Universidade Federal do Rio Grande do Sul

Estruturas de Dados 2018/2 – Árvores Binárias de Pesquisa

Profª Viviane Pereira Moreira do Instituto de Informática
Bruna Casagranda Cagliari – cartão 00290515.
Frederico Messa Schwartzhaupt – cartão 00304244.

1. INTRODUÇÃO

Este relatório tem o objetivo de analisar o desempenho do fator de balanceamento AVL, em árvores binárias de pesquisa. Para a comparação, foi realizada a organização de um conjunto de palavras em dois dicionários, utilizando o método de balanceamento em somente um deles. O objetivo era verificar se de fato, a implementação do método de balanceamento, criado por Adelson-Velsky e Landis (A.V.L.), resultaria em uma redução significativa do tempo de execução, nos processos posteriores.

Utilizando as duas árvores, ABPs e AVLs, o presente trabalho busca atribuir valorações a frases disponibilizadas dentro de um arquivo texto. Obtida a valoração, concluímos a análise de sentimentos, também dita como mineração de opiniões, da opinião lida. A análise consiste em identificar o sentimento que os usuários apresentam a respeito de alguma entidade de interesse (um produto específico, uma empresa, um lugar, uma pessoa, dentre outros) baseado no conteúdo disponível. O objetivo principal é permitir que um usuário obtenha um relatório contendo o que as pessoas andam dizendo sobre algum item sem precisar encontrar e ler todas as opiniões e notícias a respeito.

2. ÁRVORES BINÁRIAS DE PESQUISA

2.1. Definição

Uma árvore binária de pesquisa (ABP) é um conjunto de registros usado para armazenamento e busca de dados. Os dados são alocados de acordo com a informação contida. Para as ABPs, os elos (nós que guardam as informações) são dispostos para direita ou esquerda: maiores para direita e menores para esquerda. O trajeto para determinar o lugar do novo nó inserido começa pela raiz e percorre a árvore, por meio da comparação de informações - maior ou menor, até encontrar o nó folha, onde é alocado.

Inicialmente, deve-se inicializar uma estrutura para armazenar os nós. É por meio desta que registramos os elos esquerdo e direito, que serão filhos do nó. Através da recursividade, é possível implementar uma função de inserção eficiente, de tal forma que cada nodo é visitado somente uma vez.

2.2. ABPs de Adelson Velsky e Landis (AVLs)

Entre as árvore binárias de pesquisa, as AVLs (Adelson-Velsky e Landis) são árvores de busca balanceadas pela altura. Uma árvore AVL é dita balanceada quando a diferença entre as alturas das sub-árvores de todos os seus nós não é maior do que um. Por sua vez, o balanceamento de um nó é definido como a altura – nível máximo de suas folhas – de sua subárvore esquerda menos a altura de sua subárvore direita.

O fator de balanceamento (FB) busca reduzir o número de operações necessárias para encontrar um elemento aleatório da árvore. Então, para garantir que a árvore permaneça balanceada, Adelson-Velsky e Landis criaram algoritmos de rotação para serem utilizados em conjunto com as operações básicas de inserção e remoção de ABPs, ocasiões onde a árvore pode se desbalancear.

Uma rotação é realizada no nodo que estiver com fator de balanceamento maior que 1 ou menor que -1. Para determinar qual das quatro rotações é necessária para rebalancear a árvore, tanto o sinal do FB do nodo quanto os sinais dos FBs de seus filhos direito e esquerdo são observados. A seguir, estão descritas as possíveis situações:

- Rotação simples à direita: FB do nodo é positivo e o fator da sua subárvore esquerda é positivo. Feita a rotação, o filho do nodo com FB desbalanceado se torna o pai, o nodo se torna o filho direito e o neto do nodo se torna o filho esquerdo.
- Rotação simples à esquerda: FB do nodo é negativo e o fator da sua subárvore direita é negativo. Feita a rotação, o filho do nodo com FB desbalanceado se torna o pai, o nodo se torna o filho esquerdo e o neto do nodo se torna o filho direito.
- Rotação dupla à direita (ou duas rotações à direita consecutivas): FB do nodo é positivo e o fator da sua subárvore esquerda é negativo. Feita a rotação, o neto do nodo com FB desbalanceado se torna o pai, o nodo se torna o filho direito e o neto do nodo se torna o filho esquerdo.
- Rotação dupla à esquerda (ou duas rotações à esquerda consecutivas): FB do nodo é negativo e o fator da sua subárvore direita é positivo. Feita

a rotação, o neto do nodo com FB desbalanceado se torna o pai, o nodo se torna o filho direito e o neto do nodo se torna o filho esquerdo.

A remoção em árvores AVL segue o mesmo procedimento realizado em árvores ABPs. Entretanto, o método de balanceamento e a necessidade de rotações ainda procedem para que a AVL esteja sempre balanceada. Porém, diferente da inserção que necessita somente uma ou nenhuma rotação, a remoção de um nó pode exigir mais de uma operação.

3. IMPLEMENTAÇÃO

3.1. Código

O algoritmo possibilita a escolha do usuário entre os dois tipos de árvores logo após a inicialização do executável. Dada a escolha instruída por um breve menu, o código prossegue fazendo a inserção das palavras encontradas no arquivo texto disponibilizado. No presente trabalho, o arquivo texto se trata de um conjunto de palavras com valoração que são usadas para fazer a análise de sentimentos de cada frase dentro de um texto dado por um arquivo texto, obtendo assim a análise de sentimentos do material.

Cada escolha do usuário leva o código a funções diferentes de inserção. Dentro de cada função de inserção, o algoritmo tem um contador que busca contar o número de comparações – até que todas palavras e suas respectivas valorações estejam inseridas – a fim de comparar a eficiência de ambas as árvores de pesquisa em relação a inserção de nodos.

Terminada esta etapa, a leitura do arquivo com o conjunto de frases, e a respectiva atribuição de valores a elas, são feitas dentro de um laço que perdura até chegar ao final do arquivo texto. A função de consulta é a mesma para ambos os casos e, dentro de seu escopo, segue um contador de comparações tal como nas funções de inserções, a fim de verificar a eficiência de ambas as árvores em relação a busca de um nodo. Esta função tem como objetivo buscar – uma a uma – cada palavra do texto, dentro da árvore criada no laço de inserção. Quando encontrada, o contador de sentimentos soma a atribuição dada a ela. No final da análise de cada frase, a polaridade contabilizada é escrita junto de sua respectiva frase, em uma mesma linha.

Ao final da mineração de opiniões, obtém-se o arquivo de saída, que consiste em um arquivo texto nomeado pelo próprio usuário. Este arquivo traz os valores obtidos da análise de sentimentos no início de cada frase, expressando a polaridade atribuída à mesma.

3.2. Resultados

Os gráficos abaixo expressam o desempenho do código após uma série de testes utilizando dois dicionários lexicográficos de sentimentos e dois arquivos textos – que contêm opiniões a respeito de filmes – como parâmetros de entrada. Os dois dicionários contêm exatamente as mesmas informações, todavia, somente um deles está organizado lexicograficamente.

Por sua vez, os arquivos texto a serem analisados possuem uma quantidade distinta de informações. Enquanto um deles possui exatamente mil frases, o outro possui cinco vezes mais. Por esta razão, o desempenho de consulta do código foi dividido em duas imagens.

Para uma melhor visualização, o eixo horizontal – que representa o número de comparações realizadas – foi incomumente representado em base-2 logarítmica. Uma visualização linear ocultaria as barras de menor dimensão dada a desproporção entre os dados.

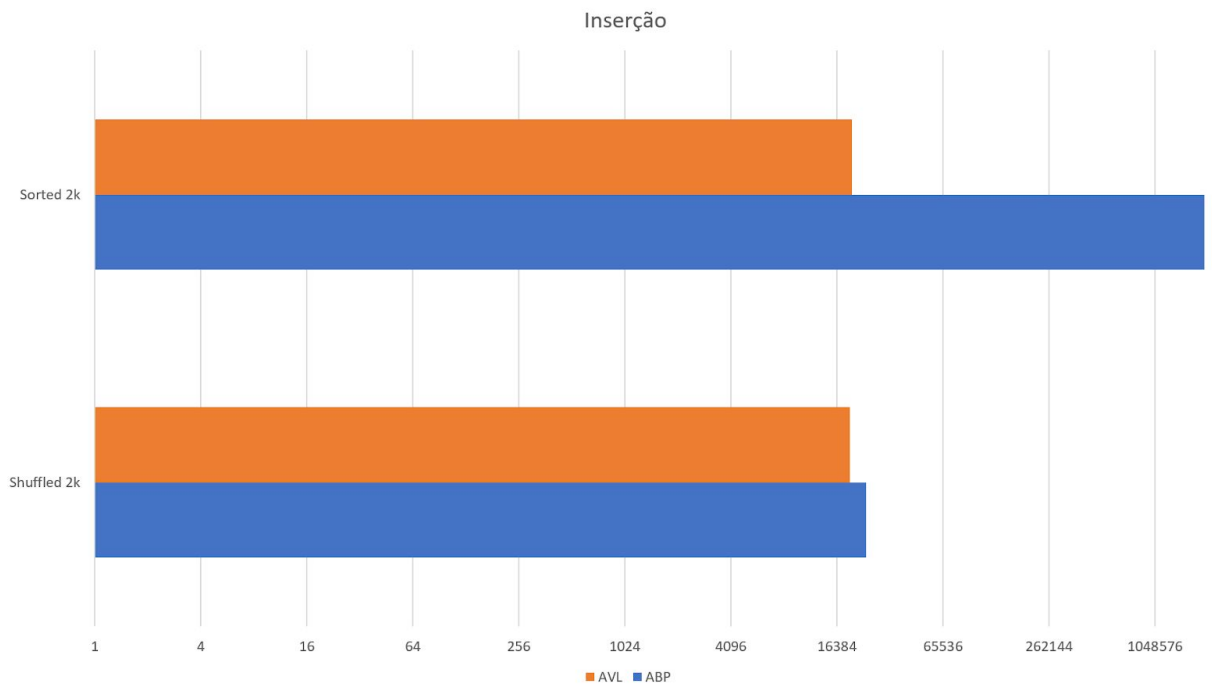

Gráfico 1: Representa o desempenho do código ao inserir os dicionários lexicográficos ordenado *(Sorted 2k)* e não-ordenado *(Shuffled 2k)*, em árvores ABPs simples e AVL.

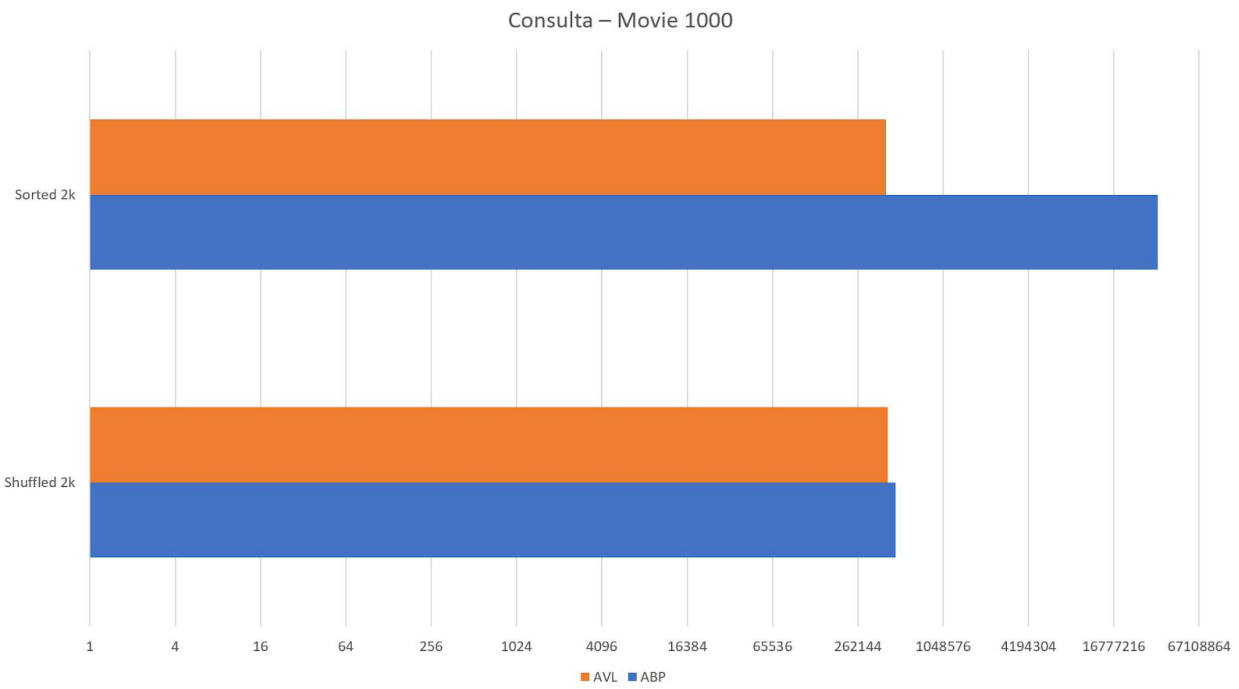

Gráfico 2: Representa o desempenho do código ao consultar os dicionários lexicográficos ordenado *(Sorted 2k)* e não-ordenado *(Shuffled 2k)*, já inseridos em árvores ABPs simples e AVL, para avaliar o arquivo texto de mil frases (*Movie 1000*).

Gráfico 3: Representa o desempenho do código ao consultar os dicionários lexicográficos ordenado *(Sorted 2k)* e não-ordenado *(Shuffled 2k)*, já inseridos em árvores ABPs simples e AVL, para avaliar o arquivo texto de cinco mil frases (*Movie 5000*).

4. CONCLUSÃO

A série de execuções demonstrou consistência, utilizando exatamente a mesma quantidade de comparação após executar mais de uma vez a mesma entrada. Também percebe-se que desempenho do algoritmo no processo de inserção depende exclusivamente do dicionário lexicográfico de sentimentos dado como entrada, enquanto para o processo de consulta, o desempenho – em termos de comparações – depende simultaneamente do dicionário de sentimentos e do arquivo texto a ser analisado.

Através da análise dos gráficos expressos no item anterior, nota-se que a ABP simples demonstrou ser menos eficiente que a AVL em todos os casos, entretanto demonstrou maior discrepância para inserção e consulta dos dados obtidos em um arquivo texto previamente ordenado. Conclui-se que para inserção dos nós já ordenados, a árvore ABP transforma-se em uma lista simplesmente encadeada, pois todo novo nó é alocado como filho direito do seu antecessor, reduzindo significamente sua eficiência, tanto para inserção quanto para consulta.

Portanto, a implementação do balanceamento AVL é de fato importante para melhorar a eficiência do algoritmo, em termos de número de comparações para ambas as duas operações supracitadas.

5. REFERÊNCIAS

[1] A. Rodrigues, L. Vieira, L. Malagoli e N. Timmermann. *Mineração de opiniões / Análise de sentimentos.*
Disponível em:http://www.inf.ufsc.br/~luis.alvares/INE5644/MineracaoOpiniao.pdf acessado em 04/12/18)

[2] Wikipedia. *Árvore AVL*.
Disponível em: https://pt.wikipedia.org/wiki/%C3%81rvore_AVL acessado em 04/12/18)