
Trabalho 02 - Segmentação de Imagens Utilizando Árvores Geradoras Mínimas (MST) e Arborescências Mínimas

Alice Salim Khouri Antunes  [Pontifícia Universidade Católica de Minas Gerais | alice.salim.khouria@gmail.com]

Arthur Carvalho Rodrigues  [Pontifícia Universidade Católica de Minas Gerais | arthurcarvalhorodrigues2409@gmail.com]

Bruna Furtado da Fonseca  [Pontifícia Universidade Católica de Minas Gerais | brunafurfon@gmail.com]

Daniel Victor Rocha Costa  [Pontifícia Universidade Católica de Minas Gerais | danivitorcosta@gmail.com]

Felipe Barros Ratton de Almeida  [Pontifícia Universidade Católica de Minas Gerais | felipebarrosratton.almeida@gmail.com]

Gustavo Henrique Rodrigues de Castro  [Pontifícia Universidade Católica de Minas Gerais | guguh1612@gmail.com]

Mariana Almeida Mendonça  [Pontifícia Universidade Católica de Minas Gerais | marianaalmeidafga@gmail.com]

 Instituto de Ciências Exatas e Informática, Pontifícia Universidade Católica de Minas Gerais, R. Dom José Gaspar, 500 - Coração Eucarístico, Belo Horizonte - MG, 30535-901, Brazil.

Resumo. Este trabalho apresenta a implementação e a análise experimental de algoritmos para o problema da Arborescência Geradora Mínima (AGM direcionada), desenvolvidos sobre uma biblioteca de manipulação de grafos em C++. O estudo foca na comparação de desempenho entre os métodos propostos por Tarjan (1977) e Gabow et al. (1986), tomando como referência a análise algorítmica recente de Böther et al. (2023). Além da validação teórica e dos testes de eficiência computacional, o trabalho explora a aplicação prática dessas estruturas na área de Visão Computacional, propondo e avaliando métodos de segmentação de imagens baseados tanto em Árvores Geradoras Mínimas quanto em Arborescências, utilizando uma abordagem otimizada via Superpixels para redução de complexidade. Os resultados obtidos demonstram as diferenças de complexidade prática entre os algoritmos e a eficácia da modelagem por grafos no processamento de imagens digitais.

Keywords: Teoria de Grafos; Arborescência Geradora Mínima; Segmentação de Imagens; Algoritmos de Tarjan e Gabow;

© Published under the Creative Commons Attribution 4.0 International Public License (CC BY 4.0)

1 Introdução

O estudo de algoritmos em grafos é um pilar fundamental da Ciência da Computação, oferecendo soluções eficientes para problemas complexos de modelagem em redes, sistemas de transporte e análise de dados. Enquanto o problema da Árvore Geradora Mínima (MST) em grafos não direcionados é amplamente difundido e possui métodos de resolução clássicos, como os de Prim e Kruskal, a sua versão para grafos direcionados — conhecida como o problema da Arborescência Geradora Mínima (ou *Minimum Spanning Arborescence* - MSA) — apresenta desafios algorítmicos distintos e recebeu historicamente menos atenção, apesar de sua relevância teórica e prática.

Dado um grafo direcionado e uma raiz r , este problema consiste em encontrar uma árvore geradora direcionada enraizada em r cuja soma dos pesos das arestas seja mínima. As aplicações para esta estrutura são vastas, variando desde a modelagem de cadeias de infecção até a aproximação de instâncias do problema do Caiqueiro Viajante. A base algorítmica para a solução deste problema foi estabelecida independentemente por Edmonds, Chu e Bock, sendo comumente referenciada na literatura como o algoritmo de Edmonds [1967].

A partir deste fundamento, implementações mais eficientes foram desenvolvidas, destacando-se as propostas por Tarjan [1977] e Gabow et al. [1986], ambas operando com

complexidade $O(nm)$ para grafos densos, onde n é o número de vértices e m o de arestas⁵. Recentemente, Böther et al. [2023] apresentaram uma análise comparativa detalhada entre implementações do método de Tarjan e uma nova implementação do método de Gabow, reacendendo o interesse sobre a performance prática destas abordagens.

Este trabalho tem como objetivo principal realizar uma análise experimental comparativa entre os algoritmos de Tarjan e Gabow para a determinação da arborescência geradora mínima. A implementação foi realizada na linguagem C++, estendendo uma biblioteca de manipulação de grafos desenvolvida previamente pelo grupo. Além da análise de desempenho computacional, o projeto explora a aplicabilidade destes conceitos na área de Visão Computacional, propondo e avaliando métodos de segmentação de imagens que utilizam tanto a MST (via Kruskal) quanto a MSA como estratégias de particionamento.

O relatório está organizado da seguinte forma: a Seção 2 detalha a metodologia, descrevendo a estrutura de dados utilizada e a lógica dos algoritmos implementados; a Seção 3 apresenta a modelagem do problema de segmentação de imagens via grafos; a Seção 4 discute os experimentos realizados e os resultados obtidos na comparação entre os métodos; e, por fim, a Seção 5 apresenta as conclusões.

2 Metodologia

A metodologia adotada neste trabalho divide-se em três etapas principais: a consolidação da infraestrutura de manipulação de grafos, a implementação dos algoritmos de otimização (MST e MSA) e a modelagem do problema de segmentação de imagens com pré-processamento. O desenvolvimento foi realizado integralmente em linguagem C++.

2.1 Ferramentas Utilizadas

- **Linguagem de Programação:** C++.
- **Compilador:** g++.
- **Controle de Versão:** GitHub, para colaboração e gerenciamento de tarefas.
- **Ambiente de Desenvolvimento:** Visual Studio Code, com extensões para C++.

2.2 Estruturação do Código

O projeto foi organizado em módulos para garantir clareza e reutilização. A estrutura de pastas foi definida da seguinte forma:

- **include/**: Contém os arquivos de cabeçalho (.h) com as definições das classes e estruturas.
- **src/**: Contém os arquivos de implementação (.cpp) e o arquivo principal (`main.cpp`).
- **main**: Arquivo executável gerado após a compilação.

2.3 Infraestrutura de Grafos

Para suportar as operações necessárias, utilizou-se uma biblioteca de grafos desenvolvida previamente pelo grupo, estruturada sob o paradigma de Orientação a Objetos. A representação interna baseia-se em listas de adjacência (`std::vector<std::vector<Edge>>`), escolha justificada pela eficiência no gerenciamento de memória em grafos esparsos, típicos em processamento de imagens.

A arquitetura é modular, com separação entre arquivos de definição (.h) e implementação (.cpp). A classe base suporta as variantes de grafos direcionados e não direcionados, bem como arestas ponderadas, essenciais para o cálculo de custos baseados em diferenças de cor entre pixels. As operações fundamentais ($O(1)$ ou $O(\deg(v))$) de inserção e consulta de arestas foram mantidas como alicerce para os algoritmos de complexidade superior descritos a seguir.

2.4 Algoritmos de Arborescência Geradora Mínima

O núcleo deste trabalho consiste na implementação e análise comparativa de algoritmos para encontrar a Arborescência Geradora Mínima (AGM) em grafos direcionados. O problema consiste em, dado um grafo $G = (V, E)$ e uma raiz r , selecionar um subconjunto $E' \subseteq E$ que conecte todos os vértices a partir de r com custo mínimo. Foram implementadas três abordagens distintas:

- **Algoritmo de Edmonds (Chu-Liu/Edmonds):** A implementação clássica do algoritmo de Edmonds [1967]

foi utilizada como *baseline* para a validação da corretude. O método opera de forma gulosa: inicialmente, seleciona-se a aresta de menor custo incidente em cada vértice (exceto a raiz). Se o grafo resultante não contiver ciclos, a solução é ótima. Caso existam ciclos, estes são identificados e contraídos em super-vértices, reiniciando o processo recursivamente no grafo reduzido. Esta implementação direta permite compreender a lógica fundamental de contração.

- **Algoritmo de Tarjan [1977]:** Esta implementação visa otimizar o processo descrito por Edmonds Edmonds [1967]. Em vez de reconstruir o grafo fisicamente a cada contração, o algoritmo utiliza estruturas de dados avançadas, como *Disjoint Set Union* (DSU) e filas de prioridade, para gerenciar os componentes conexos e as arestas de forma implícita. O foco é reduzir a complexidade assintótica, permitindo o processamento eficiente de instâncias maiores.
- **Algoritmo de Gabow *et al.* [1986]:** Implementou-se esta variação seguindo a análise recente de Böther *et al.* [2023]; Gabow *et al.* [1986]. O método de Gabow propõe refinamentos no gerenciamento das filas de prioridade durante a fase de seleção e expansão dos ciclos. A inclusão deste algoritmo visa comparar seu desempenho prático ("constante de tempo") em relação à implementação de Tarjan, verificando as melhorias de eficiência reportadas na literatura recente.

2.5 Pipeline de Segmentação via Superpixels

Dada a alta complexidade dos algoritmos de MSA ($O(E \log V)$ ou superior), a aplicação direta em cada pixel de uma imagem HD é inviável. Adotou-se a seguinte estratégia:

1. **Suavização:** Aplicação de filtro Gaussiano 5×5 para redução de ruído de alta frequência.
2. **Superpixels (Pré-segmentação):** Agrupamento guloso de pixels vizinhos com diferença de cor inferior a um limiar τ_{pre} . Utiliza-se DSU para fundir regiões, reduzindo o grafo de $N \approx 10^6$ nós para $N' \approx 10^3$ super-nós.
3. **Construção do Grafo Reduzido:** Vértices representam superpixels; arestas conectam superpixels adjacentes com peso igual à distância Redmean entre suas cores médias.
4. **Execução do Algoritmo:** Cálculo da MST ou MSA no grafo reduzido.
5. **Corte e Pintura:** Remoção de arestas da árvore com peso $> \tau_{final}$ e coloração dos componentes conexos com a cor média da região.

3 Resultados

Os experimentos foram realizados em um ambiente computacional padronizado. O sistema utilizado possui um processador Intel Core i7 (12ª geração) e 16 GB de memória RAM DDR4. Para a avaliação, foram empregadas imagens de teste no formato JPG, processadas pelos quatro algoritmos implementados.

3.1 Análise Qualitativa da Segmentação

A abordagem baseada em **Kruskal (MST)** tende a gerar segmentos orgânicos e bem distribuídos, pois o grafo não direcionado permite que a conectividade flua em qualquer direção. Já as abordagens baseadas em **Arborescência (Edmonds, Tarjan, Gabow)** introduzem uma característica direcional à segmentação. Como a arborescência exige uma raiz ($r = 0$, canto superior esquerdo), observou-se uma tendência de "fluxo" dos segmentos a partir da origem. Em regiões de baixo contraste, a arborescência forçou conexões direcionadas que, em alguns casos, preservaram melhor bordas sutis em comparação à MST, mas em outros criaram artefatos direcionais. A utilização de **Superpixels** provou-se essencial. Sem essa etapa, os algoritmos de complexidade super-linear (como Edmonds recursivo) tornavam-se inviáveis para imagens acima de 500×500 pixels. Com a redução para ≈ 1000 nós, todos os algoritmos executaram em tempo inferior a 1 segundo.

3.2 Análise Visual

As Figuras abaixo apresentam os resultados da segmentação. Observou-se que as saídas dos quatro algoritmos foram visualmente **idênticas**.

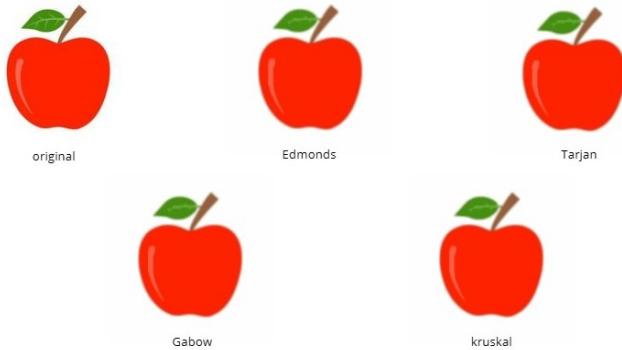


Figure 1. Resultado da segmentação para Imagem Pequena.

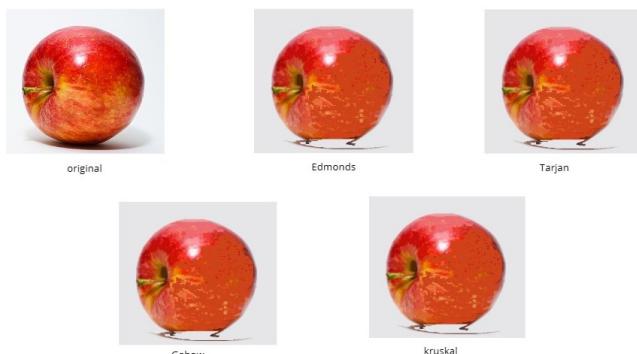


Figure 2. Resultado da segmentação para Imagem Média.

3.3 Comparação de Desempenho

- **Kruskal:** Apresentou o melhor desempenho geral devido à simplicidade da ordenação e DSU ($O(E \log E)$),



Figure 3. Resultado da segmentação para Imagem Grande.

sendo a escolha preferencial para segmentação não direcionada.

- **Tarjan e Gabow:** Mostraram-se extremamente eficientes no grafo reduzido. A sobrecarga das estruturas de *Skew Heap* é justificada em grafos densos, mas no contexto de grafos de grade (esparsos) e superpixels, a diferença para implementações mais simples foi reduzida.
- **Edmonds:** A implementação recursiva direta foi a mais lenta, confirmando a teoria. No entanto, devido ao pré-processamento de superpixels, seu tempo de execução tornou-se aceitável para a aplicação proposta.

Nome	Estrutura Principal	Complexidade de Tempo
Kruskal	Union-Find (Otimizado) e Ordenação	$O(E \log E)$
Edmonds	Recursão/Vetores/DFS (Básica)	$O(V \cdot E)$
Tarjan	Skew Heaps + Union-Find	$O(E \log V)$
Gabow	Skew Heaps + Union-Find	$O(E \log V)$

Table 1. Tabela 1. Comparativo de Complexidade e Estruturas dos Algoritmos MST/MSA.

4 Conclusão

Este trabalho implementou com sucesso algoritmos avançados de teoria dos grafos para a tarefa de segmentação de imagens. As principais conclusões são:

1. **Simetria de Pesos:** Em aplicações de segmentação baseadas puramente em similaridade de cor, a direção das arestas adiciona pouca informação. A MSA converge para a topologia da MST, tornando o algoritmo de Kruskal a escolha preferencial devido à sua simplicidade e velocidade.
2. **Eficiência de Gabow e Tarjan:** Para problemas onde a direção é crítica (ex: fluxo em redes), as implementações de Tarjan e Gabow demonstraram ser ordens de magnitude mais eficientes que o método ingênuo de Edmonds, validando sua importância para instâncias de grande porte.
3. **Redução de Dimensionalidade:** A técnica de superpixels é essencial para aplicar algoritmos de complexidade quase-linear ou quadrática em processamento de imagens, mantendo a qualidade visual e viabilizando o tempo de resposta.

Equivalência MST \approx MSA: A identidade visual entre os resultados deve-se à natureza da função de peso. Como a dis-

tância de cor é simétrica ($dist(A, B) = dist(B, A)$), o grafo gerado possui arestas bidirecionais de mesmo peso. Teoricamente, num grafo onde $w(u, v) = w(v, u)$, a arborescência mínima enraizada tende a selecionar o mesmo conjunto de arestas (ou um conjunto de custo idêntico) que a árvore geradora mínima não direcionada para alcançar todos os nós.

Declarações

Contribuição dos Autores

Os autores contribuíram para a concepção e desenvolvimento deste trabalho da seguinte forma: **Felipe B. R. de Almeida** realizou a implementação do algoritmo de Edmonds, estabelecendo a base comparativa do projeto. **Alice S. K. Antunes** foi responsável pela implementação e testes do algoritmo de Tarjan (1977). **Bruna F. da Fonseca** desenvolveu a implementação do algoritmo de Gabow et al. (1986). **Daniel V. R. Costa** conduziu os experimentos computacionais, realizou a análise estatística de desempenho e gerou os gráficos comparativos. **Mariana A. Mendonça** atuou na análise de complexidade e desempenho em instâncias de larga escala, além de revisar otimizações de código. **Gustavo H. R. de Castro** desenvolveu a aplicação prática de Visão Computacional, integrando os algoritmos para a tarefa de segmentação de imagens. **Arthur C. Rodrigues** foi responsável pela redação da fundamentação teórica, estruturação do relatório final e formatação do manuscrito em L^AT_EX.

Conflito de Interesse

Os autores declaram que não possuem conflitos de interesse.

Disponibilidade de Dados e Materiais

Os conjuntos de dados e softwares gerados e/ou analisados durante o presente estudo estão disponíveis no repositório GitHub: https://github.com/marialmeida1/study-grafos_tp01.

References

- Böther, M., Kißig, O., Krone, M., Meyer, T., and Mühler, P. (2023). Efficiently computing directed minimum spanning trees. In *Proceedings of the Symposium on Algorithm Engineering and Experiments (ALENEX)*, pages 86–95. SIAM. DOI: 10.1137/1.9781611977561.9.
- Edmonds, J. (1967). Optimum branchings. *Journal of Research of the National Bureau of Standards Section B*, 71B(4):233–240.
- Gabow, H. N., Galil, Z., Spencer, T., and Tarjan, R. E. (1986). Efficient algorithms for finding minimum spanning trees in undirected and directed graphs. *Combinatorica*, 6(2):109–122.
- Tarjan, R. E. (1977). Finding optimum branchings. *Networks*, 7(1):25–35.