Universidade de Coimbra
Faculdade de Ciências e Tecnologia da Universidade de Coimbra
Departamento de Engenharia Informática

Algoritmos e Estruturas de Dados

1º Projeto Prático
Comparação entre estruturas de dados contendo a percentagem de população com acesso a rede elétrica

Trabalho realizado por:
- Francisco Pinto Aires Basto, nº 2013138182, fbasto@student.dei.uc.pt
- João Correia Campos, nº 2013138305, jccampos@student.dei.uc.pt
- Rui Pedro Ferreira Mendes, nº 2013135669, rpmendes@student.dei.uc.pt

Introdução

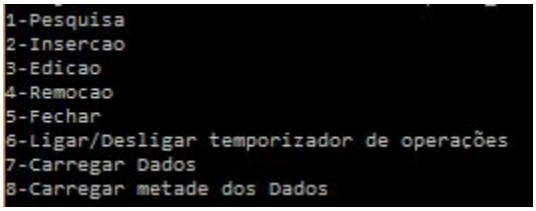
Como o objetivo do trabalho é a análise do desempenho de várias operações executadas em estruturas de dados diferentes, escolheu-se estruturas de dados que permitissem a observação das diferenças e melhorias de cada estrutura em relação à outras. Para tal, implementou-se as seguintes estruturas: Lista Ligada, Árvore Binária de Pesquisa e Árvore AVL, de forma a que cada uma fosse geralmente melhor do que a anterior respetivamente. É, portanto, expectável que no geral a Árvore AVL apresente melhores resultados do que a Árvore Binária de Pesquisa e que esta apresente melhores resultados do que a Lista Ligada, embora haja casos específicos em que tal não acontece já que, por exemplo, na inserção a Lista Ligada tem uma complexidade temporal melhor do que as outras.

Abordagens

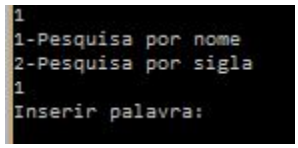
Escolheram-se 3 abordagens diferentes para realizar este estudo, nas quais foram criadas três estruturas principais, uma por cada abordagem, de modo a poder armazenar e manipular a informação disponibilizada. Como foi referido anteriormente, criaram-se duas listas ligadas, duas árvores de pesquisa binária e duas árvores AVL para os nomes dos países e para as respectivas siglas. Cada nó contém um ponteiro para a outra estrutura para termos uma ligação entre o nome do país e a sigla bem como o valor (*key* - nome do país ou sigla). A estrutura do nome tem ainda ligação à lista ligada auxiliar em que cada nó representa um ano e guarda a respetiva percentagem previamente retirada do ficheiro csv.

Interface

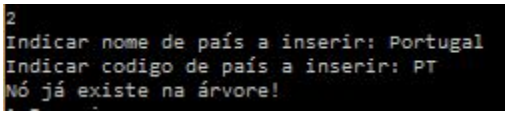
Foi criada uma interface funcional, de modo a poder efectuar operações sobre as estruturas de dados (pesquisa, inserção, edição,remoção):



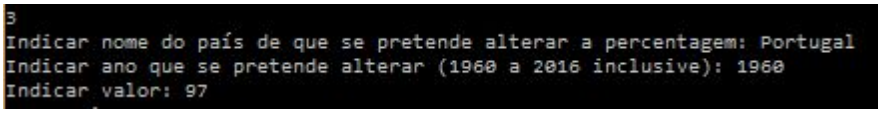
Opção 1:



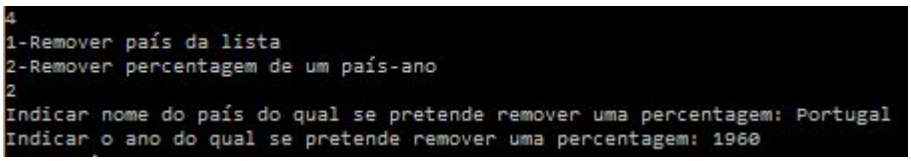
Opção 2:



Opção 3:



Opção 4:



Descrição dos testes

Criaram-se diferentes ficheiros de teste baseados ou na informação disponibilizada no ficheiro csv ou em países fictícios gerados aleatoriamente de modo a se poder efectuar testes com maior volume e tirar conclusões sobre os resultados obtidos.

CarregarDados: Inserir todos os países com todos os seus dados (nome, código e percentagens) do ficheiro *dados.csv*.

CarregarMetadeDados: Inserir metade (132 de 263) dos países com todos os seus dados (nome, código e percentagens) do ficheiro *dados.csv*.

Insere1000: Insere mil países fictícios (gerados automaticamente).
Insere10000: Insere dez mil países fictícios (gerados automaticamente).
Insere20000: Insere vinte mil países fictícios (gerados automaticamente).

Pesquisa264: Pesquisa por cada um dos países existentes no ficheiro *dados.csv*.
Pesquisa4x264: Pesquisa quatro vezes por cada um dos países existentes no ficheiro *dados.csv*.
Pesquisa40x264: Pesquisa quarenta vezes por cada um dos países existentes no ficheiro *dados.csv*.

Remocao264: Remove cada um dos países existentes no ficheiro *dados.csv*.

InsereRemove1000: Insere mil países fictícios (gerados automaticamente) e remove-os de seguida.
InsereRemove10x1000: Insere mil países fictícios (gerados automaticamente) e remove-os de seguida, 10 vezes.

Edicao264: Edita um valor percentual de cada país existente no ficheiro *dados.csv* de um ano aleatório (entre 1960 e 2016) para um valor aleatório (entre 0 e 100).
Edicao4x264: Edita um valor percentual de cada país existente no ficheiro *dados.csv* de um ano aleatório (entre 1960 e 2016) para um valor aleatório (entre 0 e 100) quatro vezes.
Edicao40x264: Edita um valor percentual de cada país existente no ficheiro *dados.csv* de um ano aleatório (entre 1960 e 2016) para um valor aleatório (entre 0 e 100) quarenta vezes.

Estudo de performances

DADOS	**Carregar dados**	**Carregar metade dos dados**
Lista Ligada	0.2811980247	0.1400980949
Árvore Binária de Pesquisa	0.2611794472	0.1320939064
Árvore AVL	0.2822170258	0.1480915546

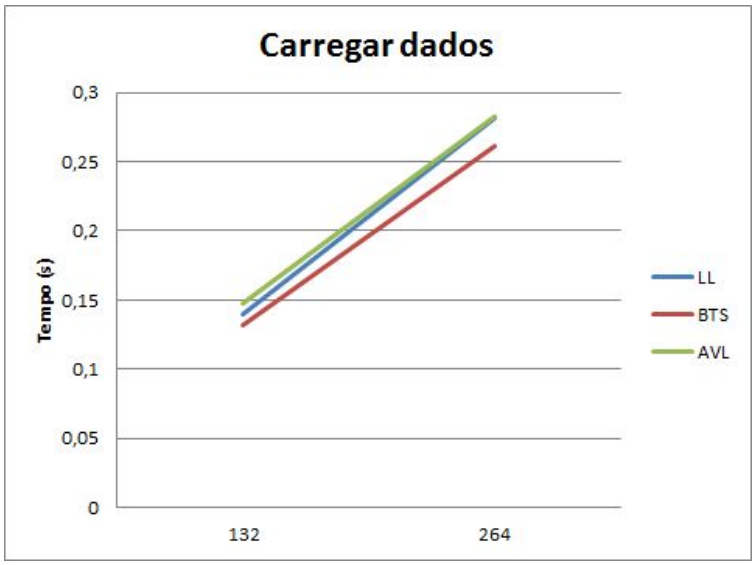
PESQUISA	**Pesquisa264**	**Pesquisa4x264**	**Pesquisa40x264**
Lista Ligada	0.3442509079	1.0567343235	11.3119938374
Árvore Binária de Pesquisa	0.3072156706	0.9926831722	10.4123361111
Árvore AVL	0.3092181683	0.9396626949	9.9790301323

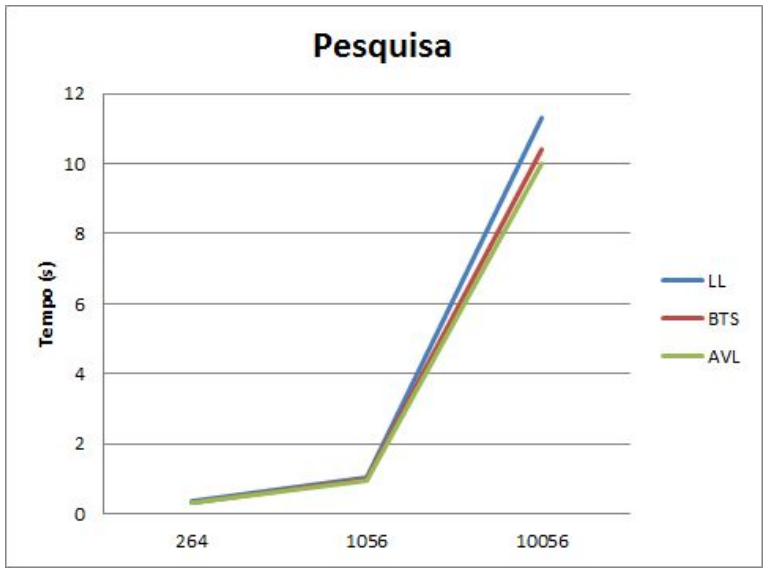
INSERÇÃO	**Insere1000**	**Insere10000**	**Insere20000**
Lista Ligada	0.3872609138	4.7963845730	9.7118794918
Árvore Binária de Pesquisa	0.3982818127	4.5421936512	8.7121305466
Árvore AVL	0.4673175812	5.9231667519	12.1014475822

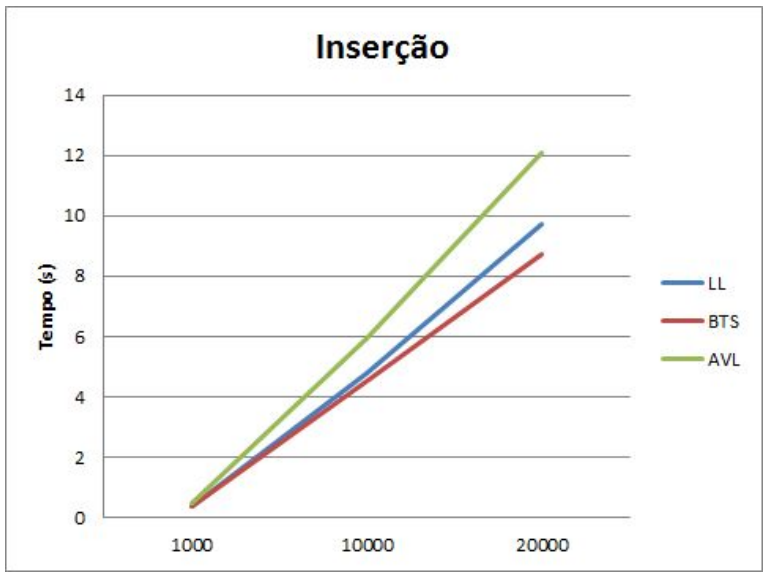
EDIÇÃO	**Edicao264**	**Edicao4x264**	**Edicao40x264**
Lista Ligada	0.1240870953	0.4923865032	4.9918809223
Árvore Binária de Pesquisa	0.1000537872	0.3852710724	4.0378508568
Árvore AVL	0.0970551968	0.3922657967	3.9477868080

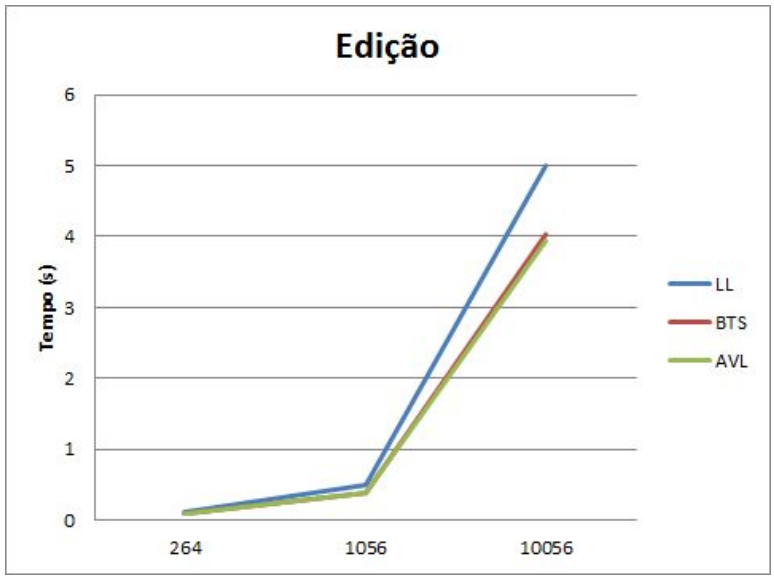
REMOÇÃO	**Remocao264**	**InsereRemove1000**	**InsereRemove10x1000**
Lista Ligada	0.1490905285	1.3249185085	14.0629527569
Árvore Binária de Pesquisa	0.0880491734	0.7445263863	7.7664687634
Árvore AVL	0.0890634060	0.8896155357	8.5990567207

tempo é medido em segundos


Carregar dados
0,3
0,25
0,2
0,15
0,1
0,05
0
Tempo (s)
132
264
LL
BTS
AVL


Pesquisa
12
10
8
6
4
2
0
Tempo (s)
264
1056
10056
LL
BTS
AVL


Inserção
14
12
10
8
6
4
2
0
Tempo (s)
1000
10000
20000
LL
BTS
AVL



Análise dos Testes

- Pesquisa

A árvore de pesquisa binária tem um tempo de execução de O(log N), pressupondo uma inserção aleatória de chaves que seja favorável ao balanceamento da árvore, sendo que no pior dos casos piore até O(n), caso as chaves tenham sido inseridas de forma ordenada (em ordem crescente ou decrescente).

Como as árvores AVL são balanceadas (filhos de um nó específico apenas diferem no máximo de uma altura de 1), apenas no pior dos casos possuem um tempo de execução de O(log N). Obtém por isso resultados semelhantes aos que a árvore binária teria na melhor das hipóteses, daí serem melhores do que esta.

Já as listas ligadas têm uma complexidade temporal de O(n), sendo esperado que estas obtenham performance inferior às árvores.

Analisando os resultados obtidos, podemos concluir que vão de encontro ao que seria esperado, observando que árvores AVL possuem os melhores valores, no entanto idênticos à binária, que por sua vez são melhores que na lista. À medida que os inputs vão crescendo significativamente, os resultados também se vão diferenciando um pouco mais.

- Inserção

Ao analisar a operação de inserção esperava-se que a lista ligada apresentasse bons resultados já que a sua complexidade temporal é de O(1) visto que esta estrutura favorece bastante a operação de inserção em detrimento das outras ao permitir inserir na cabeça da estrutura sem preocupações em ordenar.

Quanto às árvores, os tempos da árvore binária variam novamente entre O(n) e O(log N) enquanto a AVL tem uma complexidade temporal de O(log N).

Os testes de carregar o ficheiro dados.csv não permitem tirar conclusões já que os valores são todos bastante parecidos, no entanto os testes mais pesados de inserção já permitem uma análise melhor. Ao contrário do esperado, a Lista Ligada apresenta valores mais elevados do que a árvore binária. De resto, como esperado os valores da árvore AVL pioram exponencialmente com o número de inserções já que a cada inserção a árvore pode ter de se rebalancear com uma até três rotações enquanto que a árvore binária apresenta bons resultados explicados por esta não ter preocupações com o equilíbrio.

- Edição

Esta operação divide-se em dois passos: pesquisa e atribuição de novo valor. Como tal, perspetiva-se que os testes apresentem resultados claramente semelhantes aos da operação de pesquisa.

Confirmou-se que, ao comparar os valores entre as estruturas, os testes apresentaram resultados parecidos com os da operação de pesquisa e constatou-se que, novamente, fazem sentido tendo em conta as complexidades temporais da operação nas respetivas estruturas.

- Remoção

Esta operação divide-se em dois passos: pesquisa pelo nó a remover seguida pela sua remoção. Espera-se, portanto, que os resultados apresentem novamente resultados que reflitam os tempos de pesquisa. No caso da Árvore AVL, a pesquisa será mais rápida mas pode ser necessário um rebalanceamento logo espera-se que a Árvore Binária, apesar da sua pesquisa ter pior desempenho do que a AVL, apresente melhores resultados para os casos mais pesados devido à necessidade desta se re-equilibrar (como na inserção).

Esta operação revelou-se difícil de analisar pela sua relação com as outras operações. No caso da lista ligada a remoção é mais demorada fruto da sua lenta pesquisa enquanto que no caso das árvores a pesquisa é mais rápida e os resultados obtidos enquadram-se neste cenário.

Como cada estrutura inicia apenas com os valores contidos no ficheiro csv não é possível remover mais do que 264 países sem antes proceder a nova inserção. Por isso, para testar mais remoções criou-se inputs em que se insere um certo número de nós para depois já ser possível removê-los. Daí que ao ter que inserir nós, pesquisá-los e só depois removê-los os valores a obter sejam mais favoráveis às árvores do que à lista ligada. No entanto, isto não nos permitiu tirar boas conclusões quanto à remoção.

Conclusão

Podemos concluir que a árvore AVL apresentou, no geral, melhores resultados do que as outras abordagens e que se justifica o sacrifício no tempo de execução da inserção, onde esta é mais lenta para equilibrar os seus elementos e melhorar a performance nas outras operações. Consideramos que a escolha das abordagens foi correta dado que as diferenças entre elas foram visíveis e permitiram tirar conclusões já que os resultados fazem sentido e estão de acordo com o que seria de esperar das suas complexidades temporais.

Bibliografia/Fontes

https://www.tutorialspoint.com/python/python_lists.htm

https://pythonhelp.wordpress.com/2015/01/19/arvore-binaria-de-busca-em-python/

https://docs.python.org/2/library/csv.html

https://docs.python.org/3/library/csv.html

http://bigocheatsheet.com/

Código fornecido pelo professor nos slides

Slides teóricos sobre complexidades do ano anterior