuadTree com foco principal na variante PR-Quadtree, bem como projetar e implementar uma PR-QuadTree, e apresentar suas aplicações, para situalização sobre esta estrutura foi utilizado como referência o artigo de revisão de Samet (1984)[7].

2 QuadTree

QuadTree é uma forma de representação bidimensional, a característica 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, a cada divisão o quadrante é dividido em quatro partes iguais [1]. A QuadTree tem três teoremas sendo eles [4]:

- Uma QuadTree de profundidade D, armazenando um conjunto de N pontos requer tempo O((D+1)N) para ser gerada.
- Seja T uma QuadTree de profundidade D. O vizinho de um dado nó em uma dada direção, como definida acima, pode ser encontrado em tempo O(D+1).
- Seja Q uma QuadTree com M nós, então a versão balanceada de Q tem O(M) nós, e pode ser construída em tempo O((D+1)M).

A altura de uma QuadTree, é o máximo nível de suas folhas. Uma QuadTree é dita balanceada se quaisquer dois nos vizinhos diferem no máximo de um fator dois, desta forma em uma QuadTree balanceada, quaisquer duas folhas cujos nos são vizinhos, tem profundidades diferindo no máximo de 1 [5]. A vizinhança são os quadrantes vizinhos de um quadrante, sendo o quadrante Noroeste vizinho do quadrante Nordeste, que por sua vez tem como vizinhos os quadrantes Noroeste e Sudoeste que tem como vizinhos os quadrantes Nordeste e Sudeste, que tem como vizinho apenas o quadrante Sudoeste.

As principais aplicações da QuadTree são [4]:

- Processamento de imagens;

- Representação de pontos, curvas, superfícies e volumes;
- Detecção de colisão;
- programas de simulação;
- Indexação espacial.

3 R-QuadTree

A R-QuadTree foi inicialmente utilizada em processamento de imagens binárias, com critério de divisão dividir para conquistar, onde os quadrantes eram divididos até que restasse apenas uma informação por quadrante, as regiões com apenas uma informação não eram divididas, não alterando sub-regiões uniformemente pretas ou brancas.

Os nós podem ser de três cores, sendo elas branco, cinza e preto, os nós são ditos brancos, quando não contem a parte da região, cinzas, quando contém parte da região e parte da não-região, e brancos quando não contém dados da região a distribuição pode ser vista na figura 1.A complexidade do algoritmo da R-QuadTree são:

- localização de ponto é $O(\log_n)$.
- Conversão de imagem matricial: O(n).
- Vizinhança: $O(\log_n)$.

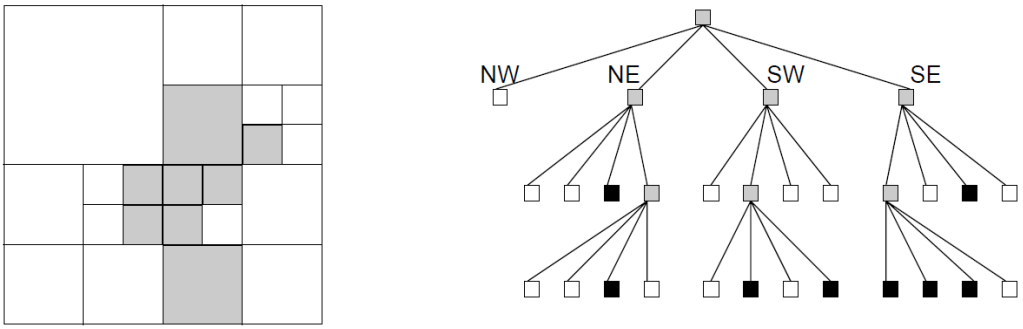

Figure 1: Distribuição R-QuadTree
Adaptado de [3]

4 P-QuadTree

Extensão multidimensional da árvore de busca binária, onde os pontos são armazenados em nós internos, ela é sensível à ordem de inserção dos pontos, caso os pontos sejam inseridos em ordem distintas a distribuição da árvore será diferente, para N pontos inseridos segundo uma distribuição randômica uniforme, a altura esperada da árvore é $O(\log_n)$.

O algoritmo de inserção é um algoritmo "ingênuo" e assemelha-se à inserção realizada em em árvores binárias. Se os pontos são conhecidos previamente, convém ordená-los segundo x ou y e escolher as medianas como raízes das subárvores.

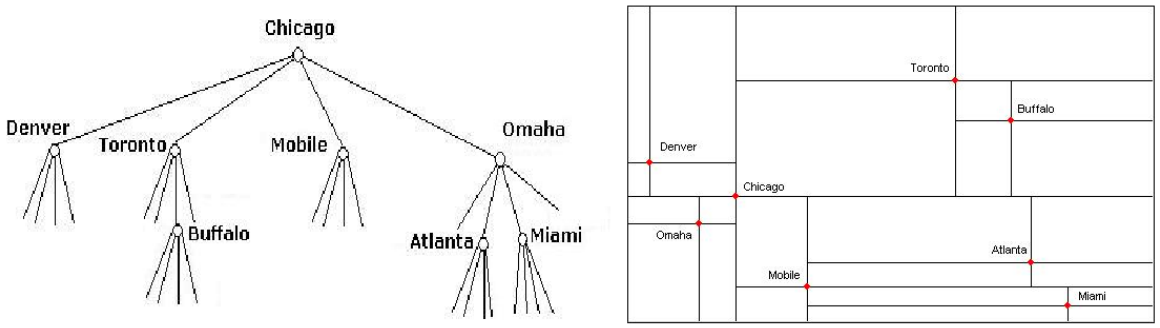

Figure 2: Distribuição P-QuadTree
Adaptado de [8]

5 PR-QuadTree

A PR-QuadTree deriva da R-QuadTree e é muito semelhante a ela, a diferença é o tipo de informação armazenada sobre os quadrantes, em uma R-QuadTree, é armazenado um valor uniforme que se aplicava a toda a área do quadrante, ja os quadrantes de uma PR-QuadTree, no entanto, armazenam uma lista de pontos que existem dentro do quadrante.

Devido a subdivisão ocorrer sempre que mais de um ponto aparece num mesmo quadrante, a altura depende da distribuição espacial dos pontos. Como o P-QuadTree, o PR-QuadTree também pode ter uma altura linear quando recebe um conjunto "ruim"[2], mas elas diferenciam-se quando levado em consideração o conjunto de entrada e a distribuição na árvore, uma vez que na PR-QuadTree a ordem que os pontos são inseridos não importa, o mesmo conjunto inserido de diferentes formas apresenta o mesmo resultados de distribuição espacial na árvore.

A altura máxima depende da distancia euclidiana que separa dois pontos, dada uma região quadrada de lado S, e supondo que a distancia mínima entre dois pontos é D, a altura máxima da PR-QuadTree é $[\log(S/D)\sqrt{2}]$.

Algumas Variações da PR-Quadtree são:

- PR-bintree

  Semelhante a PR-QuadTree, mas divisão ocorre de forma binária alternando entre os eixos [3]

- BD-tree

  PR-QuadTree onde nós cinza com apenas um filho são "comprimidos" O Semelhantemente, uma BD-k-d tree usa compressão em uma PR-bintree (PR-k-dtree) [3]

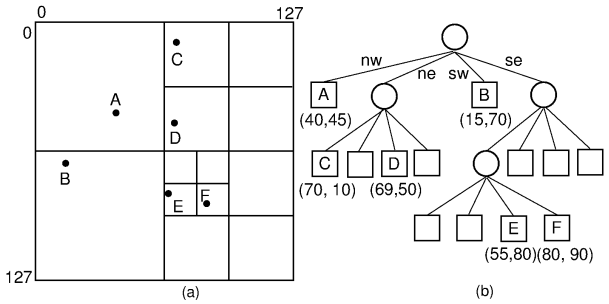

Figure 3: Distribuição PR-QuadTree
[6]

5.1 Inserção

Para que a inserção possa ser feita primeiramente deve ser obtido o quadrante que o ponto deve ser inserido, para isso deve-se recuperar o nó folha que contém as coordenadas do novo ponto, caso esse quadrante esteja vazio, o ponto pode ser inserido sem necessidade de nenhuma outra função, caso contrário é necessário realizar a divisão do quadrante, que resultará em novos 4 sub-quadrantes, este processo deve se repetir até que seja possível separar os pontos em quadrantes diferentes, após essa divisão os pontos são inseridos.

5.2 Busca

A busca de um ponto é realizada de acordo com as coordenadas, esta busca ocorre de forma recursiva, onde primeiramente é analisado o nó raiz, caso ele não contenha o ponto e não seja uma folha, essa verificação é feita passando o sub-quadrante do nó onde as coordenadas no ponto devem estar, isso é repetido até que o ponto seja achado ou até que o nó analisado não tenha mais filhos.

6 Desenvolvimento

Realizou-se a implementação de um código em C++ que possibilita a inserção e busca de pontos em uma PR-QuadTree, a implementação está disponível em "https://github.com/Pssales/PRQuadTree", foram criadas as classes *Point*, responsável por armazenar os dados e as coordenadas do ponto,*Node*, classe que representa os nós/quadrantes da PR-QuadTree e *PRQuadTree*, que representa a estrutura da árvore, como apresentado no diagrama de classes (figura 4).

7 Conclusão

Existem muitas variantes da QuadTree, devido a sua ampla variedade de aplicações, como análise de imagens, sistemas de informação geográficas e computação gráfica, neste trabalho foram apresentadas três delas, sendo a R-QuadTree, P-QuadTree e PR-Quadtre, a última apresentada de forma mais detalhada.

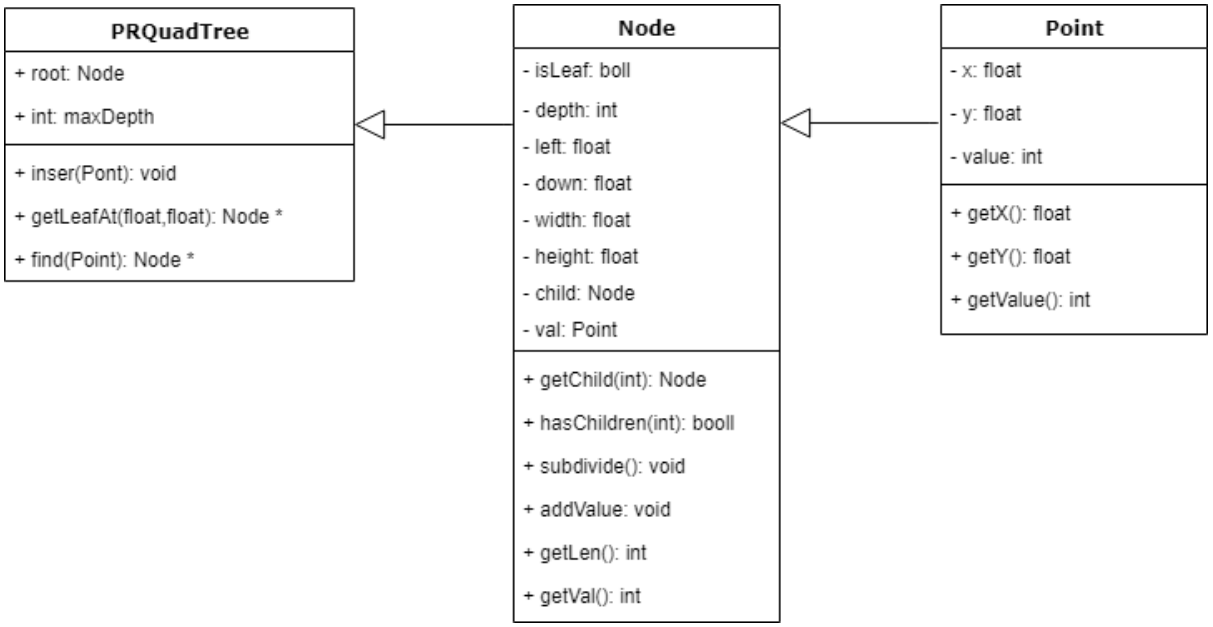

Figure 4: Diagrama de classes.

A principal característica da QuadTree é sua decomposição recursiva, essa metodologia de dividir para conquistar permite que a busca de um determinado ponto aconteça em $O(\log_4 n)$, e o tempo médio necessário para construir uma QuadTree a partir de pontos aleatórios seja $O(N\log_4 N)$.

References

[1] Antonio Carlos de Oliveira Miranda and Luciano Falcão da Silva. Quadtrees e aplicações à geração de malhas.

[2] Anthony D'Angelo. A brief introduction to quadtrees and their applications. *CCCG*, 2016.

[3] Claudio Esperança. Estruturas de dados espaciais.

[4] Yuping Lu. Quadtree. *CS 594 - Graph Theory*, 2014.

[5] Afonso Paiva Neto, Fernando Alvez Mazzini, and José Luiz Soares Luz. Quadtree.

[6] OpenDSA. The pr quadtree.

[7] Hanan Samet. The quadtree and related hierarchical data structures. *ACM Comput*, 1984.

[8] Bárbara Lopes Voorsluys and William Voorsluys. Quadtree.