Relatório do 1º Projeto Prático

Algoritmos e Estruturas de Dados 2016/2017

José Ferreira, 2014192844

Bruno Batanete, 2014203839

Introdução

O objetivo deste trabalho é analisar e comparar a evolução do acesso à rede elétrica na população mundial. Para isso consultamos um documento disponibilizado pelo *World Bank* que contém esses dados relativos a um determinado número de países.

A fim de tratar a informação disponibilizada, construimos 3 estruturas de dados , uma árvore de listas, árvores de árvores e lista ligada de árvores para obter os mesmos resultados através de estruturas de dados diferentes.

Realizamos o trabalho em *Python*, na sua versão mais recente, de modo a que reutilizar o código fornecido pelos docentes da disciplina. Todas as *sources* estão devidamente identificadas no código em si, sendo que maior parte é adaptado do código disponibilizado durante as aulas, assim como comentado de modo a facilitar a navegação e compreensão do mesmo.

Árvore de listas

Cada nó da árvore será um tuplo com o nome e a sigla do país e terá também uma lista ligada em que cada nó da lista terá outro tuplo com o ano e a percentagem de acesso à rede elétrica. Em termos de complexidade *Big-O*, esta estrutura de dados terá complexidade O(log(n) * n) para acesso e pesquisas, e no pior dos casos onde o item a procurar estará no ultimo ramo da árvore e no ultimo elemento da lista ligada, O(n)^2. Quanto à inserção e eliminação, como uma lista ligada terá complexidade O(1), teremos O(log(n)) no melhor e O(n) no pior dos casos.

Em relação à otimização do código, um exemplo de execução de cada função implementada:

- **Adicionar país novo** ~= 0,070 ms
- **Adicionar dados a um país** ~= 0.370 ms
- **Remover país** ~= 32.830 ms
- **Remover dados a um país** ~= 0.421 ms
- **Procurar todos os dados de um país** ~= 0.423 ms
- **Procurar data de um país específico** ~= 0.412 ms
- **Procurar todos os dados para um ano** ~= 2.910 ms

Notas: os valores usados são calculados através da média de três execuções de cada função e não refletem um valor base, pois este varia de uma forma gradual. O valor elevado da remoção de um nó da árvore principal tem em conta com o facto de a árvore ter de ser totalmente balanceada, novamente.

Conclusões

Podemos verificar que esta solução tem vantagens e desvantagens, sendo que as vantagens são que é eficiente para a inserção, remoção e procura, exceto quando se remove um nó da árvore principal ou quando se pretende imprimir todos os valores para um determinado ano.

Lista de árvores

Esta estrutura é o inverso da Árvore de listas, existindo apenas uma lista principal, em que cada nó terá uma árvore. Desta vez cada nó da lista principal terá o nome e a sigla do país armazenado num tuplo e será o nó da cada árvore do país a ter a informação sobre a percentagem de acesso à rede elétrica e o ano em questão.

Para a complexidade,em média e no pior dos casos, será semelhante à Árvore de listas,O(log (n)*n) . Quanto mais próximo da *head* da lista for o item a tratar, melhores resultados se irão obter. As operações possíveis sobre a estrutura têm diferentes resultados temporais, dependendo se for um pais/sigla, ou de um ano/percentagem uma vez que as operações feitas em árvores em média apresentam tempos melhores ou iguais às feitas em listas ligadas.

Em relação à otimização do código, um exemplo de execução de cada função implementada:

- **Adicionar país novo** ~= 0.012 ms
- **Adicionar dados a um país** ~= 0.103 ms
- **Remover país** ~= 0.063 ms
- **Remover dados a um país** ~= 0.008 ms
- **Procurar todos os dados de um país** ~= 0.250 ms
- **Procurar data de um país específico** ~= 0.021 ms
- **Procurar todos os dados para um ano** ~= 1.930 ms

Nota: os valores usados são calculados através da média de três execuções de cada função e não refletem um valor base, pois este varia de uma forma gradual.

Conclusões

Esta solução, para a quantidade de valores disponibilizados no ficheiro csv, tende a ser a melhor pois assim como dito na complexidade, a remoção de um no da lista ligada faz-se de forma mais eficiente, contudo, com uma maior quantidade de dados, há tendência de piorar na sua execução.

Árvores de árvores

Nesta estrutura temos uma árvore com os países onde cada nó contém um tuplo do nome e da sigla do país e uma sub-árvore com cada nó com informação dos anos e das percentagens de eletricidade de cada nó da árvore principal.

Esta estrutura é a que apresenta melhores valores se estivermos a ter em conta o tempo de execução, para uma grande quantidade de valores de energia.

A complexidade temporal para esta estrutura é de O(log (n) * log (n)) o que implica que as operações possíveis serão mais eficazes com esta estrutura uma vez que as árvores são as melhores estruturas de dados para as operações de pesquisa, remoção, inserção tendo o contra de que quando se remove ou adiciona um país tem de se ver sempre se a árvore está equilibrada e se não estiver terão de se fazer uma série de operações até ser de novo uma árvore AVL.

Em relação à otimização do código, um exemplo de execução de cada função implementada:

- **Adicionar país novo** ~= 0,049 ms
- **Adicionar dados a um país** ~= 0.372 ms
- **Remover país** ~= 23,000 ms
- **Remover dados a um país** ~= 0.381 ms
- **Procurar todos os dados de um país** ~= 0.42 ms
- **Procurar data de um país específico** ~= 0.41 ms
- **Procurar todos os dados para um ano** ~= 2.91 ms
-

Notas: os valores usados são calculados através da média de três execuções de cada função e não refletem um valor base, pois este varia de uma forma gradual. O valor elevado da remoção de um nó da árvore principal tem em conta com o facto de a árvore ter de ser totalmente balanceada, novamente.

Conclusões

Assim como nas árvores de listas, podemos verificar que esta solução tem as mesmas desvantagens porque se trata de uma uma árvore como estrutura principal, no entanto, com o aumento de dados guardados no ficheiro, esta estrutura aparenta ser a mais eficiente, pois se trata da junção de duas estruturas com complexidade O(log(n)) para qualquer o caso.

Conclusão

As estruturas de dados utilizadas para a resolução deste projeto, mostraram uma performance semelhante, tirando a função de remoção de um nó de uma árvore, que aparenta demorar a balancear a árvore. No entanto, com o aumento da informação inserida em cada estrutura, uma Árvore de árvores aparenta ser a solução mais eficiente em média, sendo que algumas das opções serão mais eficientes com uma das outras estruturas. Ainda asssim, podendo utilizar uma estrutura auxiliar, poderá ser possível otimizar ainda mais a procura de todos os valores de um ano específico, de modo a não ter que percorrer as duas estruturas em cada solução apresentada.