Implementação Interativa das Árvores B e B+

Guia de Utilização

Francilândio Lima Serafim (Mat.: 472644)
Vanessa Carvalho do Nascimento (Mat.: 471584)

UNIVERSIDADE
FEDERAL DO CEARÁ

07 de Julho de 2023

Conteúdo

1 Introdução

As árvores B e B+ disponibilizadas foram implementadas em python utilizando o jupyter notebook por meio do ambiente Anaconda. Este guia apresenta informações comuns de ambas as árvores. Caso haja alguma especificação que seja diferente entre os casos, esta será citada.

1.1 Bibliotecas necessárias

Lista de bibliotecas necessárias para executar o código:

- graphviz
- collections
- imageio
- math
- io
- numpy
- base64
- PIL
- random
- time
- gradio

  Para instalar uma biblioteca utilizando jupyter notebook do Anaconda com o Google Colab basta usar o comando *pip install bib*, sendo bib substituída pelo nome da biblioteca que se deseja instalar.

1.2 Primeiros Passos

É sugerido por parte dos autores que o usuário execute o código fonte do tipo .ipynb no ambiente Anaconda ou no Google Colab. Em ambos os casos, após a instalação das bibliotecas, deve-se rodar todas as células.

1.3 Entendendo a Estrutura da Árvore

As árvores são representadas por dicionários. As chaves nos dicionários representam os nós e os valores associados serão listas que representam o que o nó armazena. Ao criar uma nova árvore, o dicionário que antes era vazio, passa a ter a seguinte forma:

$$\{0 : [[True, -1], []]\}$$

A lista associada ao nó 0, assim qualquer outro nó que possa pertencer à árvore, possui como primeiro elemento também uma lista. Esta lista pode possuir 2 ou 3 valores. Em qualquer caso, o

primeiro deles é um a valor booleano que diz se o nó é folha (True) ou não (False) e segundo diz quem é o nó pai. O terceiro existirá apenas se o nó for folha (Caso B+), pois dirá quem é o próximo nó folha vizinho à direita. No caso da árvore B, a primeira lista dentro da lista associada ao nó sempre terá apenas 2 valores.

Os outros elementos da lista principal são alternadamente ponteiros e valores, sendo os ponteiros listas com um único elemento ou nenhum (caso sejam folhas). O segundo elemento da lista principal sempre será um ponteiro, o terceiro será um valor armazenado, o quarto, um ponteiro, e assim por diante, sempre finalizando com um ponteiro.

Um nó mais complexo de uma árvore pode ser exemplificado por

$$2 : [[False, 1], [3], 2, [11], 5, [8]]$$

Trata-se do nó 2 de uma árvore. O valor False indica que ele é um nó não folha. O 1 em seguida indica que o nó 2 é filho do nó 1. Observando os ponteiros, pode-se identificar que tal nó possui os nós 3, 11 e 8 como filhos, seguindo as características das árvores, por exemplo:

- caso seja B+, os elementos do nó 3 são menores que 2, e os do nó 11 são maiores ou iguais.

1.4 Operações Manuais usando escrita de comandos

1.5 Criação

Para criar uma nova árvore, basta utilizar o comando *cria_arv*(), que retornará uma árvore com apenas o nó raíz, que estará vazio. Em seguida, para mostrar a árvore graficamente, usa-se o comando *show*(*tree*), sendo tree a árvore que se deseja visualizar.

```
tree=cria_arv()
show(tree)
```

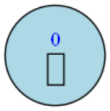

Figura 1: Criação de uma nova árvore.

1.5.1 Inserção

Para inserir um novo valor na árvore, utiliza-se o comando *insercao*(*tree*, *grau*, *novo_valor*) que retornará a nova árvore com o valor *novo_valor* inserido. A função de inserção recebe 3 parâmetros: a árvore com que se deseja fazer a operação, o grau dela e novo valor. No exemplo abaixo, foi utilizado grau 3 e foi solicitado a inserção do valor 9.

```
tree = insercao(tree,3,9)
show(tree)
```

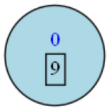

Figura 2: Inserção manual numa árvore.

A figura abaixo exemplifica sucessivas inserções numa árvore.

```
tree=cria_arv()
tree = insercao(tree,3,9)
tree = insercao(tree,3,4)
tree = insercao(tree,3,50)
tree = insercao(tree,3,56)
tree = insercao(tree,3,7)
tree = insercao(tree,3,15)
tree = insercao(tree,3,12)
tree = insercao(tree,3,5)
tree = insercao(tree,3,6)
tree = insercao(tree,3,1)
tree = insercao(tree,3,2)
tree = insercao(tree,3,20)
tree = insercao(tree,3,26)
tree = insercao(tree,3,10)
```

Figura 3: Sucessivas inserções manuais.

O resultado é dado por

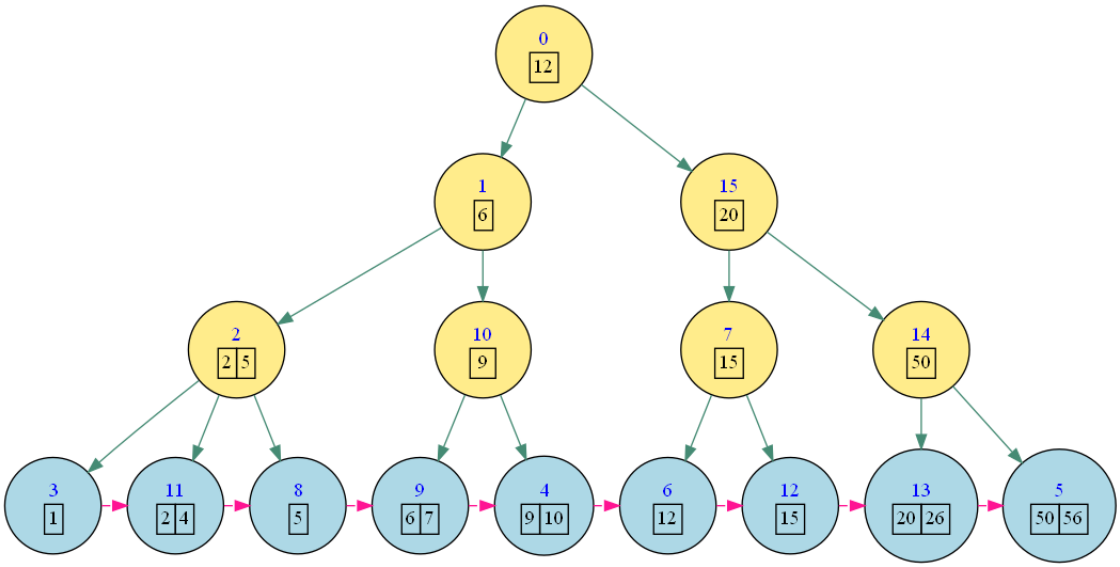

Figura 4: Resultado das sucessivas inserções manuais.

1.5.2 Deleção

Para deletar um valor da árvore, utiliza-se o comando *exclusão*(*tree*, *grau*, *valor_a_excluir*) que retornará a nova árvore com o valor *valor_a_excluir* deletado. A função de deleção recebe 3 parâmetros: a árvore com que se deseja fazer a operação, o grau dela e valor a ser excluído. No exemplo abaixo, foi utilizado grau 3 e foi solicitado a exclusão do valor 5.

```
tree=exclusão(tree,3,5)
```

Figura 5: Deleção manual numa árvore.

1.5.3 Busca

Para efetuar a busca de um valor específico numa árvore, utiliza-se o comando *busca*(*tree*, *valor*). Tal comando retornará o nó em que *valor* foi localizado. Caso não seja encontrado, a função retornará *Não existe!*.

```
f=busca(tree,10)
show(tree, key=f)
```

Figura 6: Busca manual.

Para visualizar o nó encontrado na árvore, pode-se atribuir ao parâmetro opcional *key* o nó encontrado, pois ele mostrará a árvore com o nó *key* destacado em vermelho, conforme exemplificado a seguir, numa situação de busca do valor 10, sendo $f = 4$.

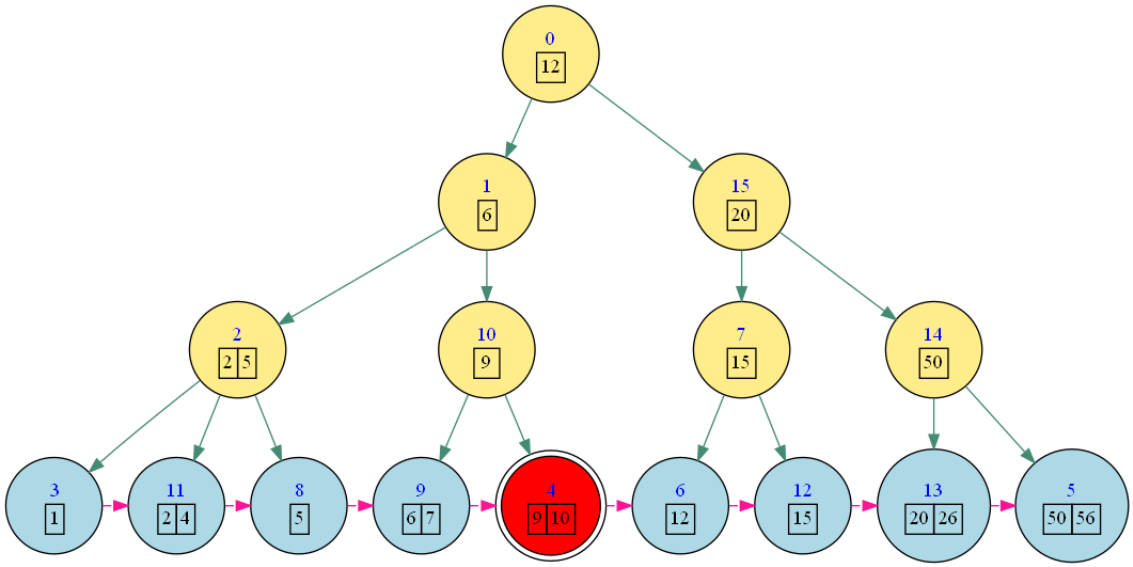

Figura 7: Resultado da busca manual

1.6 Operações em Lote usando escrita de comandos

O usuário pode fazer as operações de Inserção e Deleção em lote, conforme é discutido a seguir.

1.6.1 Inserção em Lote

O usuário pode fazer inserções em lote utilizando o comando *insere_aleatoriamente*(). Ele recebe 5 parâmetros: árvore em que se deseja fazer operações, grau, quantidade de valores a serem inseridos, valor mínimo e valor máximo. As figuras a seguir mostram a sucessão de inserções resultante.

```
tree = cria_arv()
tree=insere_aleatoriamente(tree,3,5,1,30)
```

Figura 8: Inserção em Lote manual

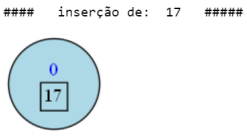

Figura 9: Inserção do 17.

Ao final, é apresentado o tempo de execução das inserções em lote, conforme mostrado na Figura 14.

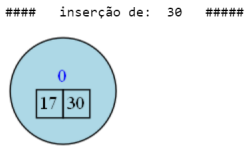

Figura 10: Inserção do 30.

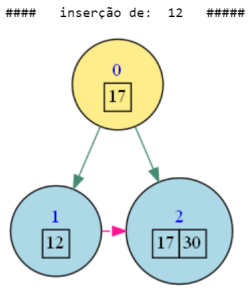

Figura 11: Inserção do 12.

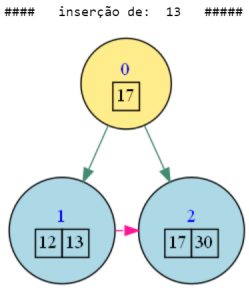

Figura 12: Inserção do 13.

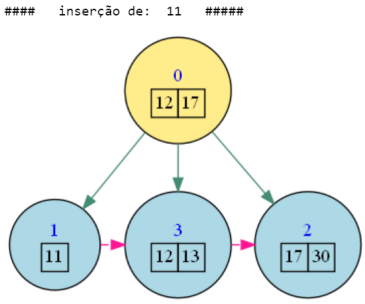

Figura 13: Inserção do 11.

```
--- 0.23145365715026855 seconds ---
```

Figura 14: Tempo de execução das inserções em lote.

1.6.2 Deleção em Lote

O usuário pode fazer deleções em lote utilizando o comando *remove_aleatoriamente*(). Ele recebe 3 parâmetros: árvore que se deseja fazer operações, grau, quantidade de valores a serem deletados.

```
tree=remove_aleatoriamente(tree,3,3)
```

Figura 15: Deleção em Lote manual.

As figuras a seguir mostram a sucessão de exclusões resultante.

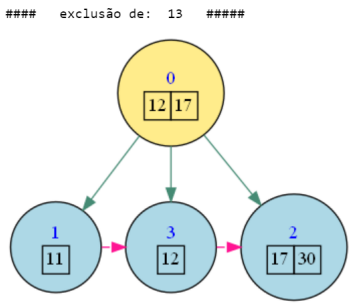

Figura 16: Deleção do 13.

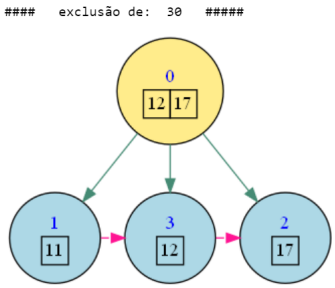

Figura 17: Deleção do 30.

Ao final, também é apresentado o tempo de execução das deleções em lote.

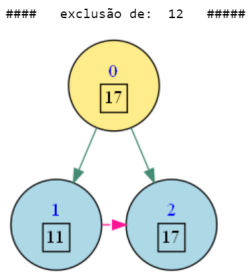

Figura 18: Deleção do 12.

1.7 Interface Interativa

O usuário pode fazer operações utilizando uma interface interativa que será executada ao final do código.

1.7.1 Criação

Na aba de criação, o usuário insere o grau da árvore e clica no botão *Criar* para criar uma nova árvore. Os graus podem variar entre 3 e 10.

Figura 19: Aba de Criação.

1.7.2 Inserção

Na aba de inserção, o usuário insere o valor que deseja inserir e clica no botão *Inserir*.

Figura 20: Aba de Inserção.

1.7.3 Inserção em Lote

Na aba de Inserção em Lote, o usuário insere a quantidade de valores que deseja adicionar e, em seguida, especifica o intervalo inserindo os valores mínimo e máximo. Por fim, clica no botão *Inserir*.

Figura 21: Aba de Inserção em Lote

1.7.4 Deleção

Na aba de Deleção, o usuário insere o valor que deseja excluir e clica em *Excluir*.

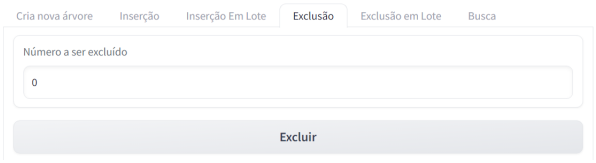

Figura 22: Aba de Deleção

1.7.5 Deleção em Lote

Na aba de Deleção em Lote, o usuário insere a quantidade de valores que deseja excluir e clica em *Excluir*.

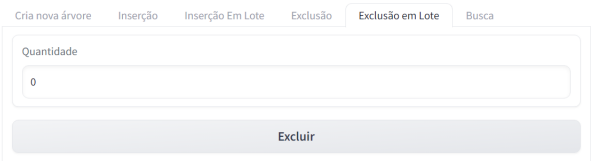

Figura 23: Aba de Deleção em Lote.

1.7.6 Busca

Na aba de Busca, o usuário insere o valor que deseja buscar e clica em *Buscar*. No campo nó será mostrado o nó em que o valor foi localizado, caso tenha sido. Caso contrário, será mostrada a frase *Não existe!*.

Em todas as operações realizadas na interface interativa será retornada graficamente a árvore com as modificações solicitadas.

Ao final das operações aparece um contador de passos e o tempo de execução, conforme mostrado na Figura 25.

Além disso, é possível ver um GIF apresentando as modificações feitas na árvore, basta usar o comando *gif()*, conforme mostrado na Figura 26.

Figura 24: Aba de Busca.

```
##### Inserção de  [414, 805, 525, 437, 219, 989, 853, 417, 421, 182, 97, 746]  ####
Contador 34
Timer 0.005299899999954505
##########################
```

Figura 25: Contador e Timer.

```
gif()
```

Figura 26: Comando para produzir o GIF.