2º Trabalho de Projeto e Análise de Algoritmos 2024
Algoritmos de Árvores e Grafos

André Luiz Brun[1]

[1]Colegiado de Ciência da Computação
Campus de Cascavel - UNIOESTE

Resumo. *Este documento consiste na especificação formal do segundo trabalho da disciplina de Projeto e Análise de Algoritmos (Csc2152) para o ano letivo de 2024. Nesta avaliação serão aplicados conceitos acerca das estratégias de tratamento de estruturas hierárquicas, as árvores e das estruturas de grafos. Neste contexto serão exploradas estratégias sem balanceamento, com balanceamento a partir de fatores de coloração (RB) e com base no fator de balanceamento (AVL). Além disso, serão explorados algoritmos clássicos de manipulação de grafos, como métodos de menor distância, buscas, maior fluxo e estratégias para obtenção da árvore geradora mínima. Neste documento são apresentadas as atividades a serem desenvolvidas e como cada processo deverá ser realizado. Além disso, o documento contém as informações sobre a formação das equipes, o objeto de trabalho de cada uma e as datas de entrega e apresentação dos relatórios.*

O objetivo do segundo trabalho da disciplina consiste em comparar o comportamento, em termos práticos, de diferentes estratégias de implementação para tratar dados de forma hierárquica e estruturas representadas através de grafos. Busca-se determinar o **Custo Empírico** de cada implementação identificando-se o **tempo gasto** e **número de comparações** entre abordagens concorrentes. Neste experimento busca-se evidenciar quando cada estratégia é mais adequada frente à(s) sua(s) concorrente(s) através de experimentos práticos.

1. Gabriel Alves Mazzuco
Marco Antonio Damo
Maria Eduarda Crema Carlos

1.1. Objetivo

Aplicar e corroborar conceitos adquiridos sobre algoritmos de manipulação de dados de forma hierárquica. Para tanto, deverão ser implementadas duas soluções para árvores binárias de busca.

Na primeira solução deverá ser implementada uma **Árvore Binária de Busca (ABB)**, sem balanceamento.

Em seguida, afim de comparação, deverá ser implementada a estratégia de **Rubro Negra (RB)**.

1.2. Critérios de avaliação

Para realizar a comparação dos métodos a análise deverá ser feita em duas etapas:

1. Na primeira, construção, deverá ser estimada a quantidade de comparações entre chaves para descobrir a posição em que o elemento deve ser inserido e o tempo gasto para construir a estrutura inteira;
2. Na segunda, consulta, deverá ser contabilizado o número de comparações para se buscar os elementos na estrutura construída na etapa anterior;
3. Além da contagem de comparações deverão ser contabilizados os tempos gastos para se construir cada uma das árvores, bem como os temos necessários para se consultar todos os elementos presentes no arquivo de entrada.

1.3. Como

- As linguagens que podem ser utilizadas no desenvolvimento do trabalho são: C, C++, Java e Python;
- A mesma linguagem deve ser adotada para ambos os métodos. Além disso, deve-se empregar as mesmas estratégias para a realização do trabalho;
- A forma com que os métodos serão implementados é determinada pelo grupo;
- A entrada dos dados deve ser feita com base nos arquivos texto disponíveis no link a seguir: Link Grupo 1;

2. Fabrício Cordeiro Marcos
Fernando Schumaker Fiedler
Marlon Pereira

2.1. Objetivo

Aplicar e corroborar conceitos adquiridos sobre algoritmos de manipulação de dados de forma hierárquica. Para tanto, deverão ser implementadas duas soluções para árvores binárias de busca.

Inicialmente deverá ser construída uma árvore de busca binária **AVL** que servirá de estrutura para as buscas a serem avaliadas, ou seja, os desempenhos observados estarão atrelados ao comportamento da árvore obtida.

Na primeira solução deverá ser implementada uma **Busca em Profundidade (DFS)**. Em seguida, afim de comparação, deverá ser implementada a estratégia de **Busca em Largura (BFS)**. A ideia é comparar as duas abordagens em termos cronológicos e em número de comparações entre chaves para se realizar operações de busca.

2.2. Critérios de avaliação

Para realizar a comparação dos métodos a análise deverá ser feita em três etapas:

1. Na primeira, construção, deverá ser construída uma **Árvore AVL** a qual servirá de base para as duas opções de busca;
2. Na segunda, consulta DFS, deverá ser contabilizado o número de comparações para se buscar os elementos na estrutura construída na etapa anterior. Além disso, deverá ser contabilizado o tempo cronológico gasto para buscar todos os elementos do arquivo de consulta.
3. Na terceira etapa o processo será similar à etapa 2, porém, neste momento será implementada e avaliada uma busca em largura;

2.3. Como

- As linguagens que podem ser utilizadas no desenvolvimento do trabalho são: C, C++, Java e Python;
- A mesma linguagem deve ser adotada para ambos os métodos. Além disso, deve-se empregar as mesmas estratégias para a realização do trabalho;
- A forma com que os métodos serão implementados é determinada pelo grupo;
- Considerem como vértice inicial da busca sempre aquele de menor índice;
- A entrada dos dados deve ser feita com base nos arquivos texto disponíveis no link a seguir: Link Grupo 2;

3. Eduardo Pimentel dos Santos
Fabio Novack da Silva
Gustavo Magalhães Faino

3.1. Objetivo

Aplicar e corroborar conceitos adquiridos sobre algoritmos de manipulação de dados de forma hierárquica. Para tanto, deverão ser implementadas duas soluções para árvores binárias de busca.

Na primeira solução deverá ser implementada uma **Árvore Binária de Busca (ABB)**, sem balanceamento.

Em seguida, afim de comparação, deverá ser implementada a estratégia de **Árvore AVL**.

3.2. Critérios de avaliação

Para realizar a comparação dos métodos a análise deverá ser feita em duas etapas:

1. Na primeira, construção, deverá ser estimada a quantidade de comparações entre chaves para descobrir a posição em que o elemento deve ser inserido e o tempo gasto para construir a estrutura inteira;
2. Na segunda, consulta, deverá ser contabilizado o número de comparações para se buscar os elementos na estrutura construída na etapa anterior;
3. Além da contagem de comparações deverão ser contabilizados os tempos gastos para se construir cada uma das árvores, bem como os temos necessários para se consultar todos os elementos presentes no arquivo de entrada.

3.3. Como

- As linguagens que podem ser utilizadas no desenvolvimento do trabalho são: C, C++, Java e Python;
- A mesma linguagem deve ser adotada para ambos os métodos. Além disso, deve-se empregar as mesmas estratégias para a realização do trabalho;
- A forma com que os métodos serão implementados é determinada pelo grupo;
- A entrada dos dados deve ser feita com base nos arquivos texto disponíveis no link a seguir: Link Grupo 3;

4. David Antonio Brocardo
Gabriel Santos da Silva

4.1. Objetivo

Aplicar e corroborar conceitos adquiridos sobre algoritmos de manipulação de dados de forma hierárquica. Para tanto, deverão ser implementadas duas soluções para árvores binárias de busca.

Inicialmente deverá ser construída uma árvore de busca binária **Rubro Negra (RB)** que servirá de estrutura para as buscas a serem avaliadas, ou seja, os desempenhos observados estarão atrelados ao comportamento da árvore obtida.

Na primeira solução deverá ser implementada uma **Busca em Profundidade (DFS)**. Em seguida, afim de comparação, deverá ser implementada a estratégia de **Busca em Largura (BFS)**. A ideia é comparar as duas abordagens em termos cronológicos e em número de comparações entre chaves para se realizar operações de busca.

4.2. Critérios de avaliação

Para realizar a comparação dos métodos a análise deverá ser feita em três etapas:

1. Na primeira, construção, deverá ser construída uma **Rubro Negra** a qual servirá de base para as duas opções de busca;
2. Na segunda, consulta DFS, deverá ser contabilizado o número de comparações para se buscar os elementos na estrutura construída na etapa anterior. Além disso, deverá ser contabilizado o tempo cronológico gasto para buscar todos os elementos do arquivo de consulta.
3. Na terceira etapa o processo será similar à etapa 2, porém, neste momento será implementada e avaliada uma busca em largura;

4.3. Como

- As linguagens que podem ser utilizadas no desenvolvimento do trabalho são: C, C++, Java e Python;
- A mesma linguagem deve ser adotada para ambos os métodos. Além disso, deve-se empregar as mesmas estratégias para a realização do trabalho;
- A forma com que os métodos serão implementados é determinada pelo grupo;
- Considerem como vértice inicial da busca sempre aquele de menor índice;
- A entrada dos dados deve ser feita com base nos arquivos texto disponíveis no link a seguir: Link Grupo 4;

5. Gabriel Tadioto Oliveira
Leonardo B. Balan de Oliveira

5.1. Objetivo

Aplicar e corroborar conceitos adquiridos sobre algoritmos de manipulação de dados de forma hierárquica. Para tanto, deverão ser implementadas duas soluções para árvores binárias de busca.

Na primeira solução deverá ser implementada uma **Árvore Binária de Busca (ABB)** utilizando uma abordagem recursiva.

Em seguida, afim de comparação, deverá ser implementado o mesmo tipo de árvore adotando, no entanto, uma estratégia iterativa.

5.2. Critérios de avaliação

Para realizar a comparação dos métodos a análise deverá ser feita em duas etapas:

1. Na primeira, construção, deverá ser estimada a quantidade de comparações entre chaves para descobrir a posição em que o elemento deve ser inserido e o tempo gasto para construir a estrutura inteira;
2. Na segunda, consulta, deverá ser contabilizado o número de comparações para se buscar os elementos na estrutura construída na etapa anterior;
3. Além da contagem de comparações deverão ser contabilizados os tempos gastos para se construir cada uma das árvores, bem como os temos necessários para se consultar todos os elementos presentes no arquivo de entrada.

5.3. Como

- As linguagens que podem ser utilizadas no desenvolvimento do trabalho são: C, C++, Java e Python;
- A mesma linguagem deve ser adotada para ambos os métodos. Além disso, deve-se empregar as mesmas estratégias para a realização do trabalho;
- A forma com que os métodos serão implementados é determinada pelo grupo;
- A entrada dos dados deve ser feita com base nos arquivos texto disponíveis no link a seguir: Link Grupo 5;

6. Augusto Barella Dal Pra
Gabriel Rodrigues dos Santos
Gustavo Portela Rautenberg

6.1. Objetivo

Aplicar e corroborar conceitos adquiridos sobre algoritmos de manipulação de dados de forma hierárquica. Para tanto, deverão ser implementadas duas soluções para árvores binárias de busca.

Na primeira solução deverá ser implementada uma **Árvore AVL**. Em seguida, afim de comparação, deverá ser implementada a estratégia de **Árvore Rubro Negra (RB)**.

6.2. Critérios de avaliação

Para realizar a comparação dos métodos a análise deverá ser feita em duas etapas:

1. Na primeira, **construção**, deverá ser estimada a quantidade de comparações entre chaves para descobrir a posição em que o elemento deve ser inserido e o tempo gasto para construir a estrutura inteira;
2. Na segunda, **consulta**, deverá ser contabilizado o número de comparações e o tempo gasto para se buscar os elementos na estrutura construída na etapa anterior;
3. Além da contagem de comparações deverão ser contabilizados os tempos gastos para se construir cada uma das árvores, bem como os temos necessários para se consultar todos os elementos presentes no arquivo de entrada.

6.3. Como

- As linguagens que podem ser utilizadas no desenvolvimento do trabalho são: C, C++, Java e Python;
- A mesma linguagem deve ser adotada para ambos os métodos. Além disso, deve-se empregar as mesmas estratégias para a realização do trabalho;
- A forma com que os métodos serão implementados é determinada pelo grupo;
- A entrada dos dados deve ser feita com base nos arquivos texto disponíveis no link a seguir: Link Grupo 6;

7. Bruno Stafuzza Maion
Lucca Abbado Neres
Rafael Roberto Hoffmann

7.1. Objetivo

Aplicar e corroborar conceitos adquiridos sobre algoritmos de roteamento cujo objetivo é encontrar o menor caminho a partir de uma origem até todos os demais vértices de um grafo. Para tanto, deverão ser implementadas duas estratégias para obtenção dos menores caminhos.

Na primeira solução deverá ser implementado o algoritmo de **Dijkstra**. Em seguida, afim de comparação, deverá ser implementada o método de **Bellman-Ford**.

7.2. Critérios de avaliação

Para realizar a comparação dos métodos a análise deverá ser feita sob duas óticas. Em um primeiro momento serão comparados o tempo (cronológico) gasto para que os algoritmos sejam executados sobre todos os vértices dos grafos. Em seguida deverão ser comparadas as distâncias obtidas pelos métodos implementados. Espera-se que ambos cheguem ao mesmo custo.

7.3. Como

- As linguagens que podem ser utilizadas no desenvolvimento do trabalho são: C, C++, Java e Python;
- A mesma linguagem deve ser adotada para ambos os métodos. Além disso, deve-se empregar as mesmas estratégias para a realização do trabalho;
- Os dois algoritmos devem ser executados na mesma máquina;
- A forma com que os métodos serão implementados é determinada pelo grupo;
- A entrada dos dados deve ser feita com base nos arquivos texto disponíveis no link a seguir: Link Grupo 7;

8. João Vitor Biederman
Luiz Felipe Fonseca Rosa
Pedro Henrique Ferreira Zoz

8.1. Objetivo

Aplicar e corroborar conceitos adquiridos sobre algoritmos de roteamento cujo objetivo é encontrar o menor caminho entre todos os pares de vértices de um grafo. Para tanto, deverão ser implementadas duas estratégias para obtenção dos menores caminhos.

Na primeira solução deverá ser implementado o algoritmo de **Floyd-Warshall**. Em seguida, afim de comparação, deverá ser implementada o método de **Johnson**.

8.2. Critérios de avaliação

Para realizar a comparação dos métodos a análise deverá ser feita sob duas óticas. Em um primeiro momento serão comparados o tempo (cronológico) gasto para que os algoritmos sejam executados sobre todos os vértices dos grafos. Em seguida deverão ser comparadas as distâncias obtidas pelos métodos implementados. Espera-se que ambos cheguem ao mesmo custo.

8.3. Como

- As linguagens que podem ser utilizadas no desenvolvimento do trabalho são: C, C++, Java e Python;
- A mesma linguagem deve ser adotada para ambos os métodos. Além disso, deve-se empregar as mesmas estratégias para a realização do trabalho;
- Os dois algoritmos devem ser executados na mesma máquina;
- A forma com que os métodos serão implementados é determinada pelo grupo;
- A entrada dos dados deve ser feita com base nos arquivos texto disponíveis no link a seguir: Link Grupo 8;

9. Gustavo Macedo
Heloisa Aparecida Alves
Luiz Fernando Becher de Araujo

9.1. Objetivo

Aplicar e corroborar conceitos adquiridos sobre algoritmos de roteamento cujo objetivo é encontrar o menor caminho entre todos os pares de vértices de um grafo. Para tanto, deverão ser implementadas duas estratégias para obtenção dos menores caminhos.

Na primeira solução deverá ser implementado o algoritmo de **Floyd-Warshall**. Em seguida, afim de comparação, deverá ser implementado o método de **Dijkstra** que será executado V vezes. Em cada execução do algoritmo um novo vértice é usado como ponto de partida. Dessa forma, será possível encontrar o menor caminho entre todos os pares de vértices.

9.2. Critérios de avaliação

Para realizar a comparação dos métodos a análise deverá ser feita sob duas óticas. Em um primeiro momento serão comparados o tempo (cronológico) gasto para que os algoritmos sejam executados sobre todos os vértices dos grafos. Em seguida deverão ser comparadas as distâncias obtidas pelos métodos implementados. Espera-se que ambos cheguem ao mesmo custo.

9.3. Como

- As linguagens que podem ser utilizadas no desenvolvimento do trabalho são: C, C++, Java e Python;
- A mesma linguagem deve ser adotada para ambos os métodos. Além disso, deve-se empregar as mesmas estratégias para a realização do trabalho;
- Os dois algoritmos devem ser executados na mesma máquina;
- A forma com que os métodos serão implementados é determinada pelo grupo;
- A entrada dos dados deve ser feita com base nos arquivos texto disponíveis no link a seguir: Link Grupo 9;

10. Eduarda Elger
Ellen Carine Bonafin Marques
Lucas David Tomalack de Souza

10.1. Objetivo

Aplicar e corroborar conceitos adquiridos sobre algoritmos cujo objetivo é encontrar o maior fluxo entre o vértice inicial (fonte) e o vértice final (escoadouro) de uma rede de fluxo. Para tanto, deverão ser implementadas duas estratégias para obtenção dos maiores fluxos.

Na primeira solução deverá ser implementado o algoritmo de fluxo máximo de **Ford-Fulkerson** para encontrar o valor da maior quantidade de fluxo que pode ser passada pela rede fornecida.

Para critérios de comparação, na segunda solução deverá ser implementado o algoritmo de fluxo máximo de **Push-Relabel** para encontrar o valor da maior quantidade de fluxo que pode ser passada pela rede fornecida.

10.2. Critérios de avaliação

- Tempo cronológico para a execução do algoritmo de Ford-Fulkerson;
- Tempo cronológico para a execução do algoritmo de Push-Relabel;
- Comparativo entre os fluxos máximos encontrados pelas duas estratégias;
- Número de caminhos com capacidade residual diferente de zero que foram processados pelo Ford-Fulkerson;
- Número médio de vezes que cada vértice foi processado no método de Push-Relabel.

10.3. Como

- O vértice inicial (fonte) para o método será sempre o primeiro vértice presente no arquivo de entrada;
- O vértice final (escoadouro) para o método será sempre o último vértice presente no arquivo de entrada;
- As linguagens que podem ser utilizadas no desenvolvimento do trabalho são: C, C++, Java e Python;
- A mesma linguagem deve ser adotada para ambos os métodos. Além disso, deve-se empregar as mesmas estratégias para a realização do trabalho;
- Os dois algoritmos devem ser executados na mesma máquina;
- A forma com que os métodos serão implementados é determinada pelo grupo;
- A entrada dos dados deve ser feita com base nos arquivos texto disponíveis no link a seguir: Link Grupo 10.

11. Caio Hideki Gomes Shimohiro
Gustavo Alberto Ohse Hanke
Vinícius Visconsini Diniz

11.1. Objetivo

Aplicar e corroborar conceitos adquiridos sobre algoritmos de roteamento cujo objetivo é encontrar o menor caminho a partir de uma origem até todos os demais vértices de um grafo. Para tanto, deverão ser implementadas duas estratégias para obtenção dos menores caminhos.

Na primeira solução deverá ser implementado o algoritmo de **Dijkstra** utilizando como estratégia de gestão dos vértices, um **heap de mínimo**, de forma que o vértice mais próximo esteja sempre na raiz da estrutura.

Na segunda solução deverá ser implementado o mesmo algoritmo de **Dijkstra**. Porém, nesta solução ao invés de empregar-se o heap de mínimo, deverá ser empregada uma **lista simples** contendo os vértices que ainda precisam ser visitados.

11.2. Critérios de avaliação

Para realizar a comparação dos métodos a análise deverá ser feita sob duas óticas. Em um primeiro momento serão comparados o tempo (cronológico) gasto para que os algoritmos sejam executados sobre todos os vértices dos grafos. Em seguida deverão ser comparadas as distâncias obtidas pelos métodos implementados. Espera-se que ambos cheguem ao mesmo custo.

11.3. Como

- As linguagens que podem ser utilizadas no desenvolvimento do trabalho são: C, C++, Java e Python;
- A mesma linguagem deve ser adotada para ambos os métodos. Além disso, deve-se empregar as mesmas estratégias para a realização do trabalho;
- Os dois algoritmos devem ser executados na mesma máquina;
- A forma com que os métodos serão implementados é determinada pelo grupo;
- A entrada dos dados deve ser feita com base nos arquivos texto disponíveis no link a seguir: Link Grupo 11;

12. Isadora Coelho Araujo
Leonardo Calsavara
Ronaldo Drecksler Farias Pachico

12.1. Objetivo

Aplicar e corroborar conceitos adquiridos sobre algoritmos utilizados na obtenção de um subgrafo conectado contendo todos os vértices com a menor soma de custos para as arestas (Árvore Geradora Mínima (AGM)). Para tanto, deverão ser implementadas duas estratégias para obtenção dos menores caminhos.

Na primeira solução deverá ser implementado o algoritmo de **Kruskal** utilizando como estratégia de gestão dos vértices para evitar formação de ciclos, o método de union-find.

Na segunda solução deverá ser implementado o mesmo algoritmo de **Prim**. Para esta solução deverá ser adotada a estratégia que armazena a menor distância até cada vértice (similar ao relaxamento adotado no Dijkstra), evitando o reprocessamento de arestas.

12.2. Critérios de avaliação

Para realizar a comparação dos métodos a análise deverá ser feita sob duas óticas. Em um primeiro momento serão comparados o tempo (cronológico) gasto para que os algoritmos sejam executados sobre todos os vértices dos grafos. Em seguida deverão ser comparadas as distâncias obtidas pelos métodos implementados. Espera-se que ambos cheguem ao mesmo custo.

12.3. Como

- As linguagens que podem ser utilizadas no desenvolvimento do trabalho são: C, C++, Java e Python;
- A mesma linguagem deve ser adotada para ambos os métodos. Além disso, deve-se empregar as mesmas estratégias para a realização do trabalho;
- Os dois algoritmos devem ser executados na mesma máquina;
- A forma com que os métodos serão implementados é determinada pelo grupo;
- A entrada dos dados deve ser feita com base nos arquivos texto disponíveis no link a seguir: Link Grupo 12;

13. Relatório

Deve ser elaborado um relatório técnico em formato pdf contendo:

- o Descrição de como foi realizado o processo empírico de determinação dos custos: cenário de realização dos experimentos e como foram tomadas as métricas exigidas.
  - Detalhar a configuração usada nos testes (Processador, SO, IDE, etc..).
- Gráficos evidenciando o comportamento dos métodos perante todos os cenários considerando o tamanho dos conjuntos de entrada.
- Análise do comportamento dos métodos durante a execução dos testes.
  - Esta análise deve ser feita com bastante critério e ser esclarecedora, apontando razões para os comportamentos observados.
  - **AS figuras e gráficos devem ser invocadas no texto e explicadas.**

O formato do relatório deve ser a formatação presente neste texto. As regras para tal podem ser obtidas no link download. No arquivo disponível pode-se utilizar a formatação em arquivo .doc ou em latex.

14. Código-fonte

Além do relatório citado, cada equipe deverá enviar os códigos fontes construídos para a execução dos experimentos. Ambos arquivos podem ser compactados e enviados como arquivo único.

Não é necessário o envio do projeto compilado, apenas do código fonte base.

Neste código devem constar os comandos utilizados para tomadas de tempo de execução e contagem das variáveis de interesse.

15. Para quando?

O trabalho deverá ser submetido no link disponibilizado na turma de disciplina dentro do ambiente Microsoft Teams até as **23:59 do dia XX/XX/2025.**

As apresentações serão realizadas nas aulas dos dias **02** e **XX/XX/2025**.

Cada grupo terá 15 minutos para apresentar o trabalho realizado, focando na descrição do problema, nos desempenhos obtidos e nos resultados das análises. **A qualidade da apresentação terá influência na nota do trabalho.**

Além da apresentação as equipes serão avaliadas pelo seu comportamento e presença durante as apresentações dos demais grupos.