# Documentação dos Algoritmos em Grafos Implementados

Pedro Carbonaro
Pontifícia Universidade Católica de Minas Gerais
Belo Horizonte, Brasil
pacgoncalves@sga.pucminas.br

Gabriel Todt
Pontifícia Universidade Católica de Minas Gerais
Belo Horizonte, Brasil
gabriel.ferreira.1389415@sga.pucminas.br

## ABSTRACT

Este documento apresenta uma análise detalhada de três algoritmos fundamentais em grafos, implementados como parte de um trabalho prático. Os algoritmos incluem (1) um método que verifica a existência de dois caminhos internamente disjuntos (ou um ciclo) entre cada par de vértices do bloco; (2) um método que identifica articulações testando a conectividade após a remoção de cada vértice; e (3) o método proposto por Tarjan (1972) para encontrar articulações em um grafo não direcionado. Cada algoritmo é detalhado em seções subsequentes, abrangendo desde a teoria por trás de seu funcionamento até sua implementação prática.

## KEYWORDS

Grafos, Algoritmos, Articulações, Conectividade, Componentes Conexos, Tarjan

**ACM Reference Format:**
Pedro Carbonaro and Gabriel Todt. 2024. Documentação dos Algoritmos em Grafos Implementados. In *Proceedings of Conference Name.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

## 1 INTRODUÇÃO

Grafos são estruturas de dados fundamentais em ciência da computação, amplamente utilizados para modelar relacionamentos entre objetos. Neste trabalho, exploramos três algoritmos essenciais em grafos que têm aplicações em uma variedade de problemas computacionais. Começamos com uma visão geral dos algoritmos e, em seguida, mergulhamos em suas implementações detalhadas e análises de desempenho.

## 2 VISÃO GERAL DOS ALGORITMOS

Os três algoritmos abordados nesta documentação são:

(1) Verificação de dois caminhos internamente disjuntos (ou ciclo) entre cada par de vértices do bloco.
(2) Identificação de articulações verificando a existência de componentes após a remoção de cada vértice.
(3) Método proposto por Tarjan (1972) para encontrar componentes fortemente conexos em um grafo não direcionado.


*Conference Name, Date, Location*

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

Esses algoritmos são fundamentais para resolver uma variedade de problemas em grafos, incluindo roteamento de rede, análise de redes sociais e muito mais.

## 3 ALGORITMOS INDIVIDUAIS

### 3.1 Verificação de dois caminhos disjuntos (ou ciclo) entre cada par de vértices do bloco

O algoritmo desenvolvido testa todos os pares possíveis de vértices, procurando caminhos internamente disjuntos entre eles. Ou seja, ele encontra um caminho do vértice fonte até o vértice destino e, em seguida, procura por outro caminho que não utilize nenhum vértice já utilizado pela primeira busca, exceto a fonte e o destino. A implementação desenvolvida faz uso de duas buscas em profundidade, onde os vértices são marcados como já utilizados e, dessa forma, não serão visitados novamente na segunda busca.

### 3.2 Identificação de articulações

Articulações são vértices cuja remoção divide um grafo em componentes. Em um problema real, como uma rede de computadores, podemos pensar em uma articulação como um ponto crítico, já que se este ponto deixar de funcionar, a rede é dividida e alguns dispositivos não conseguem se comunicar com outros.

Para encontrar articulações, podemos remover um vértice e fazer uma busca. Se todos os vértices restantes forem encontrados, então o vértice removido não é uma articulação; no entanto, se algum vértice restante não for encontrado na busca, então o vértice removido é uma articulação. Esse teste pode ser feito para todos os vértices, dessa forma serão encontradas todas as articulações.

### 3.3 Método proposto por Tarjan (1972)

O método de Tarjan é amplamente utilizado para encontrar componentes fortemente conexos em um grafo direcionado. Ele é baseado em busca em profundidade e utiliza uma pilha para rastrear os vértices visitados. A complexidade deste algoritmo é linear no número de vértices e arestas do grafo.

## 4 COMPARAÇÃO DOS ALGORITMOS

Nesta sessão iremos comparar o desempenho de cada algoritmo em relação ao tempo de relógio para a execução utilizando grafos de diversos tamanhos, ou seja, número de vértices e arestas. Os testes foram realizados em uma máquina com as seguintes configurações: Processador Intel Core i5 9400f, 16 GB memória RAM 2660 MHz, rodando em um sistema Linux Pop OS 22.04 LTS.

## 5 CONCLUSÃO

Neste trabalho, realizamos uma análise detalhada de três algoritmos em grafos: verificação de dois caminhos disjuntos, identificação de

| | 100 | 1k | 10k | 100k |
|---|---|---|---|---|
| Disjointed paths | < 1 segundo | 27 segundo | > 2 horas | > 4 horas |
| Remove articulation | < 1 segundos | 7 minutos | > 2 horas | > 6 horas |
| Tarjan | < 1 Segundos | < 1 Segundos | < 2 Segundos | 7 Segundos |

**Figure 1: Diferença de desempenho de cada um dos algoritmos.**

articulações e o método proposto por Tarjan para encontrar articulações. Cada algoritmo foi estudado em profundidade, desde sua teoria até sua implementação prática, com análises de desempenho comparativas.

Os resultados dos testes de desempenho revelaram insights interessantes sobre o comportamento de cada algoritmo em diferentes tipos de grafos. Observamos que, em geral, o método proposto por Tarjan demonstrou ser altamente eficiente, com uma complexidade linear em relação ao número de vértices e arestas.

## 6 CÓDIGO DESENVOLVIDO

O código-fonte dos algoritmos implementados está disponível no repositório GitHub: https://github.com/PedroCarbonaroG/Graphs.

## REFERENCES

[1] Princeton Graphs Algorithms, *Algoritmos auxiliares para estudo de grafos*, Disponível em: https://algs4.cs.princeton.edu/40graphs/,

[2] Tarjan Graphs Algorithms, *Algoritmo criado por Tarjan para componentes conexos*, Disponível em: https://epubs.siam.org/doi/abs/10.1137/0201010,