Testes

Débora Zupeli Bossois
João Olavo Baião de Vasconcelos

3 de março de 2005

1 Testes comparativos utilizando a KDTree

Rodamos o algoritmo KDTree para alguns conjuntos de pontos a fim de obter uma tabela com os tempos de execução de cada conjunto e a quantidade de memória usada por cada conjunto.

KDTree		
N.ptos	**Tempo**	**Memória** (10^5)
att48	0m0.018s	8,363 bytes
kroA100	0m0.027s	17,000 bytes
lin318	0m0.060s	54,347 bytes
pr1002	0m0.154s	177,806 bytes
nrw1379	0m0.212s	241,071 bytes
pla7397	0m1.255s	1.415,285 bytes
pla33810	0m6.165s	6.682,207 bytes
pla85900	0m16.313s	17,505.565 bytes

Tabela 1: Tempo e uso de memória do módulo KDTree

AQUI FICAM OS COMENTÁRIOS!!

2 Testes de correção do Módulo Binpq

Serão aqui mostrados testes de correção dos operadores utilizados no módulo *Binpq*.

Operadores:

- findmin
- delmin
- merge
- insertBinpq

As variações de parâmetros que aqui fazemos são pensadas analizando cada algoritmo.

2.1 Operador *findmin*

Para o teste do operador *findmin*, utilizamos um conjunto de 5 árvores de prioridade. Foram elas:

- (NodePQ 2 (NodePQ 3 LeafPQ LeafPQ) (NodePQ 4 LeafPQ LeafPQ))
- (NodePQ 2 (NodePQ 3 LeafPQ LeafPQ) LeafPQ)
- (NodePQ 2 LeafPQ (NodePQ 4 LeafPQ LeafPQ))
- (NodePQ 2 LeafPQ LeafPQ)
- LeafPQ

Eis aqui os resultados para cada dado:

```
Binpq> findmin (NodePQ 2 (NodePQ 3 LeafPQ LeafPQ) (NodePQ 4 LeafPQ LeafPQ))
Just 2
Binpq> findmin (NodePQ 2 (NodePQ 3 LeafPQ LeafPQ) LeafPQ)
Just 2
Binpq> findmin (NodePQ 2 LeafPQ (NodePQ 4 LeafPQ LeafPQ))
Just 2
Binpq> findmin (NodePQ 2 LeafPQ LeafPQ)
Just 2
Binpq> findmin LeafPQ
Nothing
```

Essas árvores cobrem todas as possibilidades de parâmetros que a função pode receber, e todas as respostas obtidas foram as esperadas.

2.2 Operador *delmin*

As mesmas opções de parâmetros utilizadas no operador acima também cobrem todas as possibilidades de parâmetros do operador *delmin*. Vamos aos testes:

```
*Binpq> delmin (NodePQ 2 (NodePQ 3 LeafPQ LeafPQ) (NodePQ 4 LeafPQ LeafPQ))
NodePQ 3 (NodePQ 4 LeafPQ LeafPQ) LeafPQ
*Binpq> delmin (NodePQ 2 (NodePQ 3 LeafPQ LeafPQ) LeafPQ)
NodePQ 3 LeafPQ LeafPQ
*Binpq> delmin (NodePQ 2 LeafPQ (NodePQ 4 LeafPQ LeafPQ))
NodePQ 4 LeafPQ LeafPQ
*Binpq> delmin (NodePQ 2 LeafPQ LeafPQ)
LeafPQ
*Binpq> delmin LeafPQ
LeafPQ
```

Obtemos os resultados corretos.

2.3 Operador *merge*

Aplicamos as diversas variações possíveis para *merge* e, como visto no resultado abaixo, as operações foram concluídas com êxito.

```
*Binpq> merge LeafPQ LeafPQ
LeafPQ
*Binpq> merge LeafPQ (NodePQ 2 LeafPQ LeafPQ)
NodePQ 2 LeafPQ LeafPQ
*Binpq> merge (NodePQ 2 LeafPQ LeafPQ) LeafPQ
NodePQ 2 LeafPQ LeafPQ
*Binpq> merge (NodePQ 2 LeafPQ LeafPQ) (NodePQ 10 LeafPQ LeafPQ)
NodePQ 2 (NodePQ 10 LeafPQ LeafPQ) LeafPQ
*Binpq> merge (NodePQ 10 LeafPQ LeafPQ) (NodePQ 2 LeafPQ LeafPQ)
NodePQ 2 (NodePQ 10 LeafPQ LeafPQ) LeafPQ
*Binpq> merge (NodePQ 2 (NodePQ 3 LeafPQ LeafPQ) LeafPQ) (NodePQ 10 LeafPQ LeafPQ)
NodePQ 2 (NodePQ 10 LeafPQ LeafPQ) (NodePQ 3 LeafPQ LeafPQ)
*Binpq> merge (NodePQ 2 LeafPQ (NodePQ 3 LeafPQ LeafPQ)) (NodePQ 10 LeafPQ LeafPQ)
NodePQ 2 (NodePQ 3 (NodePQ 10 LeafPQ LeafPQ) LeafPQ) LeafPQ
*Binpq> merge (NodePQ 2 (NodePQ 3 LeafPQ LeafPQ) (NodePQ 4 LeafPQ LeafPQ))
        (NodePQ 10 LeafPQ LeafPQ)
NodePQ 2 (NodePQ 4 (NodePQ 10 LeafPQ LeafPQ) LeafPQ) (NodePQ 3 LeafPQ LeafPQ)
*Binpq> merge (NodePQ 10 (NodePQ 13 LeafPQ LeafPQ) LeafPQ) (NodePQ 2 LeafPQ LeafPQ)
NodePQ 2 (NodePQ 10 (NodePQ 13 LeafPQ LeafPQ) LeafPQ) LeafPQ
*Binpq> merge (NodePQ 10 LeafPQ (NodePQ 13 LeafPQ LeafPQ)) (NodePQ 2 LeafPQ LeafPQ)
NodePQ 2 (NodePQ 10 LeafPQ (NodePQ 13 LeafPQ LeafPQ)) LeafPQ
*Binpq> merge (NodePQ 10 (NodePQ 13 LeafPQ LeafPQ) (NodePQ 14 LeafPQ LeafPQ))
        (NodePQ 2 LeafPQ LeafPQ)
NodePQ 2 (NodePQ 10 (NodePQ 13 LeafPQ LeafPQ) (NodePQ 14 LeafPQ LeafPQ)) LeafPQ
*Binpq> merge (NodePQ 2 LeafPQ LeafPQ) (NodePQ 10 (NodePQ 3 LeafPQ LeafPQ) LeafPQ)
NodePQ 2 (NodePQ 10 (NodePQ 3 LeafPQ LeafPQ) LeafPQ) LeafPQ
*Binpq> merge (NodePQ 2 LeafPQ LeafPQ) (NodePQ 10 LeafPQ (NodePQ 3 LeafPQ LeafPQ))
NodePQ 2 (NodePQ 10 LeafPQ (NodePQ 3 LeafPQ LeafPQ)) LeafPQ
*Binpq> merge (NodePQ 2 LeafPQ LeafPQ) (NodePQ 10 (NodePQ 3 LeafPQ LeafPQ)
        (NodePQ 4 LeafPQ LeafPQ))
NodePQ 2 (NodePQ 10 (NodePQ 3 LeafPQ LeafPQ) (NodePQ 4 LeafPQ LeafPQ)) LeafPQ
*Binpq> merge (NodePQ 10 LeafPQ LeafPQ) (NodePQ 2 (NodePQ 3 LeafPQ LeafPQ) LeafPQ)
NodePQ 2 (NodePQ 10 LeafPQ LeafPQ) (NodePQ 3 LeafPQ LeafPQ)
*Binpq> merge (NodePQ 10 LeafPQ LeafPQ) (NodePQ 2 LeafPQ (NodePQ 3 LeafPQ LeafPQ))
NodePQ 2 (NodePQ 3 (NodePQ 10 LeafPQ LeafPQ) LeafPQ) LeafPQ
*Binpq> merge (NodePQ 10 LeafPQ LeafPQ) (NodePQ 2 (NodePQ 3 LeafPQ LeafPQ)
        (NodePQ 4 LeafPQ LeafPQ))
NodePQ 2 (NodePQ 4 (NodePQ 10 LeafPQ LeafPQ) LeafPQ) (NodePQ 3 LeafPQ LeafPQ)
*Binpq> merge (NodePQ 2 (NodePQ 3 LeafPQ LeafPQ) (NodePQ 4 LeafPQ LeafPQ))
         (NodePQ 10 (NodePQ 11 LeafPQ LeafPQ) (NodePQ 12 LeafPQ LeafPQ))
NodePQ 2 (NodePQ 4 (NodePQ 10 (NodePQ 11 LeafPQ LeafPQ) (NodePQ 12 LeafPQ LeafPQ))
         LeafPQ) (NodePQ 3 LeafPQ LeafPQ)
*Binpq> merge (NodePQ 10 (NodePQ 13 LeafPQ LeafPQ) (NodePQ 14 LeafPQ LeafPQ))
         (NodePQ 2 (NodePQ 11 LeafPQ LeafPQ) (NodePQ 12 LeafPQ LeafPQ))
NodePQ 2 (NodePQ 10 (NodePQ 12 (NodePQ 14 LeafPQ LeafPQ) LeafPQ)
         (NodePQ 13 LeafPQ LeafPQ)) (NodePQ 11 LeafPQ LeafPQ)
```

2.4 Operador *insertBinpq*

Este operador insere um elemento em uma árvore de prioridade. Seguindo o algoritmo, um exemplo as possibilidades de parâmetros são:

LeafPQ 3 (NodePQ 3 LeafPQ LeafPQ) 10 (NodePQ 3 (NodePQ 4 LeafPQ LeafPQ) LeafPQ) 10 (NodePQ 3 LeafPQ (NodePQ 4 LeafPQ LeafPQ) 10 (NodePQ 3 (NodePQ 4 LeafPQ LeafPQ) (NodePQ 5 LeafPQ LeafPQ)) 10

Veja os resultados obtidos a partir dessas entradas:

```
*Binpq> insertBinpq LeafPQ  3
NodePQ 3 LeafPQ LeafPQ
*Binpq> insertBinpq (NodePQ 3 LeafPQ LeafPQ) 10
NodePQ 3 (NodePQ 10 LeafPQ LeafPQ) LeafPQ
*Binpq> insertBinpq (NodePQ 3 (NodePQ 4 LeafPQ LeafPQ) LeafPQ) 10
NodePQ 3 (NodePQ 10 LeafPQ LeafPQ) (NodePQ 4 LeafPQ LeafPQ)
*Binpq> insertBinpq (NodePQ 3 LeafPQ (NodePQ 4 LeafPQ LeafPQ)) 10
NodePQ 3 (NodePQ 4 (NodePQ 10 LeafPQ LeafPQ) LeafPQ) LeafPQ
*Binpq> insertBinpq (NodePQ 3 (NodePQ 4 LeafPQ LeafPQ) (NodePQ 5 LeafPQ LeafPQ)) 10
NodePQ 3 (NodePQ 5 (NodePQ 10 LeafPQ LeafPQ) LeafPQ) (NodePQ 4 LeafPQ LeafPQ)
```

Como já foi testado anteriormente a correção do *merge*, a utilização desses parâmetros cerca todas as possibilidades de variação das entradas. E como os resultados obtidos foram os esperados, podemos presumir que o algoritmo está correto.