

# Aplicativo Didático para a Árvore Binária Range Tree

João Paulo Ribeiro Erthal<sup>1</sup>, Thiago Gomes Carneiro<sup>1</sup>, Guilherme L. A. Mota<sup>1</sup>,  
Paulo E. D. Pinto<sup>1</sup>

<sup>1</sup>Instituto de Matemática e Estatística  
Universidade do Estado do Rio de Janeiro (UERJ)

joaopauloerthal@gmail.com, carneirotg@gmail.com, {guimota,  
pauloedp}@ime.uerj.br

**Abstract.** *This paper presents an e-learning tool dedicated to the multidimensional Range Tree (MRT) data structure. The most important application of MRT is the search for points. The herein presented e-learnig software was developed in C++ language, in the context of Qt IDE and with support of the Graphviz library. In the experiments, the quest for points was accessed using a database of municipalities in Brazil. Results show the efficiency of MRT in applications where the set of points is invariable. The developed application allows the selection of different shapes of regions around coordinates defined by the user. The outcome can be seen in terms of both bidimensional space and tree nodes.*

**Resumo.** *Este trabalho apresenta uma ferramenta de ensino dedicada às range trees multidimensionais, cuja aplicação mais importante é a busca por pontos. A ferramenta de e-learning ora apresentada foi desenvolvida em C++ a partir da IDE Qt e utilizando a biblioteca Graphviz. Nos experimentos, a busca por pontos foi avaliada com uso de uma base de dados de municípios do Brasil. Os resultados corroboram a eficiência desta estrutura em buscas onde o conjunto de pontos é invariante. A aplicação desenvolvida permite a busca em regiões com formas diversas. Os resultados podem ser vistos em termos do espaço bidimensional e em termos dos nós da estrutura.*

## 1. Introdução

Dados multidimensionais estão presentes em muitas aplicações, especialmente nos SIG (Sistemas de Informações Geográficas). Usualmente, estes dados representam coordenadas na reta, plano ou espaço mas podem representar também outros tipos de dimensões ou atributos. Existem muitas maneiras de representarmos dados multidimensionais, sendo a estrutura de dados Range Tree uma dessas maneiras. Esta, bem como inúmeras outras estruturas de dados desse grupo, podem ser encontradas em [Bentley 1979], [Samet 1990], [Samet 2006], [Stefanov 2012] e [Neyer 2012].

Visando facilitar a compreensão dos algoritmos, foi desenvolvida uma aplicação na linguagem C++ que mostra a criação da árvore de acordo com os pontos inseridos, a busca por pontos em intervalo determinado e a remoção de pontos contidos nos nós folhas da árvore, de acordo com as informações inseridas pelo usuário no sistema.

## 2. Fundamentação Teórica da Range Tree

### 2.1 Conceitos

A RangeTree é um tipo de árvore binária de busca balanceada (abbb) apresentando outras características específicas, a saber:

- todos os pontos inseridos pelo usuário encontram-se nos nós folhas da árvore e, os nós não folhas são criados a partir deles, de modo a habilitar a discriminação entre subárvores à esquerda ou à direita;
- os nós folhas encontram-se ligados em uma lista duplamente encadeada.

A RangeTree pode representar dados em  $k$  dimensões. Neste trabalho, será desenvolvida uma Range Tree para dados bidimensionais, possibilitando, por exemplo, representação de posições geográficas. A primeira dimensão refere-se à coordenada  $x$  (abscissa) e a segunda, à coordenada  $y$  (ordenada).

A RangeTree multidimensional permite buscas rápidas, com complexidade em tempo de  $O(\log N + F)$ , onde  $F$  é o número de pontos encontrados na busca, e requer, para construção, tempo de  $O(N \cdot \log^{d-1} N)$ , sendo  $d$  o número de dimensões. Com relação ao armazenamento, é necessário espaço  $O(N \cdot \log N)$ .

Inicialmente, a árvore é construída em uma dimensão, ordenada pela coordenada  $x$ . Em seguida, cada nó não folha aponta para uma Range Tree unidimensional, ordenada pelo valor da coordenada  $y$ .

### 2.2. Estrutura

Todos os dados inseridos na Range Tree encontram-se nas folhas da árvore. Cada nó da Range Tree bidimensional é composto pelos seguintes campos: LEFT, RIGHT, NEXT, PREVIOUS, MIDRANGE, ROOT2D, X, Y e NAME, conforme mostra a Figura 1. Os campos LEFT(P) e RIGHT(P) denotam o filho à esquerda e à direita, respectivamente, do nó não-folha P (serão nulos no caso de P ser um nó folha). Os campos NEXT(P) E PREVIOUS(P) denotam o nó anterior e o nó posterior da folha P (serão nulos no caso de nós não folhas). O campo MIDRANGE(P) denota o nó pai de P (será nulo no caso da raiz da árvore). O campo ROOT2D(P) é o ponteiro para a raiz da Range Tree unidimensional, associada ao nó P (não folha). Os campos NAME, X e Y são os valores dos registros armazenados nos pontos inseridos na árvore. Para os nós não folhas, X e Y servem como discriminadores entre as subárvores à esquerda e à direita, isto é, são maiores ou iguais aos valores armazenados na subárvore à esquerda e menores ou iguais aos valores armazenados na subárvore à direita.

LEFT	RIGHT	PREVIOUS	NEXT	MIDRANGE	ROOT2D	NAME	X	Y
Ponteiro para a subárvore a esquerda	Ponteiro para a subárvore a direita	Ponteiro para o nó anterior	Ponteiro para o próximo nó	Ponteiro para o nó pai	Ponteiro para a raiz da Range Tree associada	Identificador do nó	Valor da coordenada X do nó	Valor da coordenada Y do nó

Figura 1. Representação da estrutura dos nós da Range Tree bidimensional

### 2.3. Funcionamento

A Range Tree é melhor compreendida se primeiro entendermos o problema de busca em um intervalo unidimensional. Como foi dito anteriormente, uma Range Tree multidimensional é uma árvore binária de busca balanceada onde os pontos de dados estão armazenados nos nós folhas e os nós folhas são ligados, na ordem de classificação, pelo uso de uma lista duplamente encadeada. Os nós que não são folhas contêm valores intermediários que habilitam a discriminação entre as subárvores à esquerda e à direita. O problema da busca é resolvido pelo fato da estrutura ser uma árvore balanceada, permitindo que alguns nós não relevantes sejam excluídos da busca e, também, pelo uso da lista duplamente encadeada, permitindo que menos nós sejam percorridos durante a busca, pois não há necessidade de visitar os nós pais dos nós folhas.

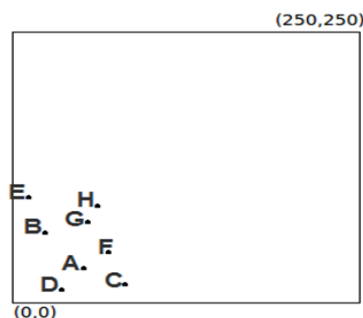
Cada nó que não é folha, além de habilitar a discriminação entre as subárvores, possui um ponteiro para uma Range Tree unidimensional dos pontos contidos nas suas subárvores à esquerda e à direita, porém ordenados pelo valor da coordenada  $y$ . Assim, a RangeTree é uma árvore binária de árvores binárias. A busca por pontos em intervalos utiliza essa estrutura, fazendo primeiramente a busca pela coordenada  $x$ , a partir da raiz da árvore e, posteriormente, a partir dos pontos da coordenada  $x$  encontrados, é feita a busca pelos pontos ordenados pela coordenada  $y$ .

A Figura 2 representa um exemplo de entrada de dados do usuário, informando nome, coordenada  $x$  e coordenada  $y$  que deverão ser inseridos na lista duplamente encadeada que, posteriormente, dará origem à árvore.

NOME	X	Y
A	50	30
B	22	63
C	80	15
D	34	10
E	10	96
F	68	45
G	53	73
H	60	88

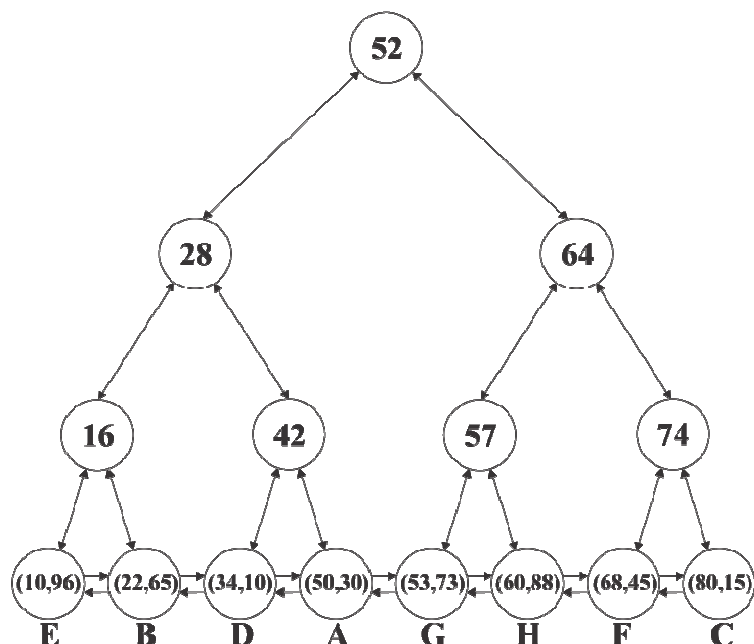
**Figura 2.** Lista de pontos no plano que serão inseridos na Range Tree

A Figura 3 mostra a distribuição, no espaço bidimensional, dos pontos definidos na Figura 2.



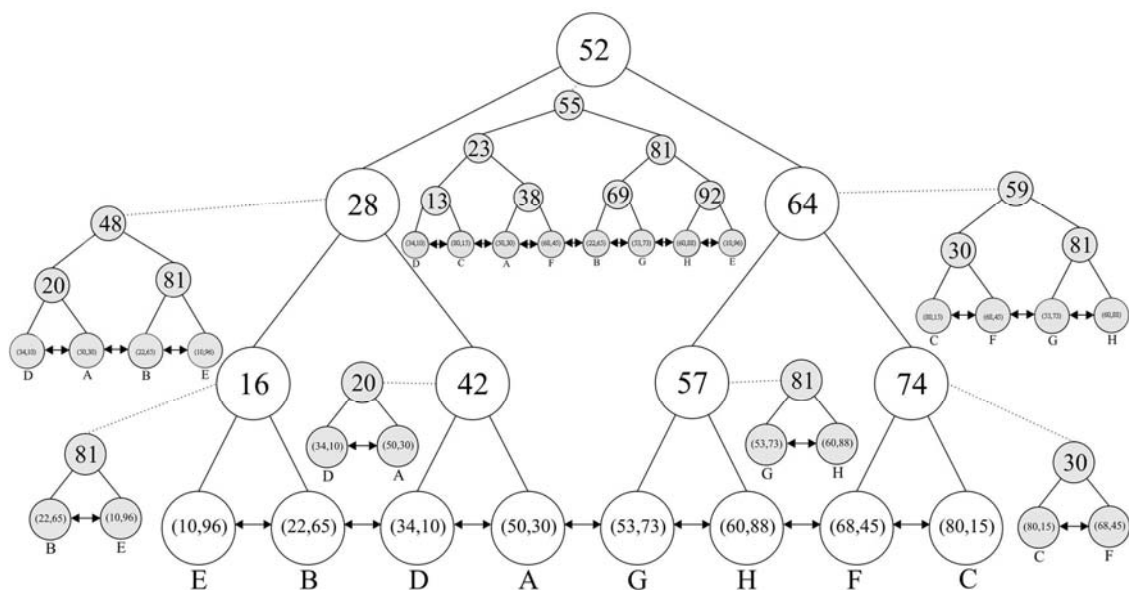
**Figura 3.** Pontos da Figura 2.2 representados no plano

A Figura 4 mostra a Range Tree unidimensional gerada a partir dos pontos listados na Figura 2.



**Figura 4. Range Tree unidimensional**

A Figura 5 mostra a Range Tree bidimensional criada a partir dos pontos inseridos na Figura 2. Os nós na cor branca representam a primeira dimensão, com os nós folhas ordenados pela coordenada  $x$  e os pontos cinza representam a segunda dimensão, com os nós folhas ordenados pela coordenada  $y$ .



**Figura 5. Range Tree bidimensional**

## 2.4. Inclusão de dados

A Range Tree é uma estrutura estática de dados, ou seja, é criada após a inserção de um número de pontos. Cada ponto informado pelo usuário é incluído de forma ordenada pelo valor da coordenada  $x$  em uma lista duplamente encadeada. Ao final da criação da lista, para garantir que a árvore seja balanceada, é verificada a quantidade de nós existente na lista.

Após a criação da lista duplamente encadeada, é gerada a Range Tree de maneira bottom-up, criando, para cada dois nós, um novo pai, nível a nível, até que só reste um nó, que será a raiz da Range Tree. Cada nó não folha (incluindo a raiz), tem seu valor calculado pela média entre o valor do maior nó folha descendente do filho à esquerda e o valor do menor nó folha descendente do filho à direita.

Durante o processo de criação da árvore para a primeira dimensão, logo após a criação de cada nó não folha, é criada a Range Tree bidimensional a ele associada, dos pontos contidos nas subárvores à esquerda e à direita do nó.

Como a Range Tree deve ser balanceada, neste trabalho foi tomada a decisão de projeto de a Range Tree sempre ser uma árvore cheia. Assim, quando a quantidade de nós folhas não for potência de 2, o último nó será replicado com nome “NULL”, de maneira que a árvore sempre seja cheia e as buscas não sejam comprometidas. Caso a árvore não fosse cheia, não poderíamos garantir que ela seria balanceada, podendo acarretar a descaracterização desta estrutura.

## 2.5. Exclusão

Tal como na inclusão, a árvore é recriada após alguma exclusão feita.

## 2.6. Buscas

Serão considerados três tipos de busca: por nó, por região e por vizinhança (esta apenas para efeitos visuais, seguindo o padrão de interface proposto por [Dacoma 2012], uma vez que a Range Tree apresenta bons resultados para buscas em regiões ortogonais). Descreveremos a busca por região, sendo as demais buscas descritas posteriormente.

Para buscar pontos contidos em um intervalo na Range Tree, usou-se o algoritmo proposto originalmente por Bentley [Bentley 1979] e desenvolvido por Samet [Samet 2006]. O algoritmo recebe como parâmetro o intervalo de  $x$ ,  $[B_x:Ex]$ , que deve ser buscado, o intervalo de  $y$ ,  $[B_y:E_y]$  que deve ser buscado e a raiz  $T$  da árvore. Inicialmente, o algoritmo procura o menor nó ancestral (nó mais afastado possível da raiz comum aos limites buscados no intervalo. No pior caso, será a própria raiz.), comum a  $B_x$  e  $Ex$ . Se o nó ancestral for um nó folha, ele é testado se pertence aos intervalos em  $x$  e  $y$ . Se o nó ancestral não for nó folha, a busca continua nos seus nós filhos, nas suas Range Tree bidimensionais associadas. Caso os nós encontrados pertençam ao intervalo  $[B_y:E_y]$ , eles serão retornados. A busca nas Range Trees associadas é baseada nos nós ordenados em função da coordenada  $y$ .

### 3. Implementação da Interface

Com o intuito de auxiliar a compreensão da estrutura de dados Range Tree, conforme exposto anteriormente, foi desenvolvida uma interface para demonstração dos algoritmos de inserção, deleção e busca.

#### 3.1. Ferramentas Utilizadas

O desenvolvimento do Software Visualizador acompanhou a infraestrutura utilizada no desenvolvimento da Interface Luana [Dacome 2012].

Para o desenvolvimento do software de interface utilizado para representar graficamente a estrutura de dados Range Tree, foi escolhida a linguagem de programação C++ devido ao seu potencial em manipular ponteiros, gerenciar memória e suportar programação orientada a objetos. Foi utilizado o sistema multiplataforma Qt que suporta código em C++ e fornece uma grande quantidade de bibliotecas para a implementação de software com interface gráfica.

Para representar os nós de forma hierárquica, foi utilizada a biblioteca para gerar grafos Graphviz [<http://www.graphviz.org>], cujos gráficos a serem gerados foram codificados utilizando a linguagem de descrição de grafos DOT.

##### 3.1.1. Graphviz

O nome Graphviz vem da abreviação de *GraphVisualization Software* (Software de Visualização de Grafos), é um pacote de ferramentas, de código aberto, desenvolvido nos laboratórios de pesquisa da AT&T para desenhar grafos especificados na linguagem DOT. O Graphviz possui algoritmos capazes de calcular automaticamente o melhor posicionamento dos nós e arestas de um grafo em uma imagem gerada no formato especificado. Para a implementação do software de interface, o Graphviz foi utilizado como uma biblioteca, adicionando ao projeto as bibliotecas libgvc.so e libgraph.so.

Os gráficos são gerados diretamente em memória principal, sem que nenhum arquivo de imagem seja criado em memória secundária para posterior abertura e exibição. Para que isso ocorra é gerado um array de char (char\*) contendo o código DOT do grafo a ser desenhado. Em seguida esse array é passado para o Graphviz para que seja renderizado. O Graphviz, então, entrega a imagem codificada em outro array de char.

##### 3.1.2. Qt

Qt [<http://qt-project.org/doc>] é um sistema multiplataforma que permite compilar aplicações para Windows, Mac, Linux ou outros sistemas Unix, suporta código em C++ e possui código aberto. Foi desenvolvido pela empresa norueguesa Trolltech, posteriormente adquirida pela Nokia. O Qt fornece uma grande quantidade de bibliotecas para implementação de aplicativos com interface gráfica. Um exemplo de grande sucesso da utilização do Qt é o projeto KDE.

Para exibir os gráficos do resultado do Plano Cartesiano e da Representação Hierárquica dos nós de uma Range Tree no software de interface, foram utilizadas algumas classes do Qt, sendo as mais relevantes: QGraphicsView, QGraphicsScene, QGraphicsItem, QByteArray e QPixmap.

Para gerar o gráfico do resultado do Plano Cartesiano, foram inseridos objetos da classe QGraphicsItem (QGraphicsRectItem para representar pontos) em um QGraphicsScene, e posteriormente este objeto QGraphicsScene é exibido em um QGraphicsView.

## **3.2. Projeto da Aplicação**

### **3.2.1. Requisitos**

Foram identificados os seguintes requisitos para a aplicação:

- A RangeTree deve ser gerada automaticamente, após o acionamento de um botão específico;
- Todas as imagens apresentadas em tela devem ser geradas na memória principal, com auxílio do Graphviz;
- Inserção, exclusão e busca de nós devem ser feitas a partir de dados inseridos pelo usuário;
- Deve ser possível apagar a árvore;
- Devem ser apresentados os resultados obtidos nas buscas;
- A cada atualização na estrutura da árvore, a tela deve ser atualizada;
- Deve haver interação com as imagens apresentadas na tela (como zoom, por exemplo);
- Deve ser mostrado o caminho percorrido durante os diferentes tipos de busca;
- Deve ser possível salvar uma árvore em arquivo;
- Deve ser possível carregar em memória uma árvore salva previamente em arquivo.

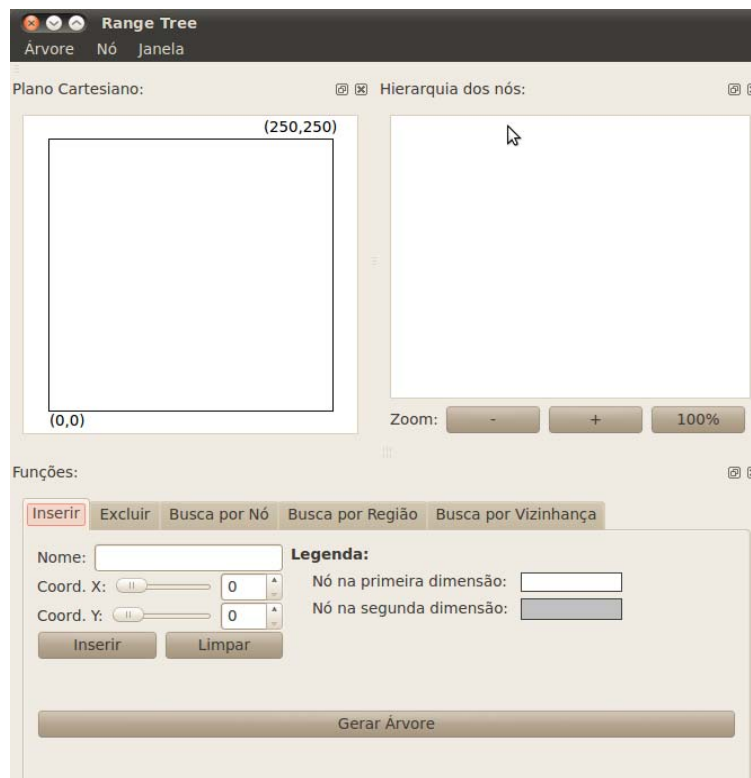
## **3.3. Interface**

Conforme mencionado anteriormente, foi desenvolvida uma interface, na linguagem C++ utilizando a IDE Qt. O resultado será apresentado a seguir, através de imagens, demonstrando as funcionalidades da interface e as operações da Range Tree. A Figura 6 apresenta a tela inicial do software.

### **3.3.1. Tela Inicial**

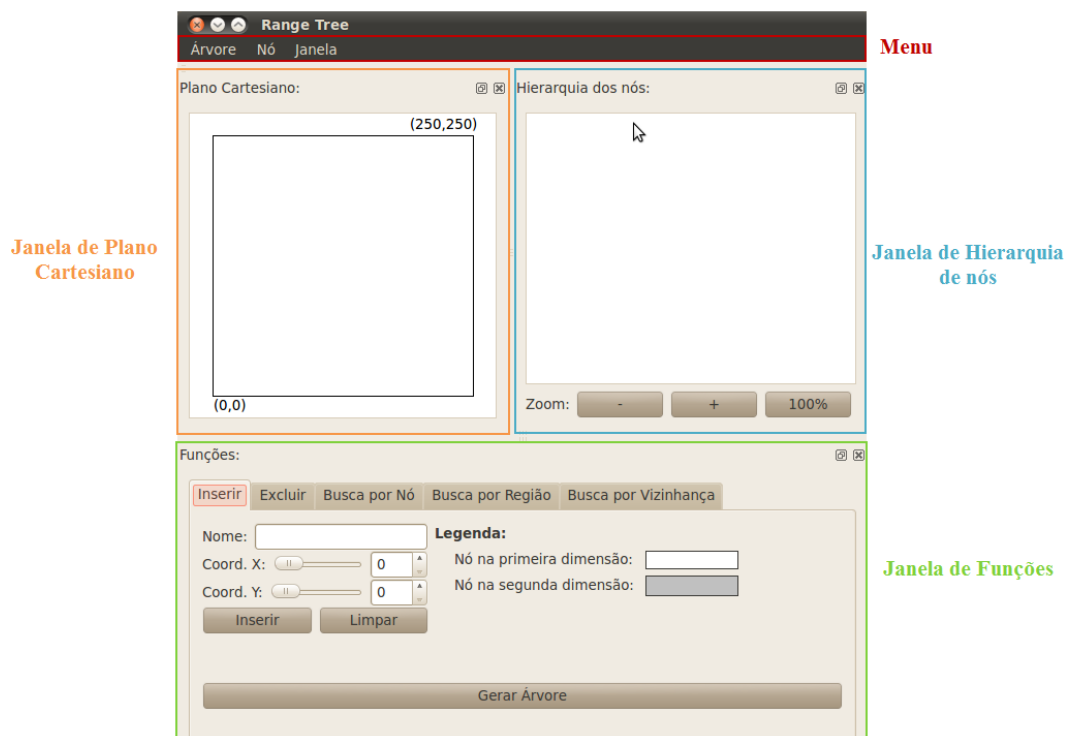
A tela inicial consiste nas seguintes partes:

- Menu
- Funções
- Janela de Plano Cartesiano
- Janela de Hierarquia dos Nós



**Figura 6. Tela inicial da Interface Range Tree**

As partes mencionadas acima são demonstradas visualmente na Figura 7, que exhibe como é feita a divisão da tela da interface.



**Figura 7. Divisão da Tela inicial da Interface Range Tree**



### 3.3.2. Menu

O software possui uma barra de menu para acesso a operações básicas de carga e salvamento em arquivos, opções de exibição, entre outros. A Figura 8 ilustra o menu.



Figura 8. Menu

### 3.3.3. Funções

A Figura 9 mostra a janela Funções, que permite a entrada de dados para atualização e consultas.

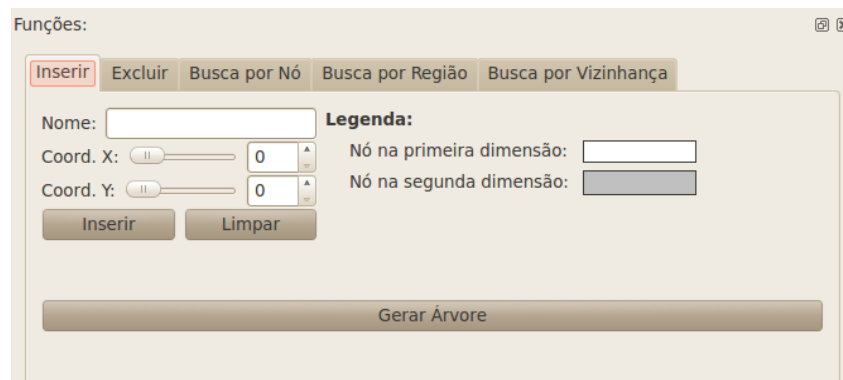


Figura 9. Janela Funções

Na inserção e exclusão, os dados são informados conforme mostra a Figura 9. Também no caso da busca por nó as informações são semelhantes.

No caso da busca por região, tem-se que informar os dados dos pontos extremos de um retângulo objeto da busca. A Figura 10 ilustra a aba correspondente. Ilustra também a janela Resultado, onde são exibidas as informações textuais resultantes da busca. Além das informações mostradas, é também traçado um retângulo na Janela de Plano Cartesiano exibida adiante, de modo que o usuário possa visualizar a região delimitada pelas abscissas  $x_1$  e  $x_2$ , e ordenadas  $y_1$  e  $y_2$ .

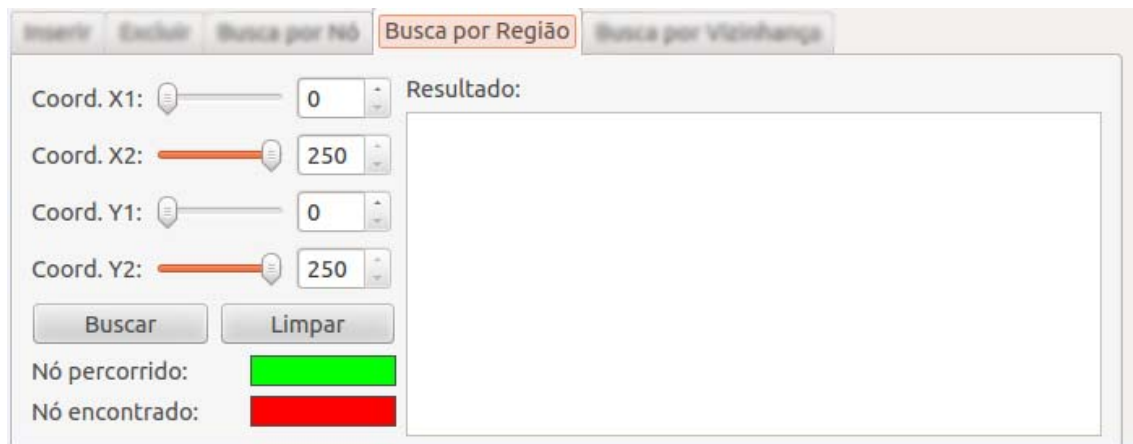


Figura 10. Aba Busca por Região

Na busca por vizinhança, é possível que o usuário busque todos os nós internos ao círculo de raio dado com centro nas coordenadas  $(x, y)$  informadas pelo usuário. A janela é mostrada na Figura 11. Além dos dados textuais a serem exibidos na janela Resultados, é também traçado um círculo na Janela de Plano Cartesiano mostrada adiante, de modo que o usuário possa visualizar a região delimitada pelo círculo definido.



Figura 11. Aba Busca por Vizinhança

### 3.3.4. Janela do Plano Cartesiano

A janela de Plano Cartesiano exibe os pontos inseridos pelo usuário no espaço bidimensional. Os pontos são apenas impressos na janela, de acordo com suas coordenadas  $x$  e  $y$ . A Figura 12 mostra a origem das coordenadas  $x$  e  $y$  (ponto  $(0,0)$ ). Ela ilustra também o resultado da inserção dos pontos da Figura 2.

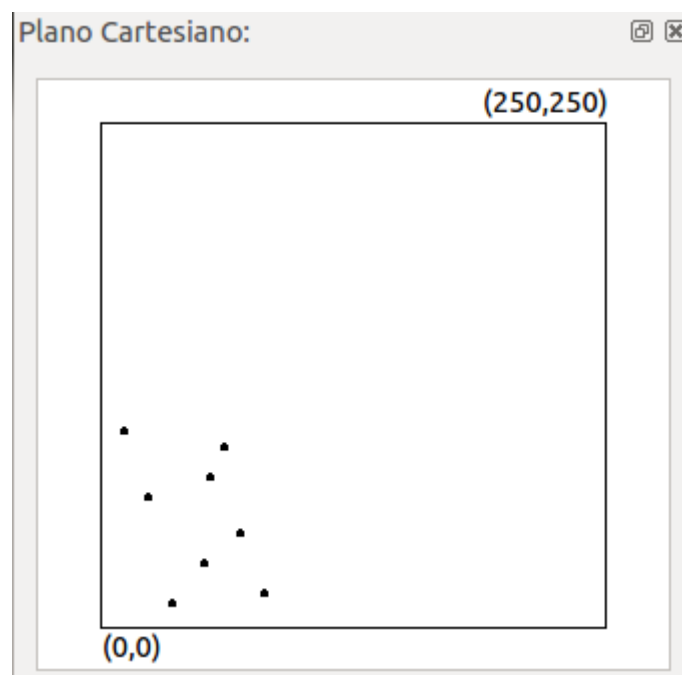


Figura 12. Janela de Plano Cartesiano com os dados da Figura 2

### 3.3.5. Janela de Hierarquia de Nós

A janela de Hierarquia dos Nós é exibida na Figura 13. Nela é mostrado o grafo da Range Tree gerada pelo usuário. No nível mais alto da Range Tree encontra-se o nó raiz. Cada nó não folha possui setas ligando-os aos seus filhos à esquerda e à direita. Os nós folhas encontram-se no nível mais baixo da Range Tree e representam os nós que foram inseridos pelo usuário.

Cada nó representado nesta janela é mostrado em formato elíptico e possui os valores das coordenadas  $x$  e  $y$  e o nome do nó, no caso de nós folhas da árvore. No caso de nós não folhas, são exibidos apenas os valores de  $x$  e  $y$ , que servem como discriminadores para os filhos à esquerda ou à direita. Os nós brancos referem-se à primeira dimensão e os nós cinzas, à segunda. A cor da elipse que representa o nó durante as buscas vai estar de acordo com a operação realizada. Os botões + e - servem para aumentar o zoom da imagem mostrada na janela e o botão 100% retorna o tamanho da imagem para o tamanho original. Esta janela, assim como a janela de Plano Cartesiano, pode ser movida e ocultada de acordo com a vontade do usuário.

A Figura 13 apresenta o resultado da inclusão dos pontos A (50,30) e B (22,63).

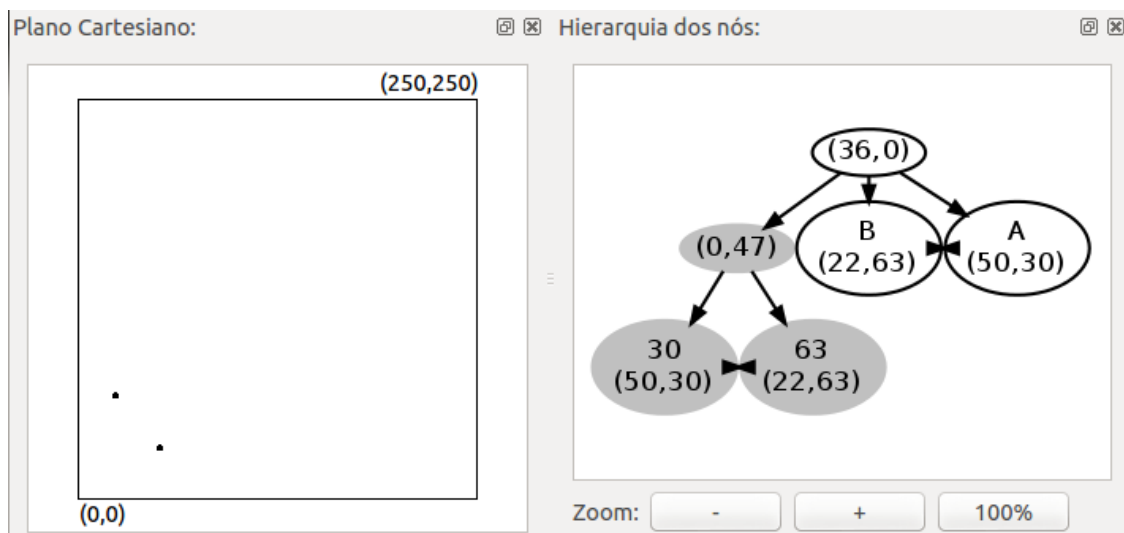
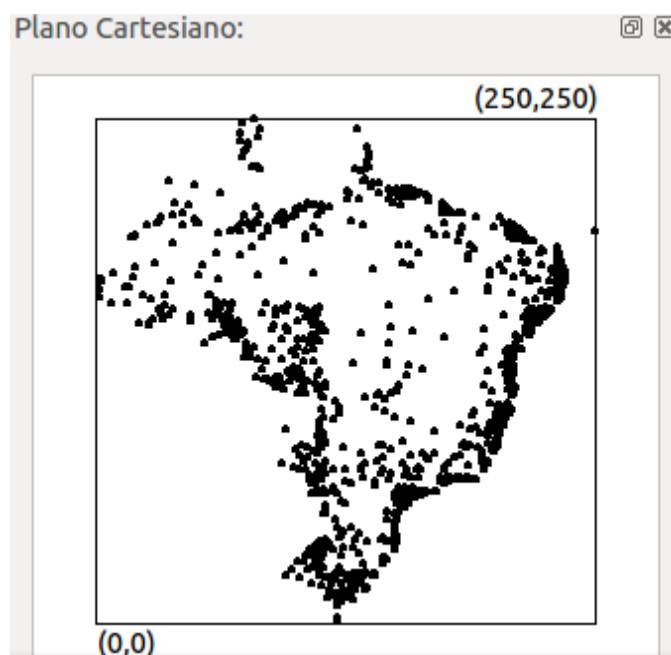


Figura 13. Janela de hierarquia dos nós após a inclusão dos nós A e B da Figura 2

### 3.4. Exemplos

Nos exemplos a seguir serão exibidas as telas da criação da Range Tree a partir do arquivo `brasil.range`, que contém as coordenadas dos municípios brasileiros. A Figura 14 mostra a tela da interface após a abertura do arquivo `brasil.range`.



**Figura 14. Janela de Plano Cartesiano após o carregamento do arquivo brasil.range**

Nesta figura, é possível observar a disposição dos municípios na janela do Plano Cartesiano. A árvore será gerada com o clique do botão Gerar Árvore, na aba Inserir. A geração da árvore fará com que a imagem da Range Tree seja gerada pelo Graphviz na janela Hierarquia de Nós.

A geração da imagem utilizando o *roadmap* (*mapa que detalha nós e vértices dos grafos*) “dot” possui uma recomendação de não ultrapassar 1000 nós em um grafo, visto que este *roadmap* é o mais complexo de prever sua estrutura final. Também é possível observar um erro de falta de memória caso sejam utilizados muitos nós para geração da imagem. Existem ainda outros *roadmaps* possíveis para geração de imagens como, por exemplo, “sfdp” e “twopi” que possuem uma complexidade menor, conseguindo assim gerar com mais rapidez e eficiência a imagem final, porém estes *roadmaps* não são hierárquicos.

Na Figura 15 mostra-se um exemplo de busca, a busca por região no mapa dos municípios brasileiros compreendidos entre o intervalo  $X1:130, X2:180, Y1:100, Y2:150$ .

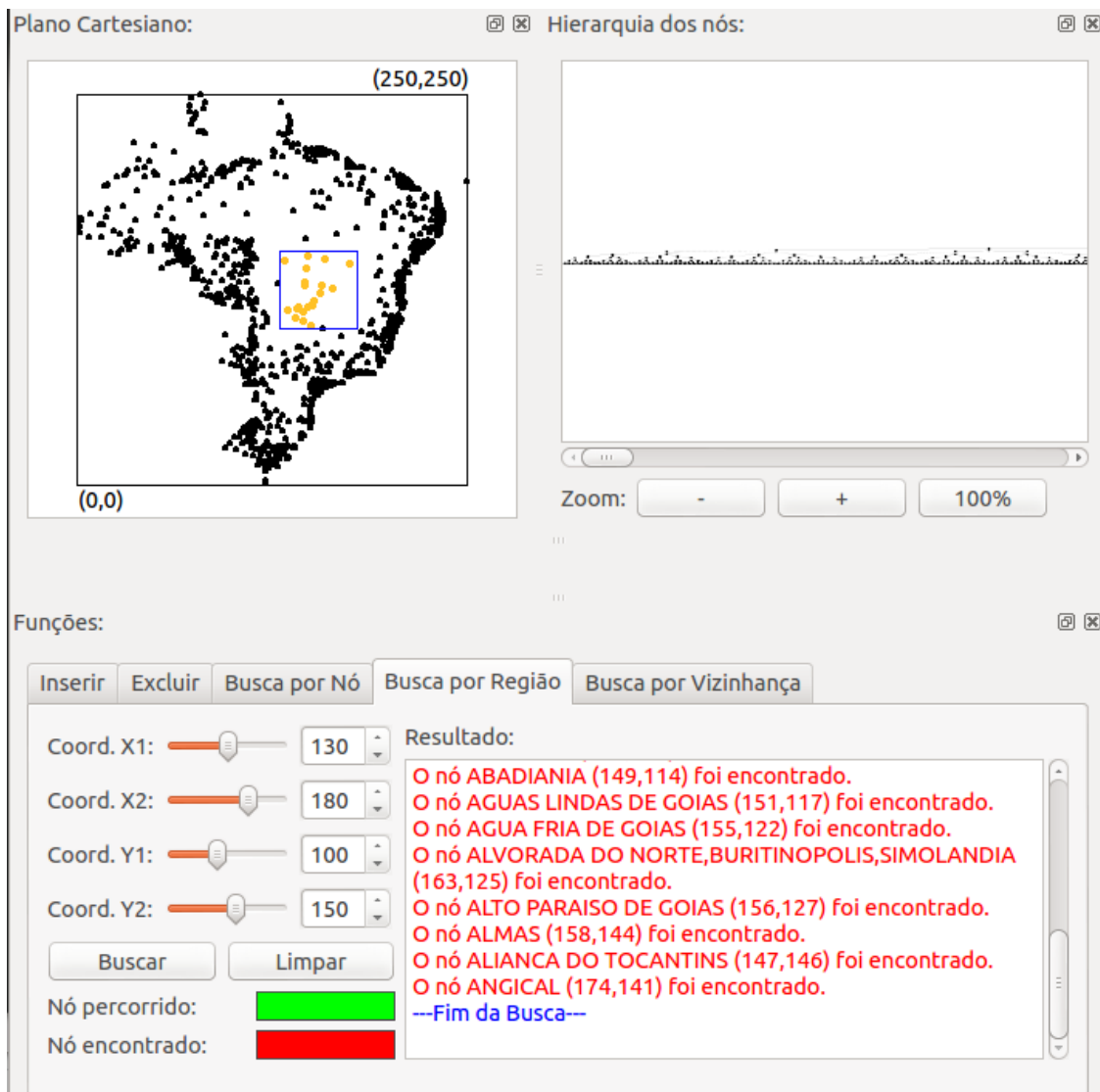


Figura 15. Busca pelos municípios contidos no intervalo X1:130, X2:180, Y1:100, Y2:150

#### 4. Conclusão

Este trabalho apresentou um software aplicativo para o ensino da estrutura de dados multidimensional Range Tree. A implementação é capaz de manipular dados em duas dimensões. A aplicação foi desenvolvida utilizando a linguagem C++ e a ferramenta Qt, com auxílio da biblioteca Graphviz e da linguagem DOT.

Este trabalho pretende contribuir com uma linha de pesquisa sobre estruturas de dados multidimensionais criada no IME/DICC pelos professores Guilherme Mota e Paulo Eustáquio, coautores deste trabalho. Devido à extensão do assunto, esta linha de pesquisas se estende a outros trabalhos, em razão da existência de várias estruturas multidimensionais distintas, não abordadas aqui, tais como a Kd-Tree ou a Quadtree.

Com o desenvolvimento das outras estruturas, será possível realizar, na linha de pesquisa, comparações entre estruturas, em diversas categorias, como por exemplo, desempenho e economia de memória.

## 5. Referências Bibliográficas

- Bentley, J.L. e Friedman, H.F. (1979) Data Structures for Range Searching. *Carnegte-Mellon Unwerslty. EUA*.
- Dacome, R. (2012) Uma Aplicação Didática da Árvore Binária Kd-Tree. Monografia de conclusão de Graduação. UERJ. Rio de Janeiro.
- <http://www.graphviz.org>, visitado em novembro 2012.
- <http://qt-project.org/doc/>, visitado em novembro 2012.
- Neyer, G. (2012) [http://www.cgal.org/Manual/latest/doc\\_html/cgal\\_manual/](http://www.cgal.org/Manual/latest/doc_html/cgal_manual/SearchStructures/Chapter_main.html) Search Structures / Chapter\_main.html, visitado em novembro 2012
- Samet, H. (2006) Foundations of Multidimensional and Metric Data Structures, Morgan Kaufmann.
- Samet, H. (1990) The Design and Analysis of Spatial Data Structures, Addison\_Wesley.
- Stefanov, E. (2012) <http://www.emilstefanov.net/Projects/RangeSearchTree.aspx>, visitado em novembro 2012.