Tópicos Avançados em Programação

Humberto Longo

Instituto de Informática
Universidade Federal de Goiás

Bacharelado em Ciência da Computação, 2016

Conceitos

k-D Tree

- Bentley, J. L. *Multidimensional binary search trees used for associative searching*. Communications of the ACM; 18(9):509–517; 1975.
- Modificação de árvore binária de busca que permite armazenamento eficiente de dados multidimensionais.
- Estrutura de dados eficiente para a contagem do número de pontos que caem dentro de um determinado subespaço k-dimensional.
  - Elementos armazenados em uma *k-d tree* são associados a uma coordenada do ponto.
- Utilizada no processamento de imagens, como um meio de particionamento de pontos em um espaço k-dimensional.
- Aplicação em processamento de imagem: objetos posicionados em uma cena, traçado de raios, ...

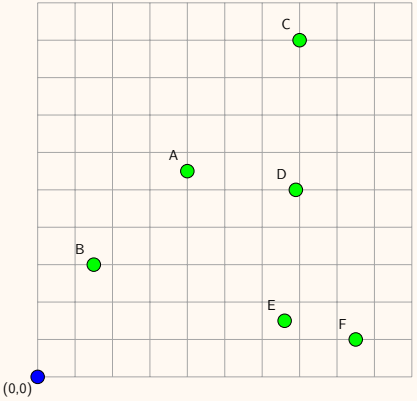
k-D Tree

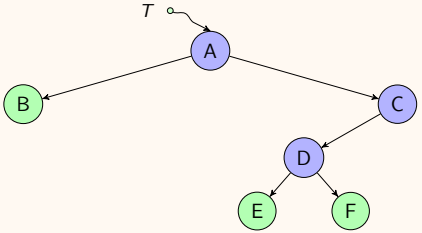
- Cada nível da árvore é associado com um discriminador específico.
  - Discriminador: dado usado para as decisões de ramificação (coordenada x ou y no caso 2-dimensional).
- Discriminador em cada nível de uma k-*d tree*:

  $0, k, 2{\cdot}k, \ldots:$ nós de busca ordenados na 1ª coordenada.
  $1, k+1, 2{\cdot}k+1, \ldots:$ nós de busca ordenados na 2ª coordenada.
  $\vdots$
  $j, k+j, 2{\cdot}k+j, \ldots:$ nós de busca ordenados na $(k+1)$-ésima coordenada.
- Embora o k refira-se à dimensão, é comum o uso de termos como "3-dimensional k-*d tree*" no lugar de "3-*d tree*".

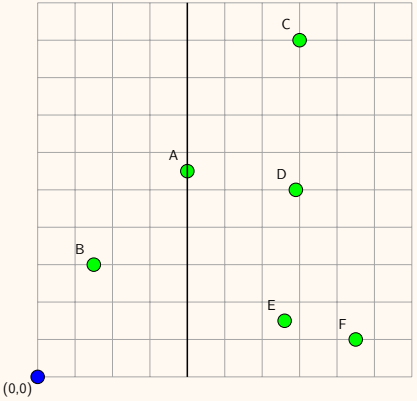
k-D Tree

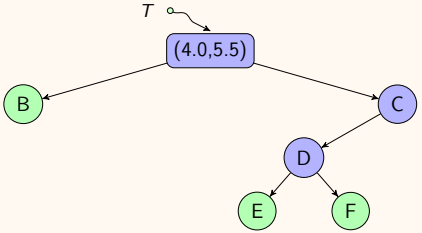
Exemplo 1.1



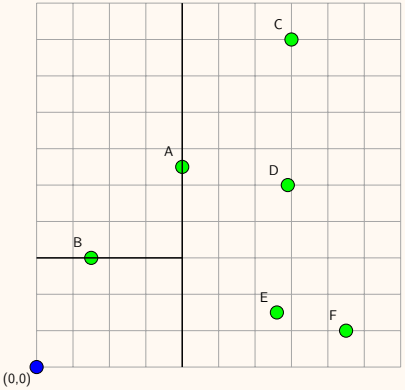


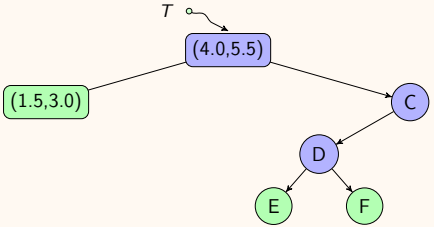
Exemplo 1.1



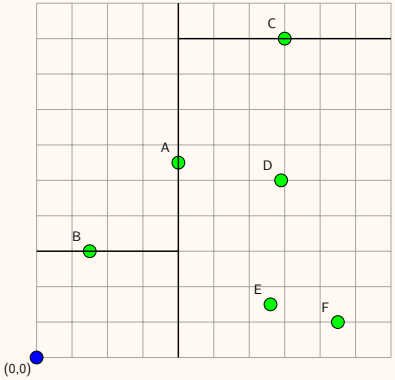


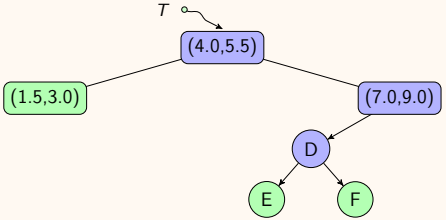
Exemplo 1.1



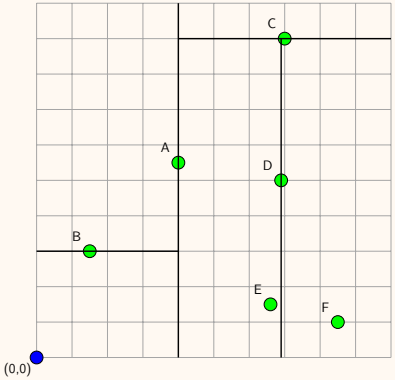


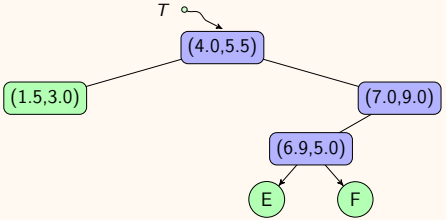
Exemplo 1.1



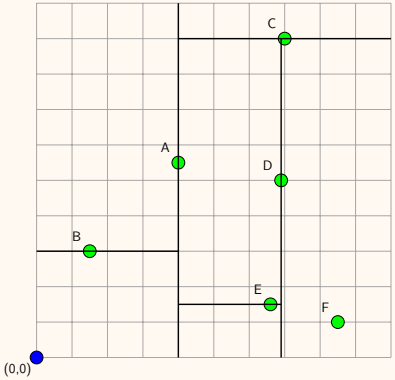


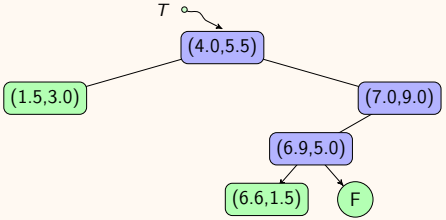
Exemplo 1.1



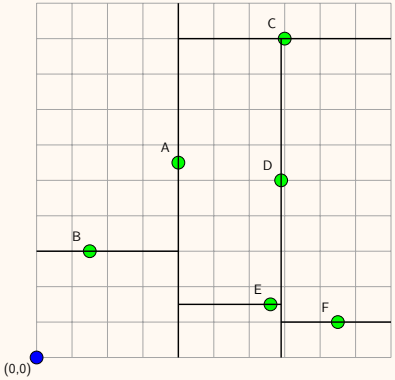


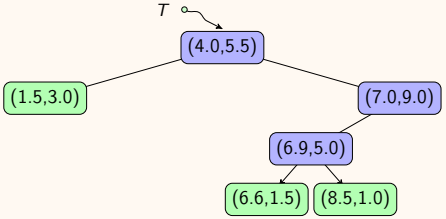
Exemplo 1.1





Exemplo 1.1





Busca de um elemento

- Busca de um elemento associado a $P = (6.9, 5.0)$:
  1. O discriminador da raiz ($A = (4.0, 5.5)$) é o x.
  2. Como $P.x > A.x$, a busca continua na subárvore à direita de A (raiz é $C = (7.0, 9.0)$).
  3. O discriminador do nó C é o y.
  4. Como $P.y < C.y$, a busca continua na subárvore à esquerda de C (raiz é $D = (6.9, 5.0)$).
  5. Dado que $P = D$, a busca é encerrada com sucesso!

Construção

- Dado um conjunto $\mathcal{C}$ de pontos, como construir uma k-*d tree* balanceada?
- Processo com duas etapas básicas:
  1. Usar a mediana relativa a uma coordenada para particionar o conjunto de pontos.
     - Vamos supor que não há duplicidade, mas se ocorrer, pode-se escolher qualquer uma das medianas.
  2. Associar o ponto relativo à mediana a um nó da árvore.

Construção

1. Ordenar os pontos em $\mathcal{C}$ com base na primeira coordenada e encontrar a mediana m.
2. Associar o ponto relativo à mediana m à raiz da *k-d-tree*.
3. Particionar o conjunto $\mathcal{C} - \{m\}$ em dois subconjuntos $\mathcal{C}_1$ e $\mathcal{C}_2$.
   - Pontos com a primeira coordenada menor ou maior do que a primeira coordenada de m, respectivamente.
4. Ordenar os pontos nas partições $\mathcal{C}_1$ e $\mathcal{C}_2$ em relação à segunda coordenada.
5. A mediana da primeira partição definirá a raiz da subárvore à esquerda e a mediana da segunda partição definirá a raiz da subárvore à direita.
6. Repetir esse processo até que cada subconjunto contenha apenas 1 ponto.

Construção

- Dado um conjunto $\mathcal{C}$ de pontos, como construir uma k-*d tree* balanceada?

$\mathcal{C} = \{$ (0.3, 9.0), (3.7, 0.4), (5.6, 7.8), (0.1, 4.8), (4.1, 8.9), (9.5, 0.7), (9.7, 0.9), (5.4, 6.5), (0.4, 6.1), (7.3, 6.9), (4.6, 5.8), (0.8, 8.9), (0.4, 4.1), (9.4, 0.2), (3.3, 0.7), (5.5, 5.4), (0.6, 0.5), (0.4, 0.6), (7.4, 9.7), (2.9, 1.5), (0.5, 8.8), (2.3, 2.3), (5.5, 0.2), (0.2, 9.7), (0.5, 0.7), (0.6, 2.8), (0.9, 5.5), (0.2, 9.1), (0.5, 9.7), (6.8, 4.2), (9.7, 1.8) $\}$

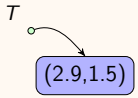
Construção

1. Ordenar os pontos em $\mathcal{C}$ com base na primeira coordenada e encontrar a mediana m:

$\mathcal{C} = \{$ (0.1, 4.8), (0.2, 9.1), (0.2, 9.7), (0.3, 9.0), (0.4, 0.6),
(0.4, 4.1), (0.4, 6.1), (0.5, 0.7), (0.5, 8.8), (0.5, 9.7),
(0.6, 0.5), (0.6, 2.8), (0.8, 8.9), (0.9, 5.5), (2.3, 2.3),
(2.9, 1.5),
(3.3, 0.7), (3.7, 0.4), (4.1, 8.9), (4.6, 5.8), (5.4, 6.5),
(5.5, 0.2), (5.5, 5.4), (5.6, 7.8), (6.8, 4.2), (7.3, 6.9),
(7.4, 9.7), (9.4, 0.2), (9.5, 0.7), (9.7, 0.9), (9.7, 1.8) $\}$

Construção

3. Associar o ponto relativo à mediana m à raiz da k-*d-tree*:



Construção

4. Particionar o conjunto $\mathcal{C} - \{m\}$ em dois subconjuntos $\mathcal{C}_1$ e $\mathcal{C}_2$:

$\mathcal{C}_1 = \{$ (0.1, 4.8), (0.2, 9.1), (0.2, 9.7), (0.3, 9.0), (0.4, 0.6),
(0.4, 4.1), (0.4, 6.1), (0.5, 0.7), (0.5, 8.8), (0.5, 9.7),
(0.6, 0.5), (0.6, 2.8), (0.8, 8.9), (0.9, 5.5), (2.3, 2.3) $\}$

$\mathcal{C}_2 = \{$ (3.3, 0.7), (3.7, 0.4), (4.1, 8.9), (4.6, 5.8), (5.4, 6.5),
(5.5, 0.2), (5.5, 5.4), (5.6, 7.8), (6.8, 4.2), (7.3, 6.9),
(7.4, 9.7), (9.4, 0.2), (9.5, 0.7), (9.7, 0.9), (9.7, 1.8) $\}$

Construção

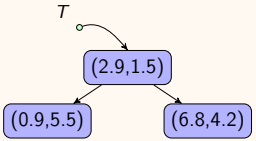
5. Ordenar os pontos nas partições $\mathcal{C}_1$ e $\mathcal{C}_2$ em relação à segunda coordenada:

$\mathcal{C}_1 = \{$ (0.6, 0.5), (0.4, 0.6), (0.5, 0.7), (2.3, 2.3),
(0.6, 2.8), (0.4, 4.1), (0.1, 4.8),
(0.9, 5.5),
(0.4, 6.1), (0.3, 9.0), (0.2, 9.1), (0.2, 9.7),
(0.5, 8.8), (0.8, 8.9), (0.5, 9.7) $\}$

$\mathcal{C}_2 = \{$ (5.5, 0.2), (9.4, 0.2), (3.7, 0.4), (3.3, 0.7),
(9.5, 0.7), (9.7, 0.9), (9.7, 1.8),
(6.8, 4.2),
(5.5, 5.4), (4.6, 5.8), (5.4, 6.5), (7.3, 6.9),
(5.6, 7.8), (4.1, 8.9), (7.4, 9.7) $\}$

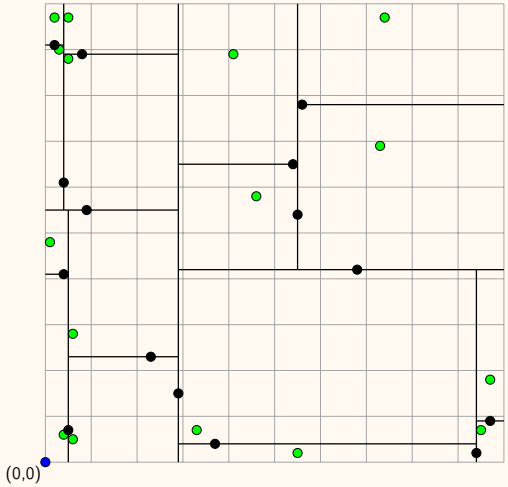
Construção

6. A mediana da primeira partição definirá a raiz da subárvore à esquerda e a mediana da segunda partição definirá a raiz da subárvore à direita:



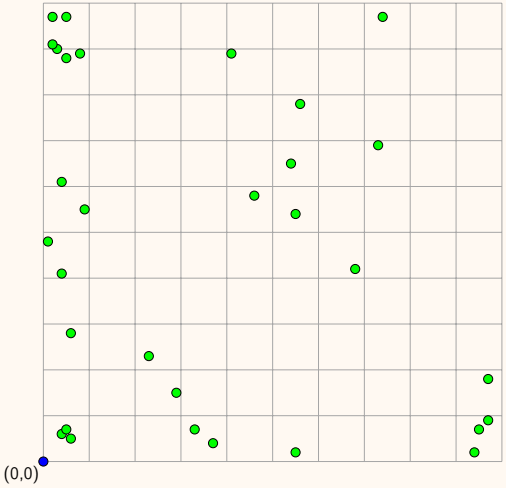
Construção

7. Repetir esse processo até que cada subconjunto contenha apenas 1 ponto:



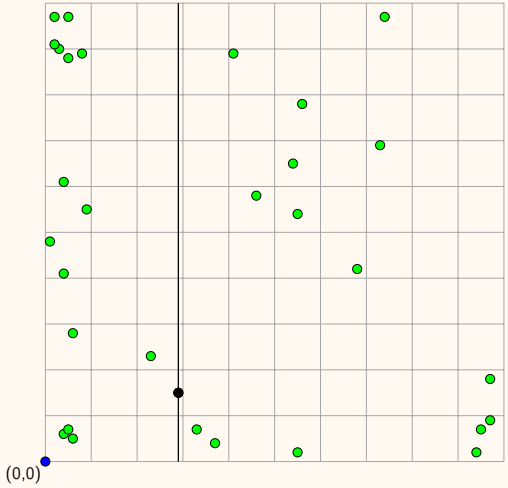
Exemplo 1.2 (Construção)

1. Usar a mediana relativa a uma coordenada para particionar o conjunto de pontos:



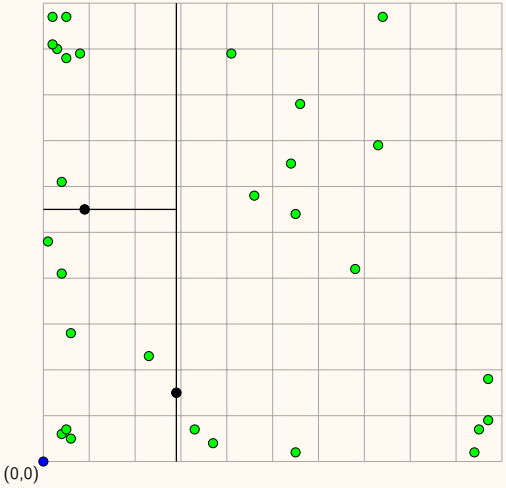
Exemplo 1.2 (Construção)

1. Usar a mediana relativa a uma coordenada para particionar o conjunto de pontos:



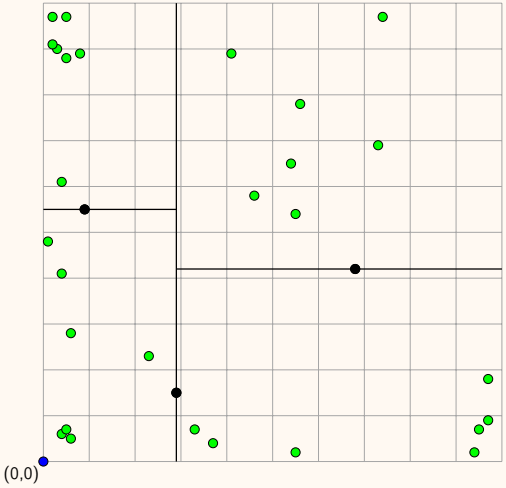
Exemplo 1.2 (Construção)

1. Usar a mediana relativa a uma coordenada para particionar o conjunto de pontos:



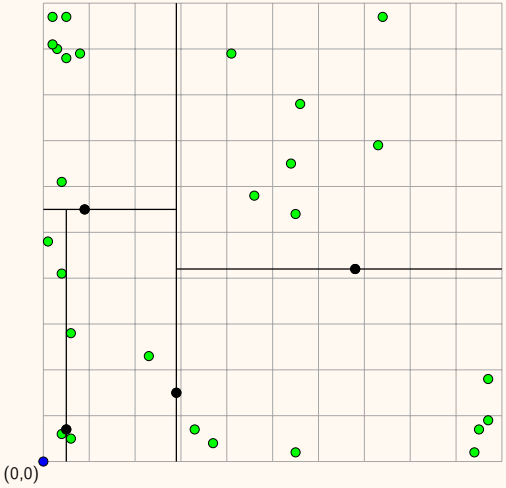
Exemplo 1.2 (Construção)

1. Usar a mediana relativa a uma coordenada para particionar o conjunto de pontos:



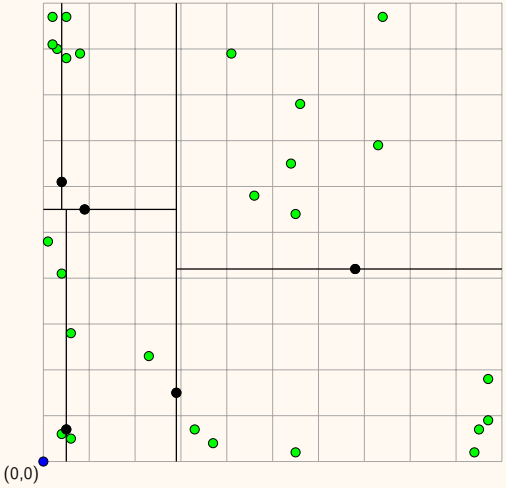
Exemplo 1.2 (Construção)

1. Usar a mediana relativa a uma coordenada para particionar o conjunto de pontos:



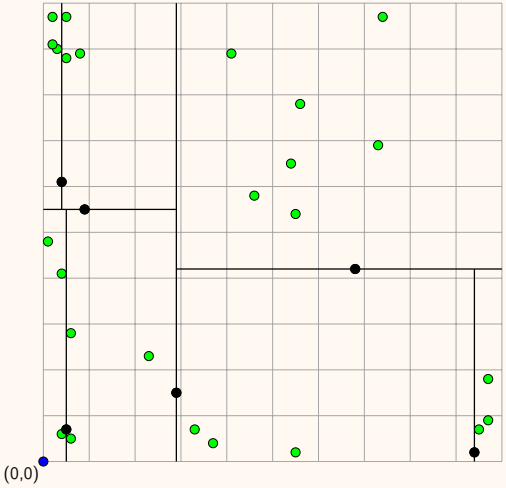
Exemplo 1.2 (Construção)

1. Usar a mediana relativa a uma coordenada para particionar o conjunto de pontos:



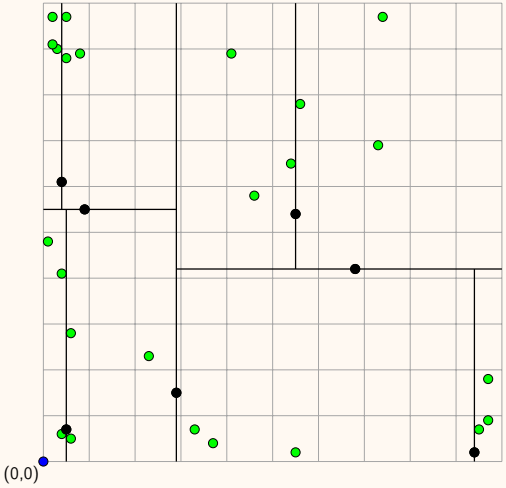
Exemplo 1.2 (Construção)

1. Usar a mediana relativa a uma coordenada para particionar o conjunto de pontos:



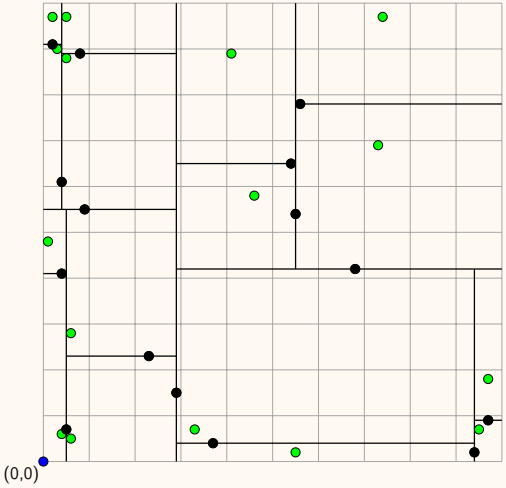
Exemplo 1.2 (Construção)

1. Usar a mediana relativa a uma coordenada para particionar o conjunto de pontos:



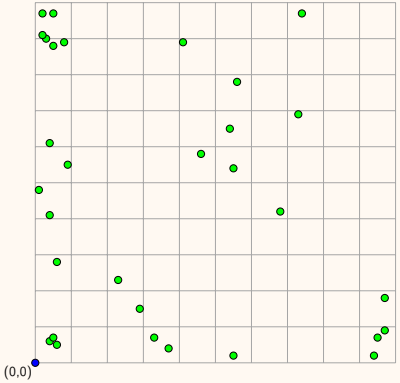
Exemplo 1.2 (Construção)

1. Usar a mediana relativa a uma coordenada para particionar o conjunto de pontos:



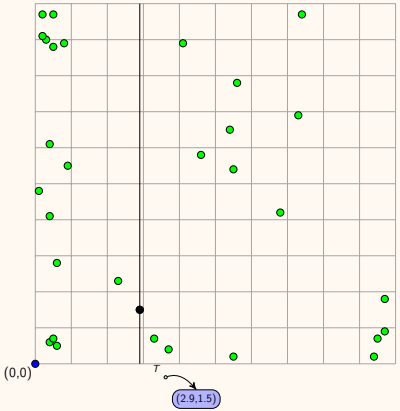
Exemplo 1.3 (Construção)

2. Associar o ponto relativo à mediana a um nó da árvore:



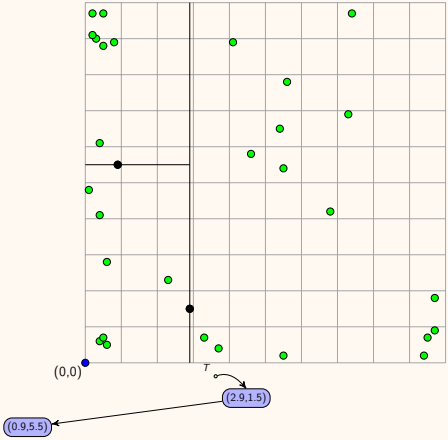
Exemplo 1.3 (Construção)

2. Associar o ponto relativo à mediana a um nó da árvore:



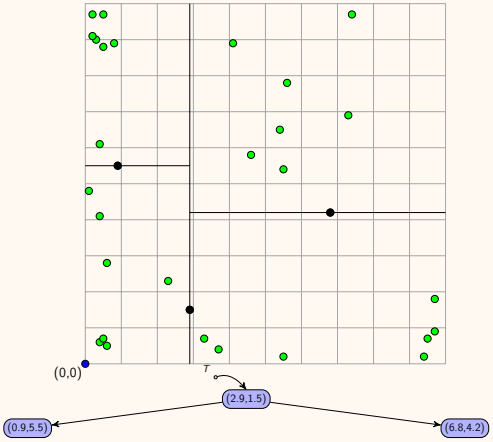
Exemplo 1.3 (Construção)

2. Associar o ponto relativo à mediana a um nó da árvore:



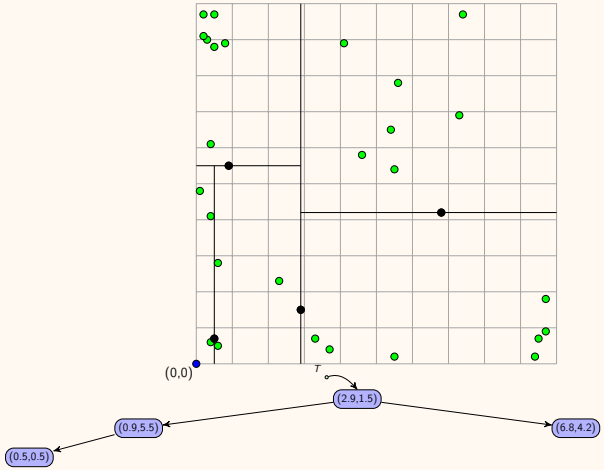
Exemplo 1.3 (Construção)

2. Associar o ponto relativo à mediana a um nó da árvore:



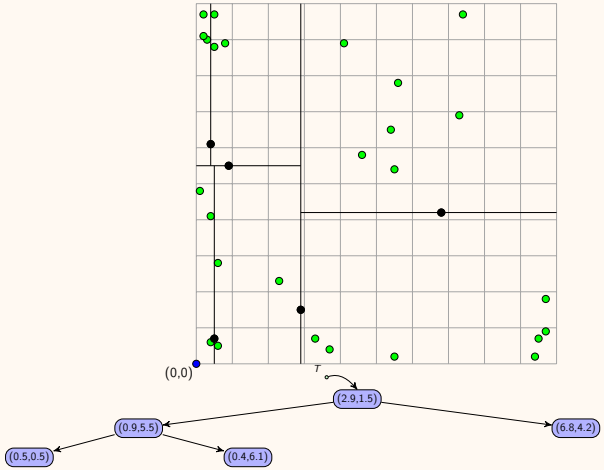
Exemplo 1.3 (Construção)

2. Associar o ponto relativo à mediana a um nó da árvore:



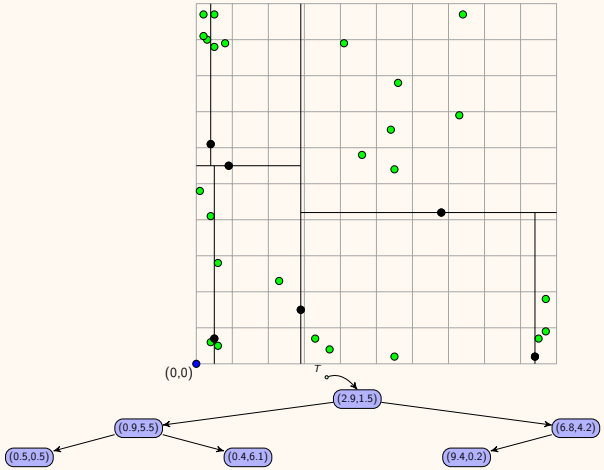
Exemplo 1.3 (Construção)

2. Associar o ponto relativo à mediana a um nó da árvore:



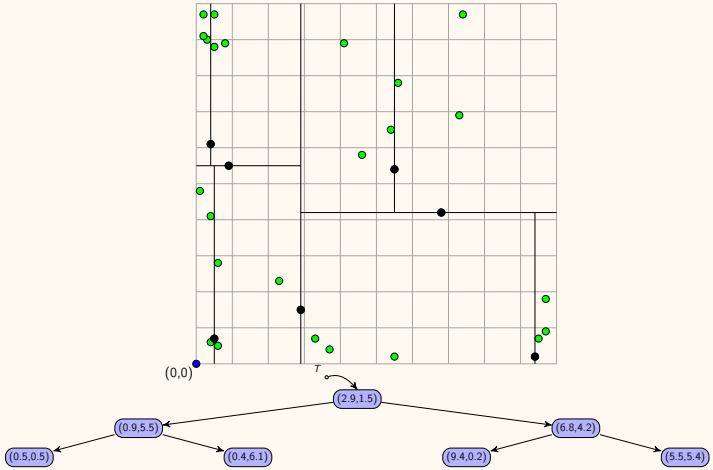
Exemplo 1.3 (Construção)

2. Associar o ponto relativo à mediana a um nó da árvore:



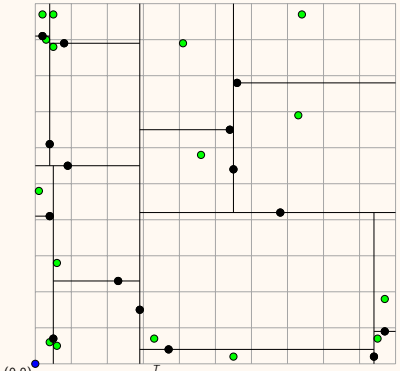
Exemplo 1.3 (Construção)

2. Associar o ponto relativo à mediana a um nó da árvore:

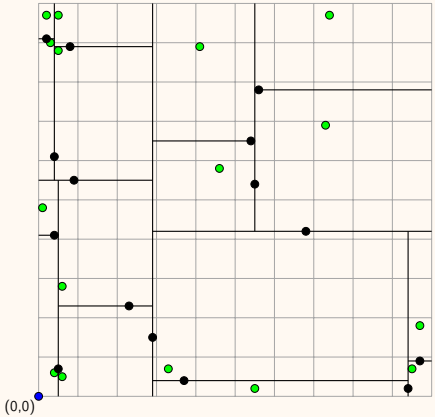


Exemplo 1.3 (Construção)

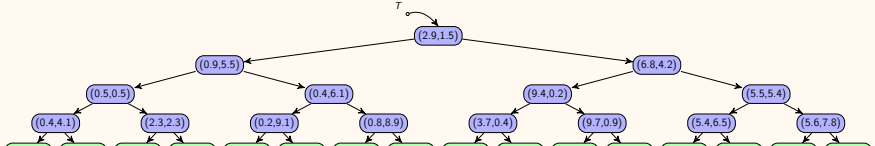
2. Associar o ponto relativo à mediana a um nó da árvore:



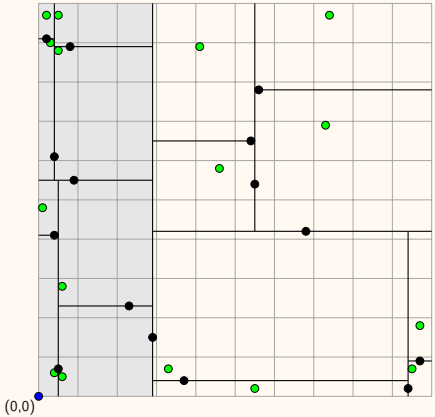
Particionamento do espaço na *k-D Tree*



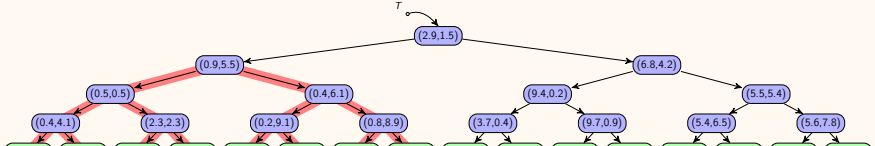
- $[0.0, 0.0] \times [2.9, 10.0]$.
- $[0.0, 5.5] \times [2.9, 10.0]$.
- $[0.0, 0.0] \times [2.9, 5.5]$.
- $[2.9, 0.0] \times [10.0, 10.0]$.
- $[2.9, 0.0] \times [10.0, 4.2]$.
- $[2.9, 4.2] \times [10.0, 10.0]$.



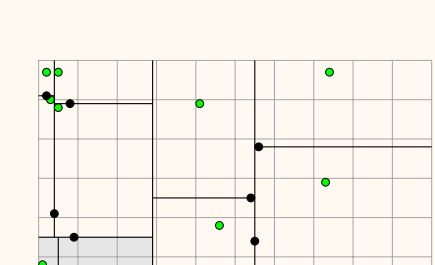
Particionamento do espaço na *k-D Tree*



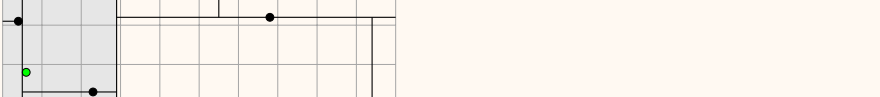
- $[0.0, 0.0] \times [2.9, 10.0]$.
- $[0.0, 5.5] \times [2.9, 10.0]$.
- $[0.0, 0.0] \times [2.9, 5.5]$.
- $[2.9, 0.0] \times [10.0, 10.0]$.
- $[2.9, 0.0] \times [10.0, 4.2]$.
- $[2.9, 4.2] \times [10.0, 10.0]$.



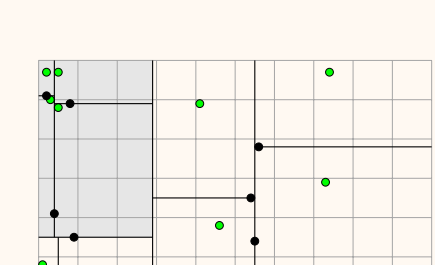
Particionamento do espaço na *k-D Tree*



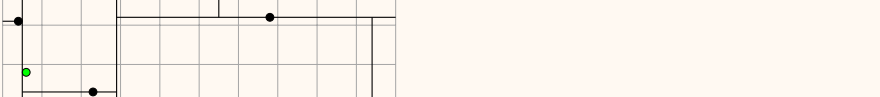
- $[0.0, 0.0] \times [2.9, 10.0]$.
- $[0.0, 5.5] \times [2.9, 10.0]$.
- $[0.0, 0.0] \times [2.9, 5.5]$.
- $[2.9, 0.0] \times [10.0, 10.0]$.
- $[2.9, 0.0] \times [10.0, 4.2]$.
- $[2.9, 4.2] \times [10.0, 10.0]$.



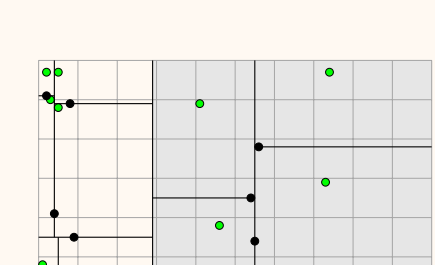
Particionamento do espaço na *k-D Tree*



- $[0.0, 0.0] \times [2.9, 10.0]$.
- $[0.0, 5.5] \times [2.9, 10.0]$.
- $[0.0, 0.0] \times [2.9, 5.5]$.
- $[2.9, 0.0] \times [10.0, 10.0]$.
- $[2.9, 0.0] \times [10.0, 4.2]$.
- $[2.9, 4.2] \times [10.0, 10.0]$.

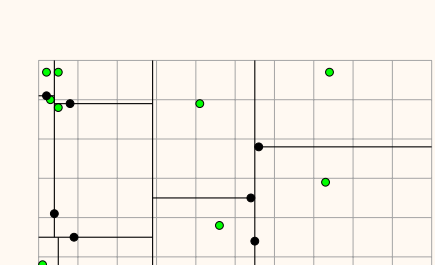


Particionamento do espaço na k-D Tree



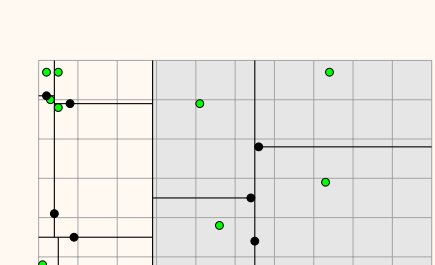
- $[0.0, 0.0] \times [2.9, 10.0]$.
- $[0.0, 5.5] \times [2.9, 10.0]$.
- $[0.0, 0.0] \times [2.9, 5.5]$.
- $[2.9, 0.0] \times [10.0, 10.0]$.
- $[2.9, 0.0] \times [10.0, 4.2]$.
- $[2.9, 4.2] \times [10.0, 10.0]$.

Particionamento do espaço na *k-D Tree*

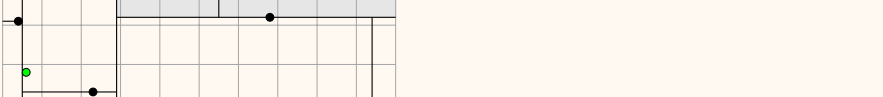


- $[0.0, 0.0] \times [2.9, 10.0]$.
- $[0.0, 5.5] \times [2.9, 10.0]$.
- $[0.0, 0.0] \times [2.9, 5.5]$.
- $[2.9, 0.0] \times [10.0, 10.0]$.
- $[2.9, 0.0] \times [10.0, 4.2]$.
- $[2.9, 4.2] \times [10.0, 10.0]$.

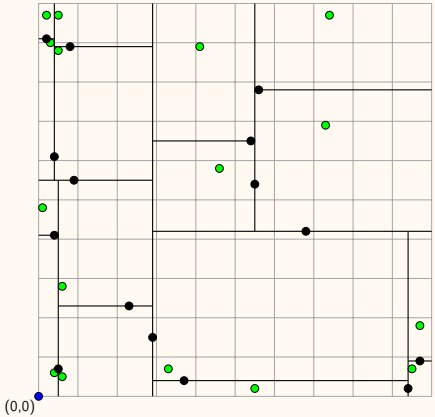
Particionamento do espaço na *k-D Tree*

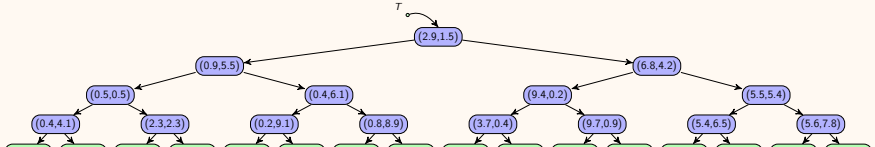


- $[0.0, 0.0] \times [2.9, 10.0]$.
- $[0.0, 5.5] \times [2.9, 10.0]$.
- $[0.0, 0.0] \times [2.9, 5.5]$.
- $[2.9, 0.0] \times [10.0, 10.0]$.
- $[2.9, 0.0] \times [10.0, 4.2]$.
- $[2.9, 4.2] \times [10.0, 10.0]$.

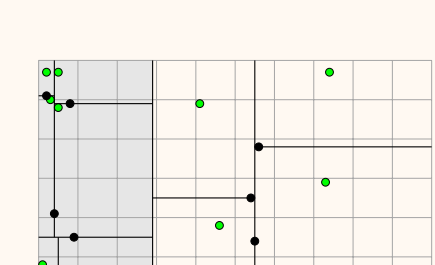


Caminhos na árvore



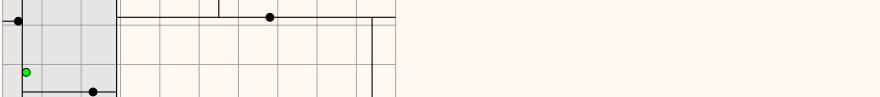


Caminhos na árvore

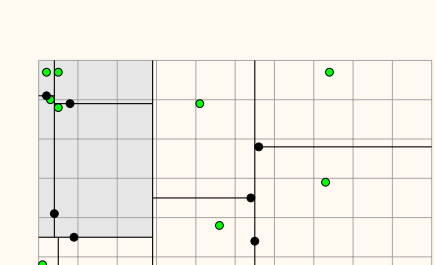


- Caminhos a partir da raiz limitam as regiões dos pontos nas sub-árvores:

$$[0.0, 0.0] \times [2.9, 10.0].$$

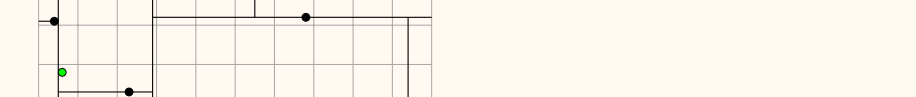


Caminhos na árvore

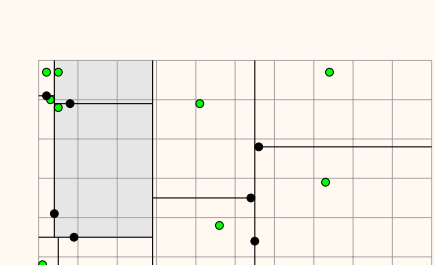


- Caminhos a partir da raiz limitam as regiões dos pontos nas sub-árvores:

$$[0.0, 5.5] \times [2.9, 10.0].$$

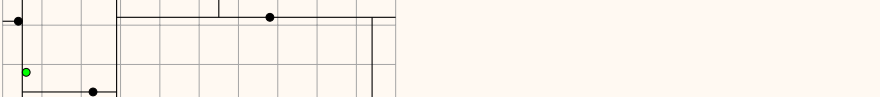


Caminhos na árvore

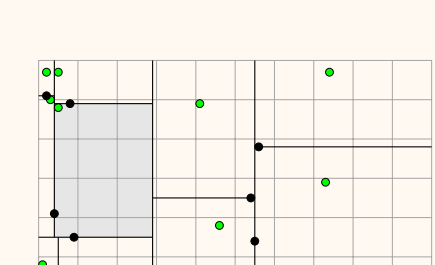


- Caminhos a partir da raiz limitam as regiões dos pontos nas sub-árvores:

$$[0.4, 5.5] \times [2.9, 10.0].$$

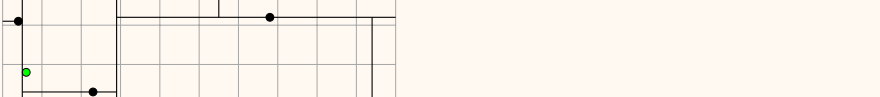


Caminhos na árvore

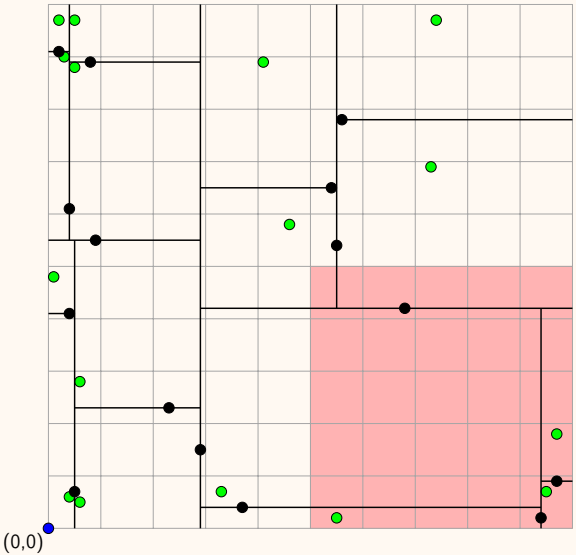


- Caminhos a partir da raiz limitam as regiões dos pontos nas sub-árvores:

$$[0.4, 5.5] \times [2.9, 8.9].$$



- Quantos pontos existem no quadrante $[5.0, 0.0] \times [10.0, 5.0]$?

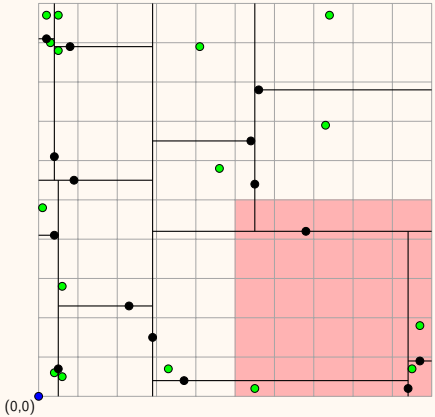


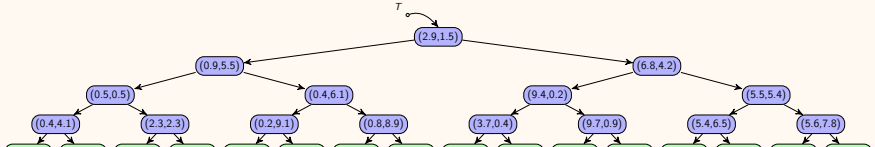
Nr. de pontos em um subespaço

Algoritmo básico para contar os pontos

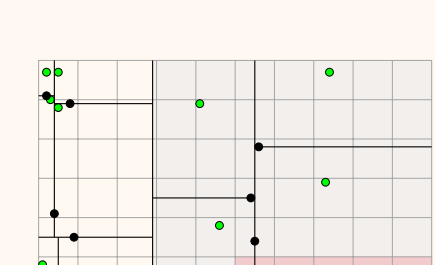
1. Verificar a coordenada correspondente ao nível corrente da k-*d tree*.
2. Se essa coordenada é menor que o intervalo correspondente do subespaço, visitar apenas a subárvore direita.
3. Se essa coordenada é maior que o intervalo correspondente do subespaço, visitar apenas a subárvore esquerda.
4. Caso contrário, verificar se a raiz está no subespaço e visitar ambas subárvores.
5. Caso especial:
   - como pode-se identificar o intervalo no qual uma subárvore pode cair, pode ser possível adicionar uma sub-árvore inteira.

Nr. de pontos em $[5.0, 0.0] \times [10.0, 5.0]$?

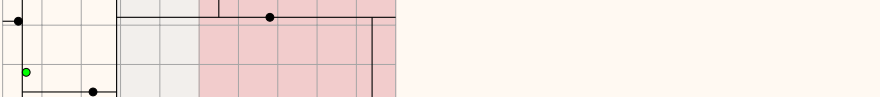




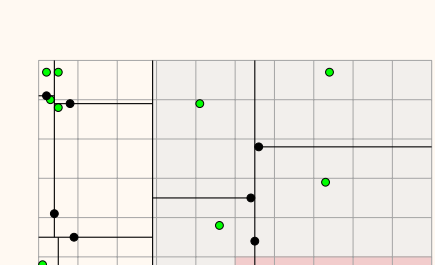
Nr. de pontos em $[5.0, 0.0] \times [10.0, 5.0]$?



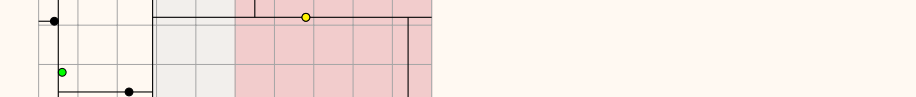
- Nr. de pontos = 0.
- Raiz $(2.9, 1.5)$ da *k-d tree*: como $2.9 < 5.0$, apenas a subárvore direita deve ser visitada.



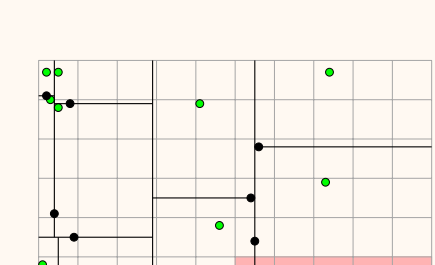
Nr. de pontos em $[5.0, 0.0] \times [10.0, 5.0]$?



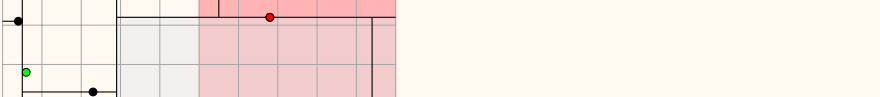
- Nr. de pontos = 1.
- $(6.8, 4.2) \in [5.0, 0.0] \times [10.0, 5.0]$ e ambas subárvores devem ser visitadas.



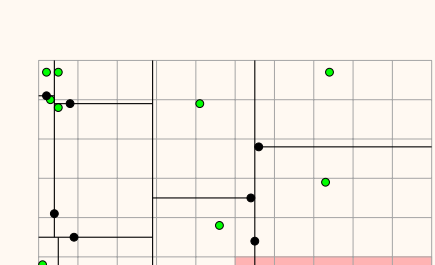
Nr. de pontos em $[5.0, 0.0] \times [10.0, 5.0]$?



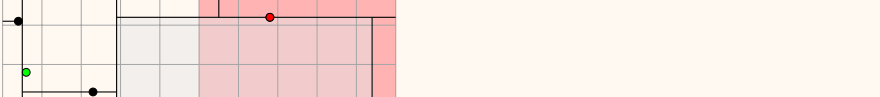
- Nr. de pontos = 2.
- $(9.4, 0.2) \in [5.0, 0.0] \times [10.0, 5.0]$ e ambas subárvores devem ser visitadas.



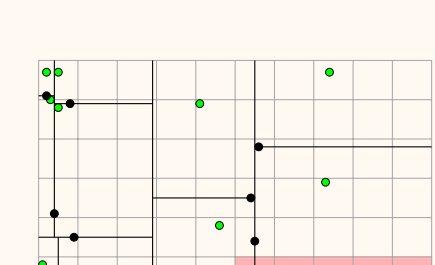
Nr. de pontos em $[5.0, 0.0] \times [10.0, 5.0]$?



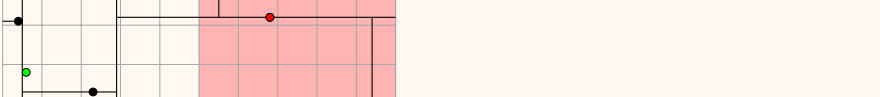
- Nr. de pontos = 2.
- $(3.7, 0.4) \notin [5.0, 0.0] \times [10.0, 5.0]$, mas ambas subárvores devem ser visitadas.



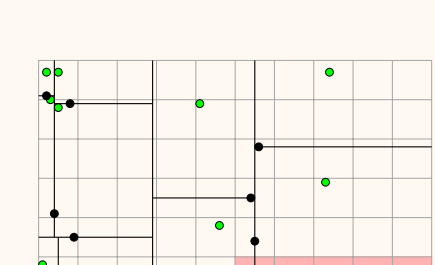
Nr. de pontos em $[5.0, 0.0] \times [10.0, 5.0]$?



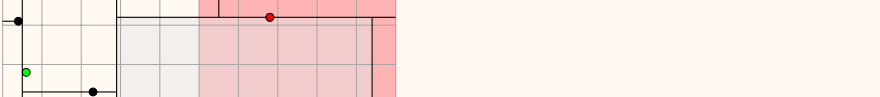
- Nr. de pontos = 3.
- $(5.5, 0.2) \in [5.0, 0.0] \times [10.0, 5.0]$ (nó folha).



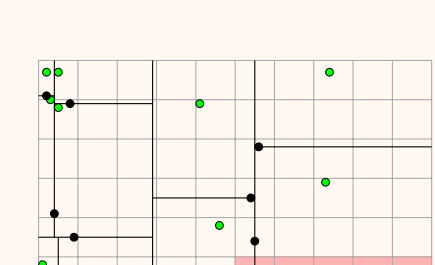
Nr. de pontos em $[5.0, 0.0] \times [10.0, 5.0]$?



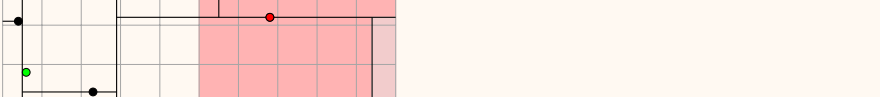
- Nr. de pontos = 3.
- $(3.3, 0.7) \notin [5.0, 0.0] \times [10.0, 5.0]$ (nó folha).



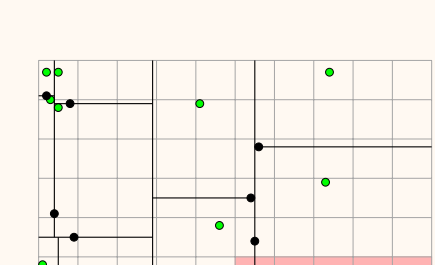
Nr. de pontos em $[5.0, 0.0] \times [10.0, 5.0]$?



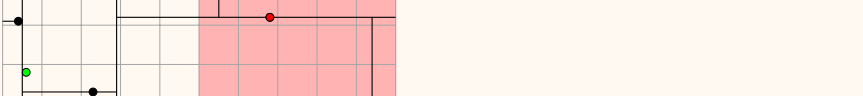
- Nr. de pontos = 4.
- $(9.7, 0.9) \in [5.0, 0.0] \times [10.0, 5.0]$ e ambas subárvores devem ser visitadas.



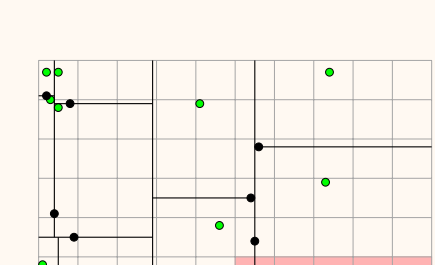
Nr. de pontos em $[5.0, 0.0] \times [10.0, 5.0]$?



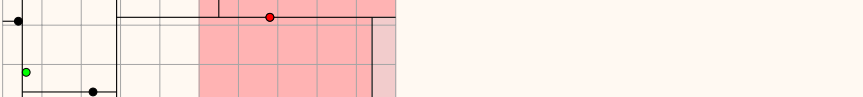
- Nr. de pontos = 5.
- $(9.5, 0.7) \in [5.0, 0.0] \times [10.0, 5.0]$ (nó folha).



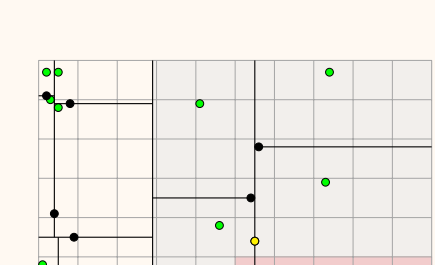
Nr. de pontos em $[5.0, 0.0] \times [10.0, 5.0]$?



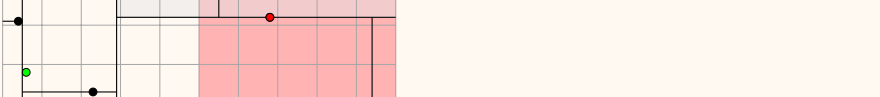
- Nr. de pontos = 6.
- $(9.7, 1.8) \in [5.0, 0.0] \times [10.0, 5.0]$ (nó folha).



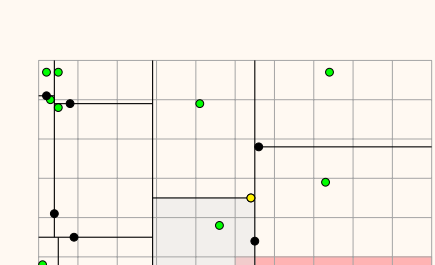
Nr. de pontos em $[5.0, 0.0] \times [10.0, 5.0]$?



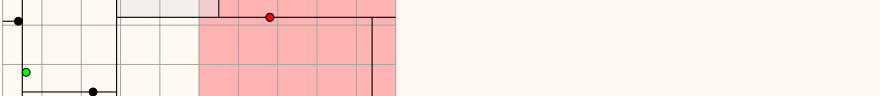
- Nr. de pontos = 6.
- $(5.5, 5.4) \notin [5.0, 0.0] \times [10.0, 5.0]$, mas ambas subárvores devem ser visitadas.



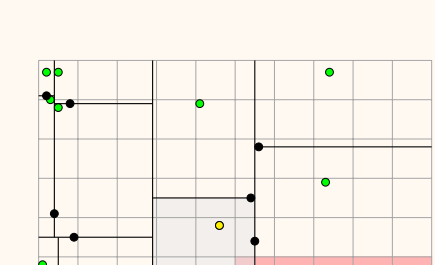
Nr. de pontos em $[5.0, 0.0] \times [10.0, 5.0]$?



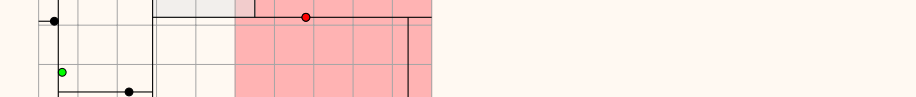
- Nr. de pontos = 6.
- $(5.4, 6.5) \notin [5.0, 0.0] \times [10.0, 5.0]$ e $6.5 > 5.0$ e apenas subárvore esquerda deve ser visitada.



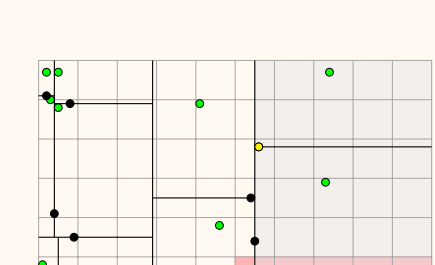
Nr. de pontos em $[5.0, 0.0] \times [10.0, 5.0]$?



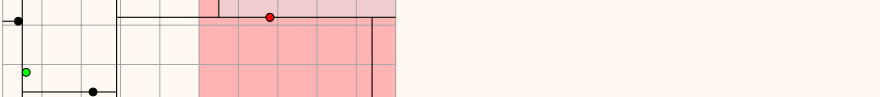
- Nr. de pontos = 6.
- $(4.6, 5.8) \notin [5.0, 0.0] \times [10.0, 5.0]$ (nó folha).



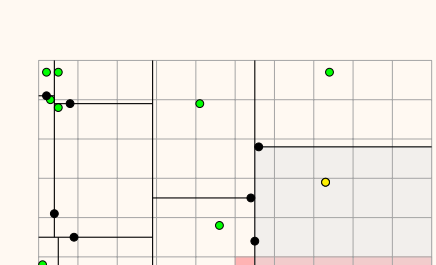
Nr. de pontos em $[5.0, 0.0] \times [10.0, 5.0]$?



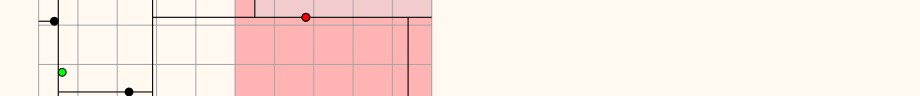
- Nr. de pontos = 6.
- $(5.6, 7.8) \notin [5.0, 0.0] \times [10.0, 5.0]$ e $7.8 > 5.0$ e apenas subárvore esquerda deve ser visitada.



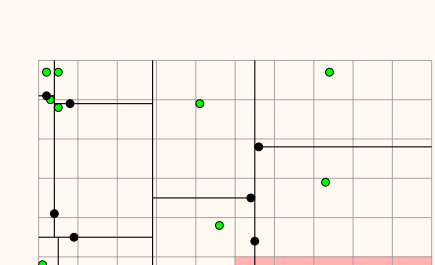
Nr. de pontos em $[5.0, 0.0] \times [10.0, 5.0]$?



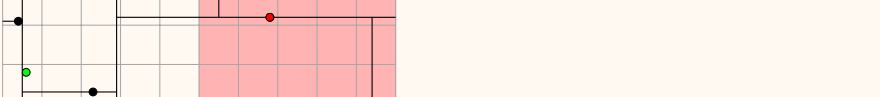
- Nr. de pontos = 6.
- $(7.3, 6.9) \notin [5.0, 0.0] \times [10.0, 5.0]$ (nó folha).



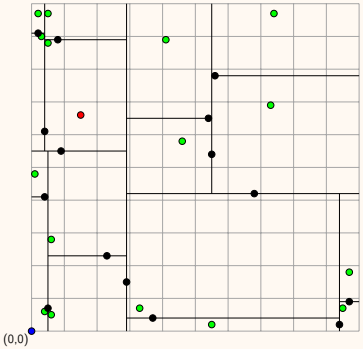
Nr. de pontos em $[5.0, 0.0] \times [10.0, 5.0]$?



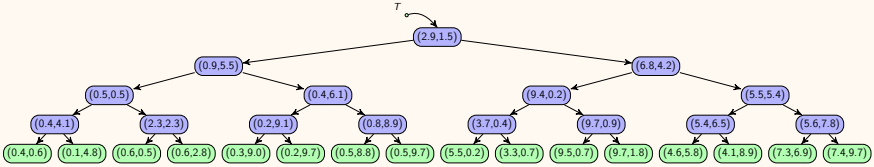
- Nr. de pontos = 6.
- Menos da metade dos nós da k-*d tree* foram visitados.



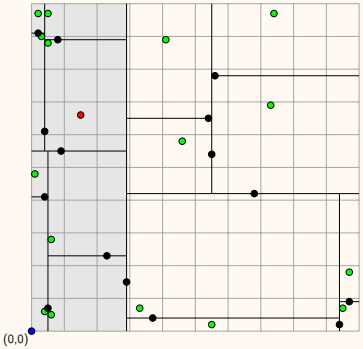
Inserção na k-d *tree*



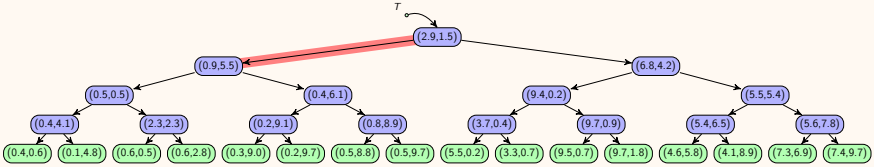
- Inserção do ponto $(1.5, 6.6)$.



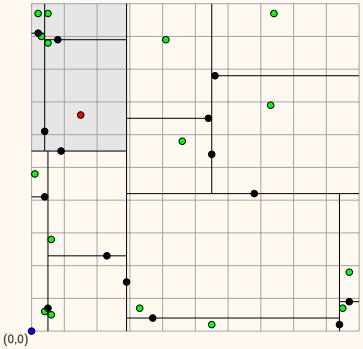
Inserção na k-d *tree*



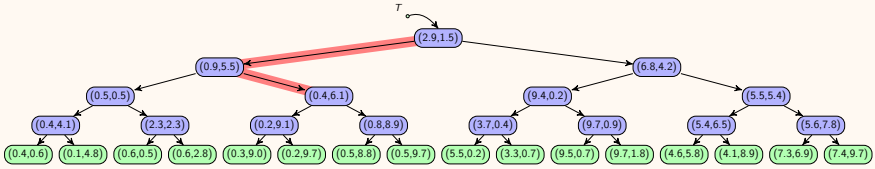
- Inserção do ponto $(1.5, 6.6)$: $1.5 < 2.9$.



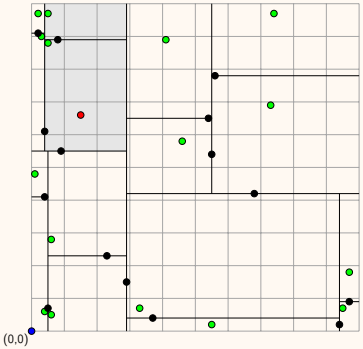
Inserção na k-d *tree*



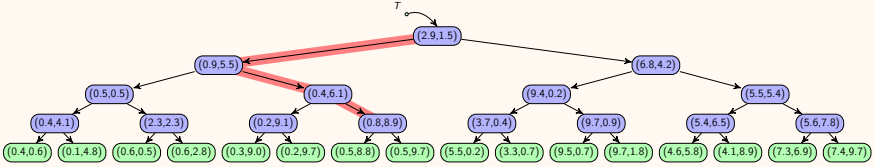
- Inserção do ponto $(1.5, 6.6)$:
  $6.6 > 5.5$.



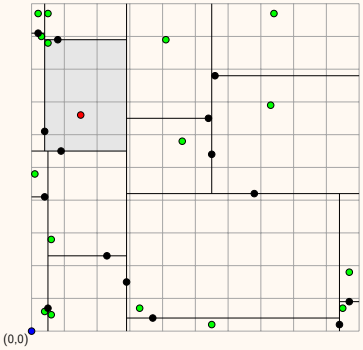
Inserção na k-d *tree*



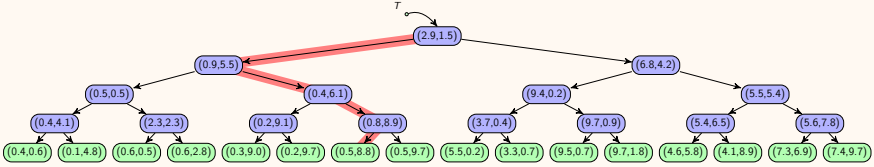
- Inserção do ponto $(1.5, 6.6)$: $1.5 > 0.4$.



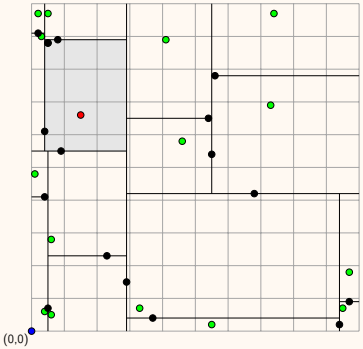
Inserção na k-d *tree*



- Inserção do ponto $(1.5, 6.6)$: $6.6 < 8.9$.

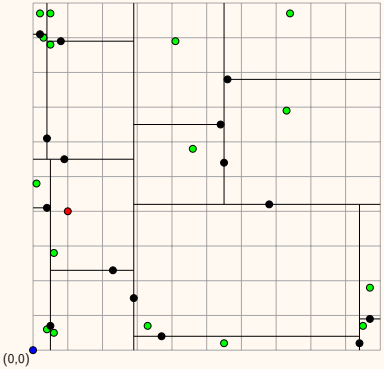


Inserção na k-d *tree*

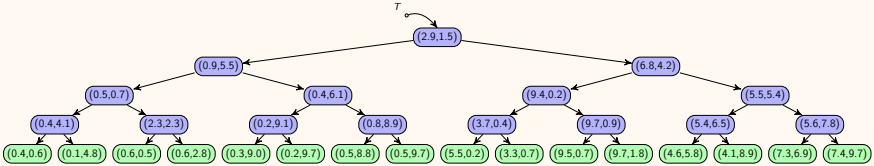


- Inserção do ponto $(1.5, 6.6)$: $1.5 > 0.5$.

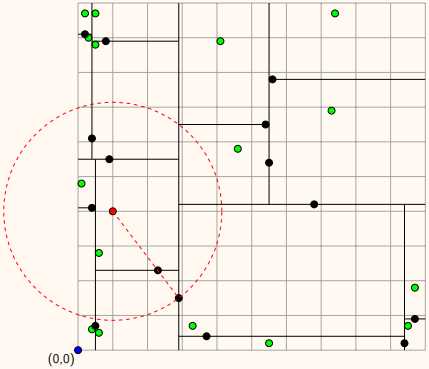
Vizinho mais próximo



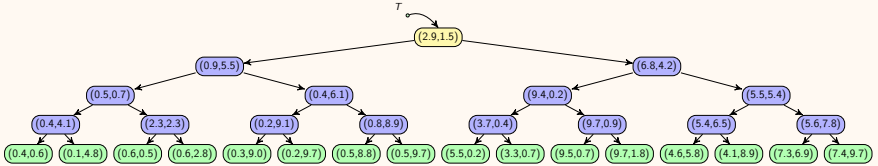
- Vizinho mais próximo do ponto $(1.0, 4.0)$?
- *NN – nearest neighbour.*



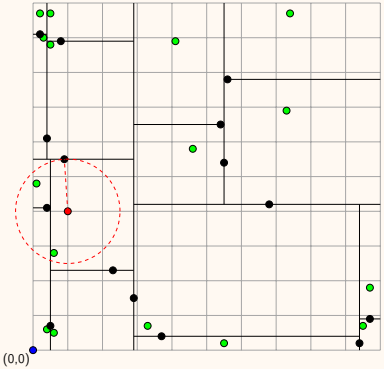
Vizinho mais próximo



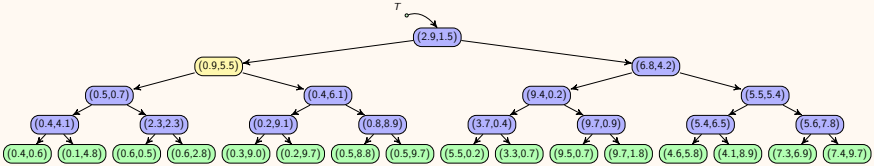
- Começar a busca em $(2.9, 1.5)$. Definir melhor estimativa como distância até $(1.0, 4.0)$. Em seguida, fazer busca em profundidade.
- Círculo NN intercepta todas as regiões (não se pode descartar qualquer região).



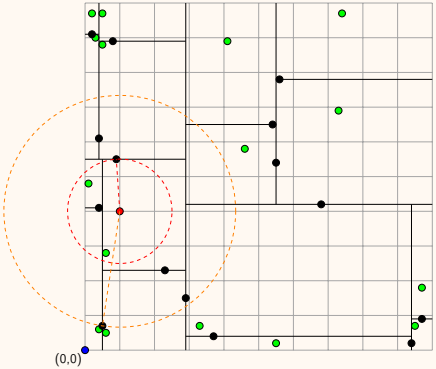
Vizinho mais próximo



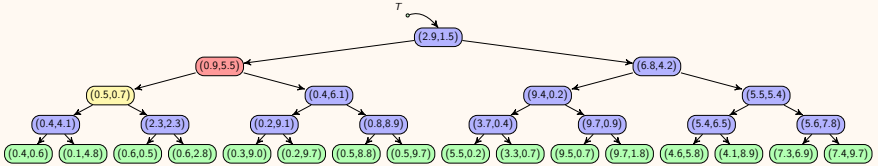
- Calcular a distância de $(0.9, 5.5)$ até $(1.0, 4.0)$ e comparar com melhor estimativa. Como a distância é menor, atualizar melhor estimativa.
- Examinar filhos à esquerda e à direita.



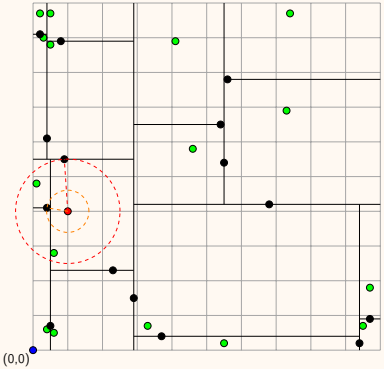
Vizinho mais próximo



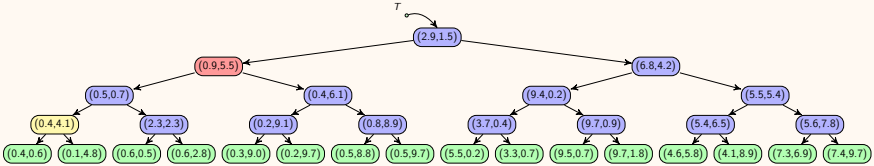
- Calcular a distância de $(0.5, 0.7)$ até $(1.0, 4.0)$ e comparar com melhor estimativa. Como a distância é maior, não atualizar melhor estimativa.
- Região não pode ser descartada, pois intercepta com círculo NN.



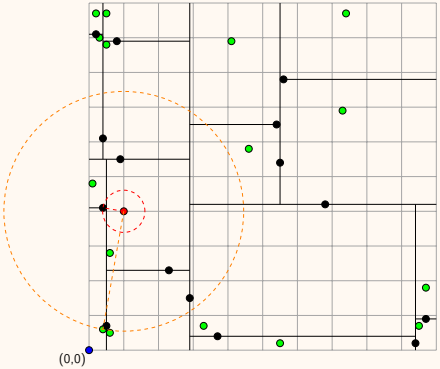
Vizinho mais próximo



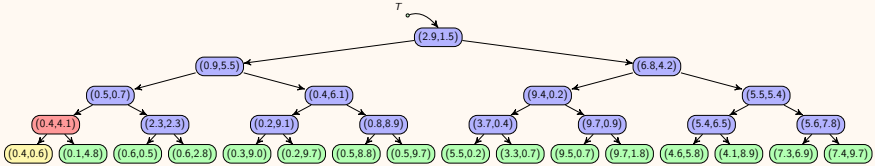
- Calcular a distância de $(0.4, 4.1)$ até $(1.0, 4.0)$ e comparar com melhor estimativa. Como a distância é menor, atualizar melhor estimativa.



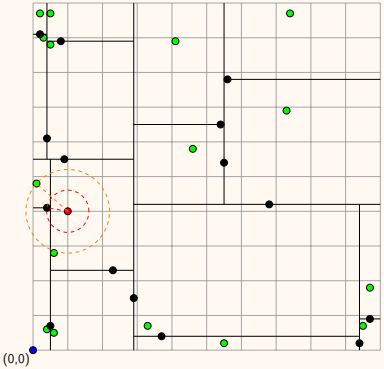
Vizinho mais próximo



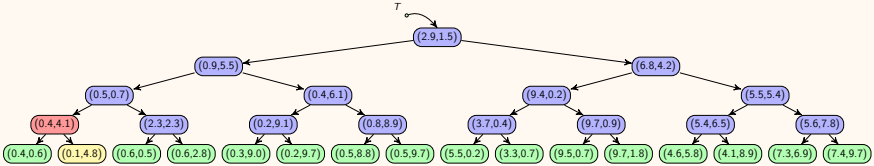
- Calcular a distância de $(0.4, 0.6)$ até $(1.0, 4.0)$ e comparar com melhor estimativa. Como a distância é maior, não atualizar melhor estimativa.



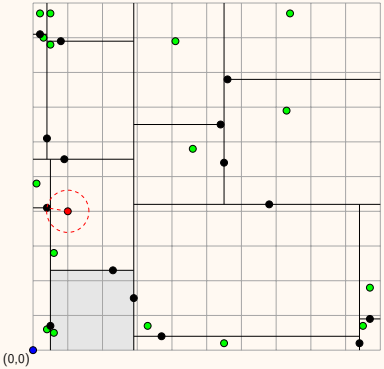
Vizinho mais próximo



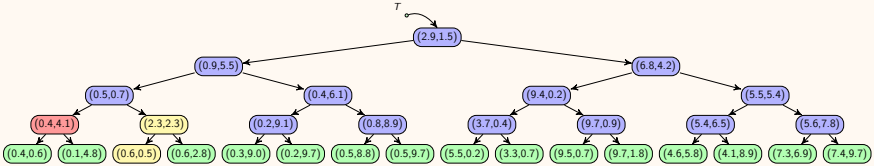
- Calcular a distância de $(0.1, 4.8)$ até $(1.0, 4.0)$ e comparar com melhor estimativa. Como a distância é maior, não atualizar melhor estimativa.



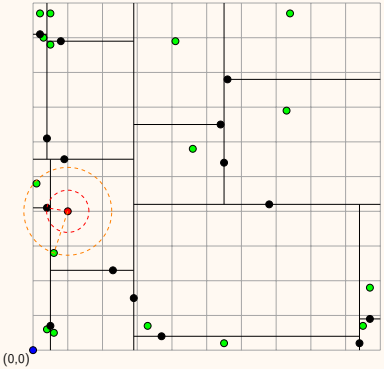
Vizinho mais próximo



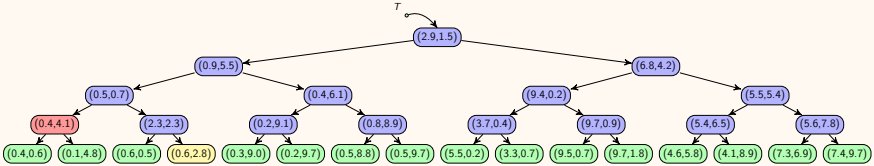
- Região não intercepta o círculo com a melhor estimativa e não pode conter NN.



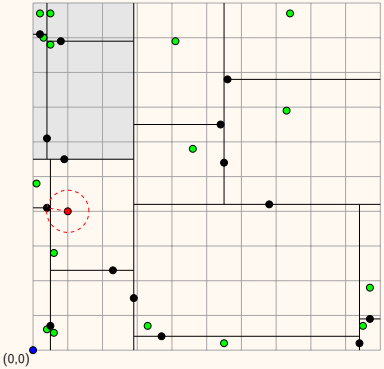
Vizinho mais próximo



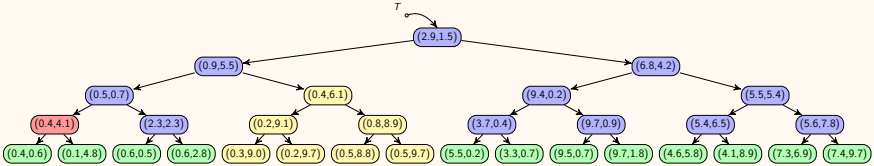
- Calcular a distância de $(0.6, 2.8)$ até $(1.0, 4.0)$ e comparar com melhor estimativa. Como a distância é maior, não atualizar melhor estimativa.



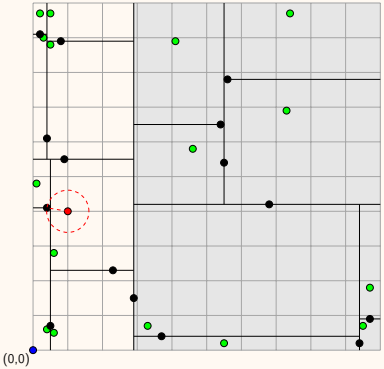
Vizinho mais próximo



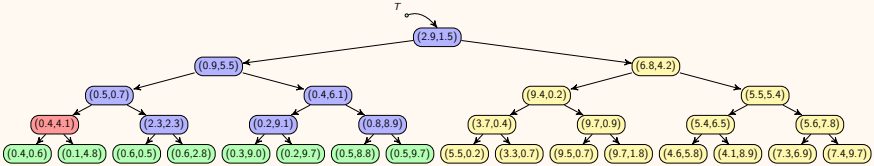
- Região não intercepta o círculo com a melhor estimativa e não pode conter NN.



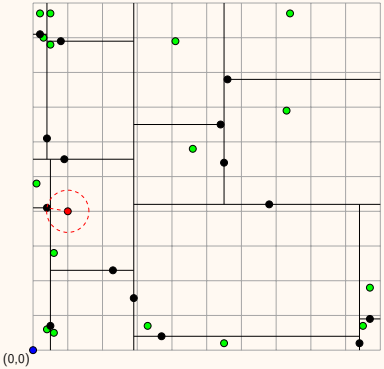
Vizinho mais próximo



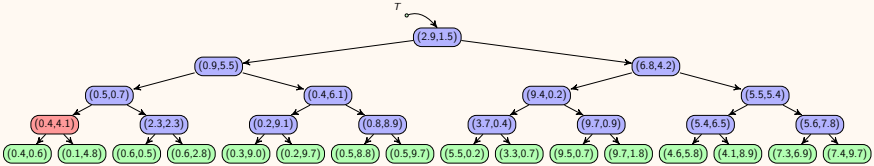
- Região não intercepta o círculo com a melhor estimativa e não pode conter NN.



Vizinho mais próximo



- Vizinho mais próximo do ponto $(1.0, 4.0)$ é o ponto $(0.4, 4.1)$.



Livros Texto I


J. L. Gersting
Fundamentos Matemáticos para a Ciência da Computação.
LTC Editora, 3ª Edição, 2001.


R. P. Grimaldi
Discrete and Combinatorial Mathematics – An Applied Introduction.
Addison Wesley, 1994.


D. J. Velleman
How To Prove It – A Structured Approach.
Cambridge University Press, 1996.


T. H. Cormen, C. E. Leiserson e R. L. Rivest.
Introduction to Algorithms.
McGraw-Hill, New York, 1990.

Livros Texto II


C. H. Papadimitriou, U. V. Vazirani e S. Dasgupta.
Algoritmos.
Mcgraw-Hill Brasil, 2009.


U. Manber.
Algorithms: A Creative Approach.
Addison-Wesley, 1989.


D. E. Knuth.
The Art of Computer Programming. Volume 1 – Fundamental Algorithms.
Addison Wesley, 1998.


D. E. Knuth.
The Art of Computer Programming. Volume 2 – Sorting and Searching.
Addison Wesley, 1998.


N. Ziviani.
Projeto de Algoritmos com Implementações em Pascal e C.
Editora Thomson. 2a Edição. 2004.