Relatório Trabalho Final de Estrutura de Dados: Contando Calorias

Aluno: Isaac Bueno do Canto
Matrícula: 335218
Turma: B

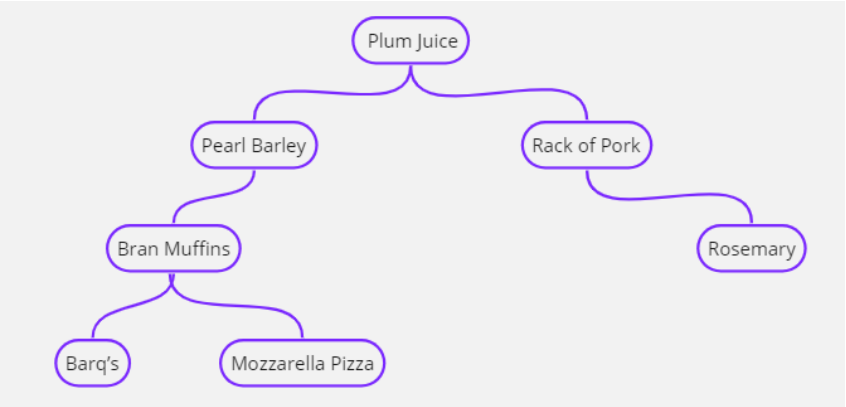
Objetivo: Comparar o desempenho das Árvores Binárias de Pesquisa em uma aplicação de contagem de calorias.

Problema: Uma nutricionista acompanha diversos pacientes em dieta. A fim de ajudá-los a controlar sua alimentação, ela pede que eles registrem em um arquivo todos os alimentos que ingerem diariamente. Ao final da semana, os pacientes enviam todos os arquivos para ela que precisa então calcular o total de calorias ingeridas por eles dia a dia. Para esse cálculo, ela utiliza uma tabela que possui os nomes dos alimentos e a quantidade de calorias que eles possuem (em uma porção de 100 gramas). Este é um processo trabalhoso e sujeito a erros. Por isso, ela decidiu automatizá-lo.

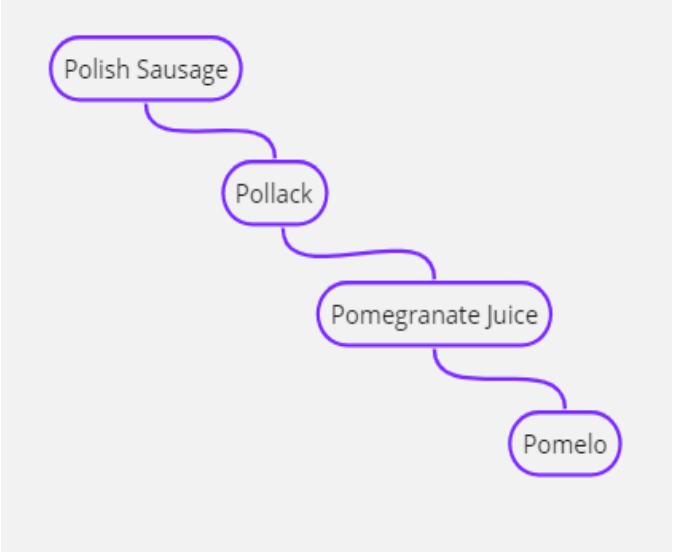
Funcionamento da aplicação (solução):

O programa recebe como entrada um arquivo .csv com a tabela de dados da nutricionista e outro .csv com os alimentos ingeridos e as respectivas quantidades. Então, salva os dados numa estrutura de árvore para fazer as devidas consultas e calcular as calorias ingeridas pelo paciente. Como saída, têm-se um arquivo com a quantidade de calorias ingerida por alimento e porção e o total do dia. Também, a fim de comparar o desempenho de duas árvores de pesquisa diferentes, ABP e AVL, a aplicação realiza as mesmas operações de carregar a tabela em ambas as árvores e consultá-las para comparar seu balanço final e número de comparações, cujas estatísticas também são escritas no arquivo de saída.

Quanto ao funcionamento da aplicação, ela baseia-se na utilização de uma estrutura de dados denominada Árvore Binária de Pesquisa. Nessa estrutura, os dados são armazenados seguindo uma ordem, nesse caso alfabética. Assim a busca torna-se pouco custosa quando comparada a outras estruturas como uma lista, em que no pior caso, quando está ao fim ou não existe, têm-se de checar todos os valores dela. Como no exemplo abaixo, um fragmento da própria base de dados da aplicação, cada nodo possui uma subárvore na esquerda e outra na direita, os da esquerda são os valores “menores” que o valor da chave, enquanto que a da direita contém os “maiores”.



Assim, para buscar o valor Rosemary, como em termos de ordem é maior que “Plum Juice”, a busca é realizada apenas na subárvore da direita e assim sucessivamente, tornando a busca pouco custosa. Nesse ponto, as operações de inserção e remoção de nodos em árvore também são contempladas, pois é necessário encontrar a posição certa do nodo ou o que se deseja remover para executá-las. Por isso, essas operações também são consideradas vantagens das ABPs.



Entretanto, para árvores de pesquisa comuns não é garantido que a inserção de nodos se dará de forma balanceada, podendo até em um pior caso serem inseridas em forma de zig-zag, como o exemplo acima, pois foram inseridas em ordem alfabética. Assim as vantagens de busca se perdem. Para tal, há tipos de árvores balanceadas, que inserem os nós de forma equilibrada reorganizando os anteriores para que a árvore esteja sempre balanceada, com o objetivo de otimizar as operações de consulta e diminuir o número médio de comparações.

Portanto, nessa aplicação foram utilizados 2 tipos de árvores de pesquisa a fim de comparar os resultados de desempenho, uma que mantém o balanceamento por altura, a AVL e outra não, a ABP. Ambas foram utilizadas de forma igual na aplicação a partir dos seguintes passos e funções:

```
pNodoA *abp = NULL;
pNodoA *avl = NULL;

char alimentos[LISTA][MAX_PALAVRA];
int valores_alimentos[LISTA][3];
int estatisticas_avl[6];
int estatisticas_abp[6];

abp = leArquivoParaArvore(argv[1], abp, 0);
consultaArquivoABP(argv[2], abp, estatisticas_abp, alimentos, valores_alimentos, 0);

avl = leArquivoParaArvore(argv[1], avl, 1);
consultaArquivoABP(argv[2], avl, estatisticas_avl, alimentos, valores_alimentos, 1);

escreveNoArquivo(argv[3], alimentos, valores_alimentos, estatisticas_abp, estatisticas_avl);

SetConsoleOutputCP(CPAGE_DEFAULT);
return 0;
```

pNodoA *leArquivoParaArvore(char *file_name, pNodoA *root, int type);

Essa função recebe como parâmetros o arquivo com todos os alimentos que se deseja armazenar na árvore, um ponteiro para a árvore em questão e o tipo, 0 para abp e 1 para avl, e retorna um ponteiro da árvore passada com os novos valores. A partir do tipo de árvore utiliza as funções padrão InsereABP ou InsereAVL. Nesse passo, são contabilizados as rotações realizadas para balancear a árvore, como em ABPs não são realizadas rotações ele se mantém 0 sempre.

int consultaArquivoABP(char *file_entrada, pNodoA *abp, int estatisticas[], char alimentos[][], int valores_alimentos[][], int isAvl);

Essa função recebe como parâmetros o arquivo com os alimentos diários do paciente, a árvore que se realizará a busca, um array para armazenar as estatísticas da árvore, um para guardar os alimentos e outro para os respectivos valores calculados, por fim um inteiro para indicar se ela é avl, para não precisar salvar os alimentos 2 vezes nos arrays. Assim, retorna 1 caso ocorreu um erro ao abrir os arquivos e 0 se não.

Dessa forma, a aplicação lê os valores para a ABP, realiza as consultas e salva nos arrays de alimentos e valores_alimentos, assim como as estatísticas da ABP no array correspondente. Depois lê o mesmos valores para a AVL, realiza a consulta também com os mesmos valores e salva as estatísticas no respectivo array. Por fim executa-se a função para escrever no arquivo de saída.

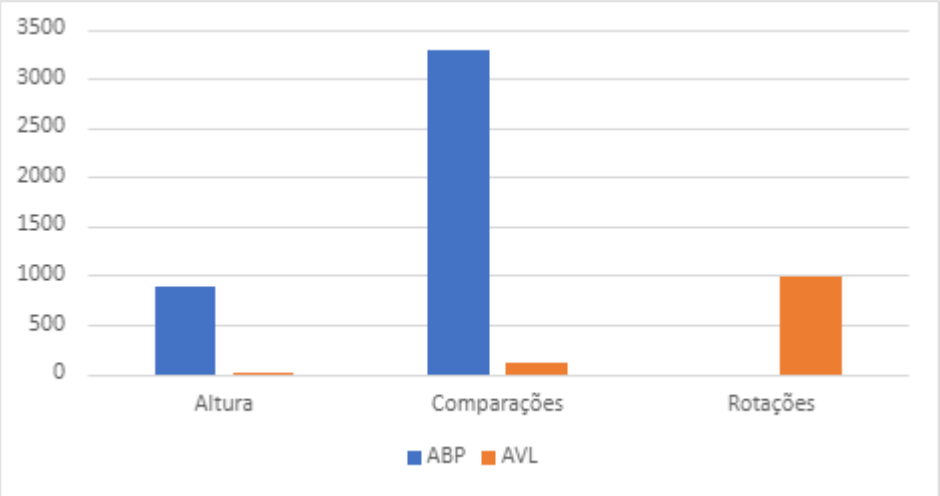
```
150g de activia (74 calorias por 100g) = 111 calorias
110g de tangerine (53 calorias por 100g) = 58 calorias
120g de bagel (257 calorias por 100g) = 308 calorias
200g de coffee (1 calorias por 100g) = 2 calorias
200g de beef fillet (189 calorias por 100g) = 378 calorias
150g de rigatoni (353 calorias por 100g) = 529 calorias
100g de banoffee pie (395 calorias por 100g) = 395 calorias
50g de arugula (25 calorias por 100g) = 12 calorias
5g de caesar dressing (429 calorias por 100g) = 21 calorias
30g de camembert (300 calorias por 100g) = 90 calorias
50g de chocolate (529 calorias por 100g) = 264 calorias
200g de white wine (82 calorias por 100g) = 164 calorias

Total de 2332 calorias consumidas no dia.
===========ABP===========
Numero de Nodos: 1000
altura: 893
Rotaçoes: 0
comparacoes: 3299
=========================

===========AVL=============
Numero de Nodos: 1000
altura: 10
Rotacoes: 986
comparacoes: 109
===========================

```

A saída consiste no exemplo acima, no qual foi utilizado uma lista de valores bastante ordenada, o que tornou discrepante as estatísticas entre as árvores, como a altura, que influencia no número de comparações finais. Por isso, a fim de exemplificar essa diferença, torna-se interessante comparar diferentes casos de listas de valores para comparar devidamente essas 2 árvores nas tabelas a seguir.

Arquivo de dados em ordem alfabética:

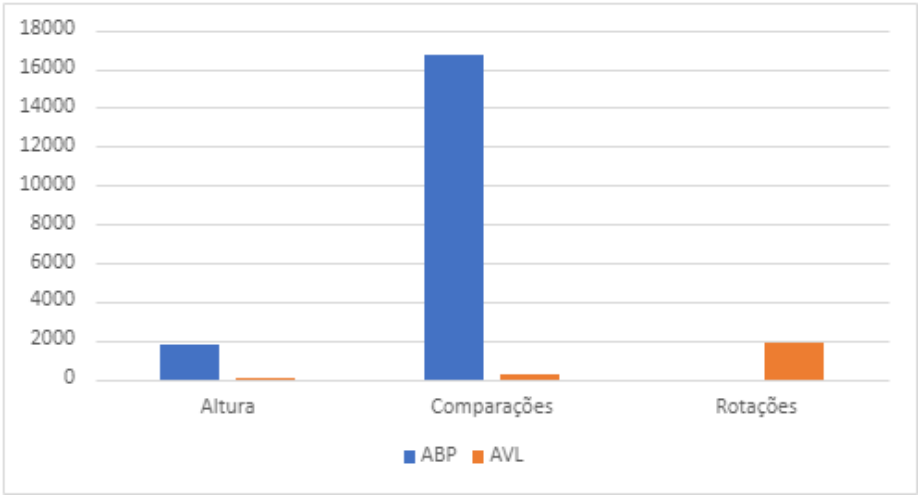
Tratando-se de valores em sua maioria ordenados, as ABPs mostram-se ineficientes, haja vista a imensa altura em comparação com a quantidade de nodos, e uma imensa quantidade de comparações. Com a AVL se reduziu a altura em 98% e o número de comparações em 96%. Mesmo que se tenha realizado 986 rotações para balancear a árvore, isso é realizado apenas na inserção dos valores da base de dados, o que faz compensar ao comparar com o número de comparações para apenas 12 buscas.

1000 valores ordenados / entradas do dia 1			
	ABP	AVL	Otimização %
Nodos	1000	1000	-
Altura	893	10	▼98%
Comparações	3299	109	▼96%
Rotações	0	986	-



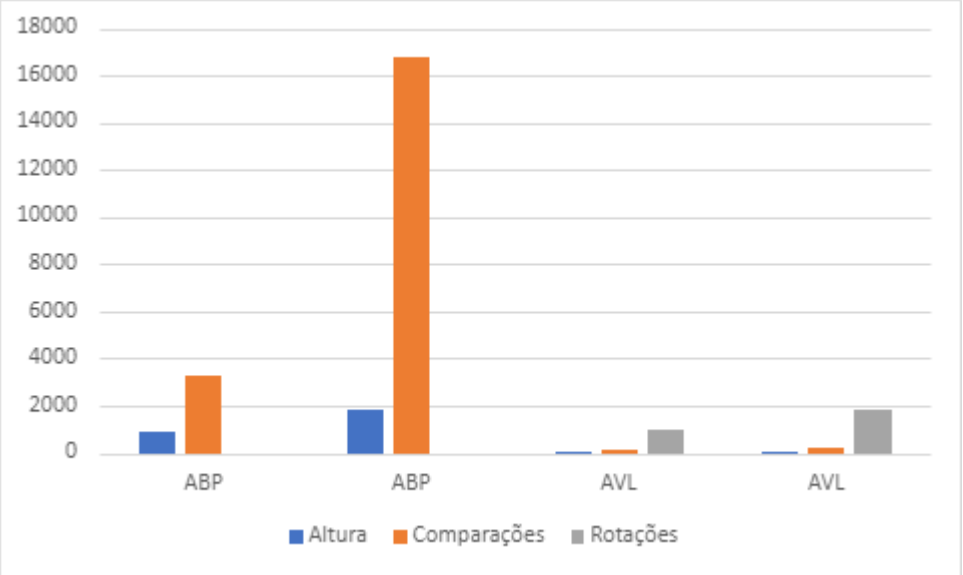
O mesmo acontece para as 20 entradas do dia 2, a partir de um arquivo com 1993 valores. Nesta tabela é interessante notar que praticamente dobrou-se a quantidade de nodos da base de dados anterior, e a AVL só aumentou a altura em duas unidades, enquanto a ABP possui 1000 novos níveis. As comparações também se mantêm uniformes, com quase o dobro, enquanto que da ABP aumentou em 5 vezes, mesmo que sejam entradas de dias diferentes, é um grande aumento para menos que o dobro de novas informações.

1993 valores ordenados / entradas do dia 2			
	ABP	AVL	Otimização %
Nodos	1993	1993	-
Altura	1814	12	▼99,3%
Comparações	16742	200	▼98,8%
Rotações	0	1869	-



Por fim, a comparação final relativa aos comentários anteriores, levando em consideração quantidades de nodos, comparando os resultados entre os mesmos tipos de árvores e com as demais. As discrepâncias tornam-se ainda mais claras no gráfico, tanto entre as árvores quanto entre a ABPs com 1000 e 1993 nodos, à direita.

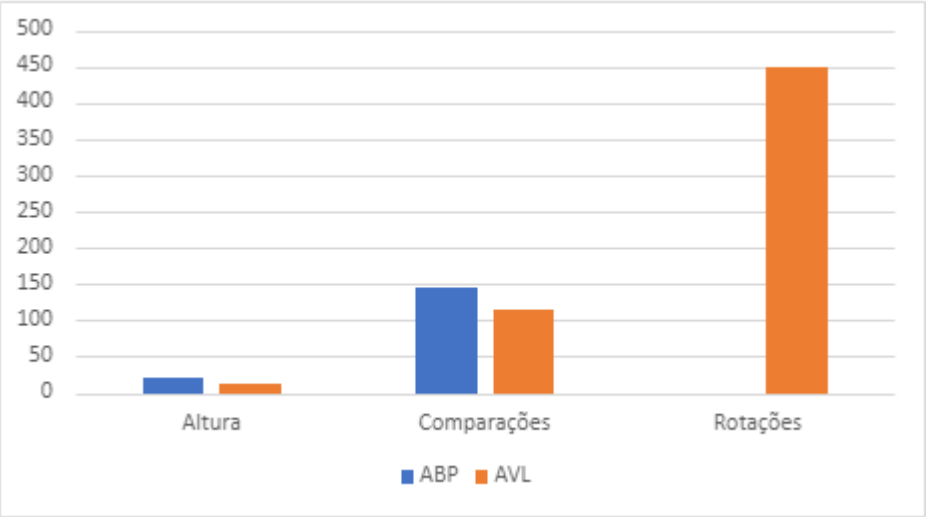
Valores Ordenados				
	ABP	ABP	AVL	AVL
Nodos	1000	1993	1000	1993
Altura	893	1814	10	12
Comparações	3299	16742	109	200
Rotações	0	0	986	1869



Arquivo de dados desordenados:

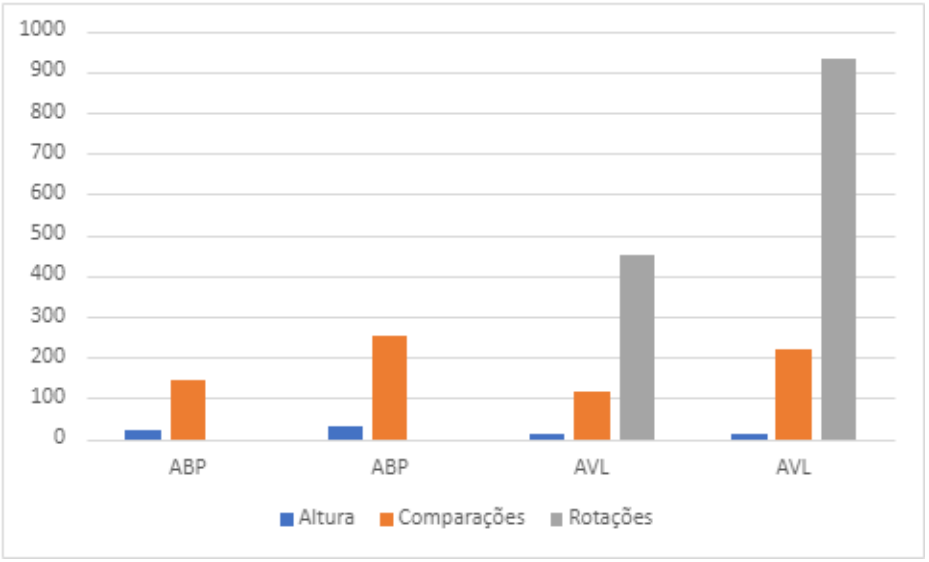
Tratando-se de valores desordenados, os valores já não são tão discrepantes e as ABPs podem ser eficientes. A entrada de 1000 valores na árvore é a mesma da anterior, assim como os 12 alimentos do dia 1. Agora, entre a ABP e a AVL, a altura reduziu quase pela metade e o número de comparações reduziu em 20%, o que nesse caso de poucas buscas não se torna tão considerável dadas as 449 rotações que a AVL teve de realizar, que é um número que se sobressai nesse gráfico.

1000 valores desordenados / entradas do dia 1			
	ABP	AVL	Otimização %
Nodos	1000	1000	-
Altura	22	12	▼45%
Comparações	144	114	▼20%
Rotações	0	449	-

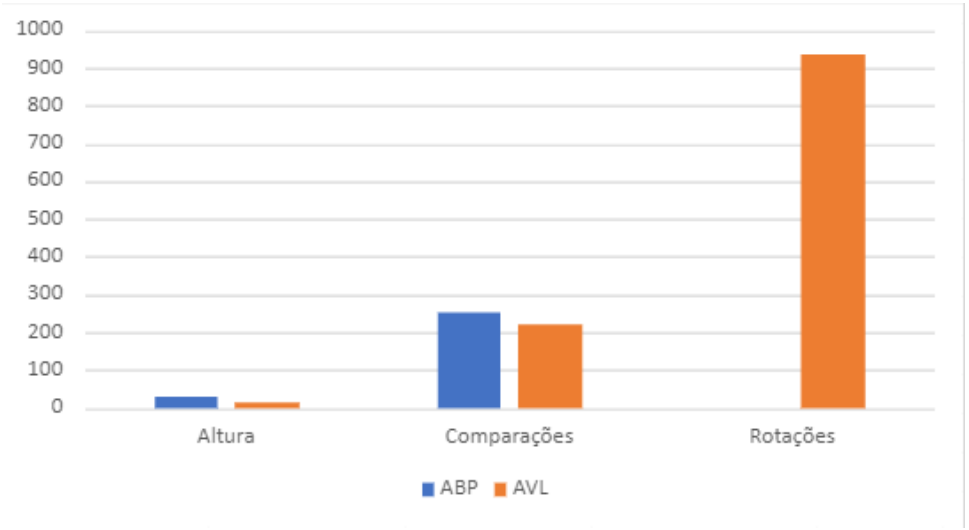


Aumentando a entrada de valores na árvore para 1993 e os alimentos do dia 2 para 20, os valores se mantêm bastante uniformes aos 1000 de antes. Ambas aumentaram em poucos níveis a altura mesmo com quase o dobro de entradas e as comparações aumentaram uniformemente aos novos valores, ainda que a AVL sempre se mantenha mais eficiente. É interessante notar que os gráficos são praticamente iguais com a diferença da escala.

1993 valores desordenados / entradas do dia 2			
	ABP	AVL	Otimização %
Nodos	1993	1993	-
Altura	29	13	▼55%
Comparações	253	219	▼13%
Rotações	0	934	-



Então, a comparação final entre as árvores com entradas desordenadas, em que se sobressai a quantidade de rotações que a AVL tem de fazer, embora as outras estatísticas não difiram tanto entre elas como anteriormente.



Por fim, é importante destacar a diferença nos valores de otimização para valores desordenados. É nesse ponto que mora a grande diferença entre essas duas árvores: no pior caso. Como dito anteriormente, ABPs não são balanceadas então no pior caso, de inserir dados ordenados, os nodos são inseridos quase como um zig-zag. Para isso existe a AVL que reduziu em 99,3% a altura da árvore para a mesma quantidade de nodos e isso representou uma redução de 98,8% no número de comparações. Demonstrando sua incrível eficiência apesar de serem 2 árvores com praticamente as mesmas propriedades, a única diferença que mora na altura já representou números tão grandes.

1993 valores ordenados / entradas do dia 2			
	ABP	AVL	Otimização %
Nodos	1993	1993	-
Altura	1814	12	▼99,3%
Comparações	16742	200	▼98,8%
Rotações	0	1869	-