

Relatório de AED

Departamento de Eletrónica, Telecomunicações e Informática

Martim Peralta Gomes, nº 119488

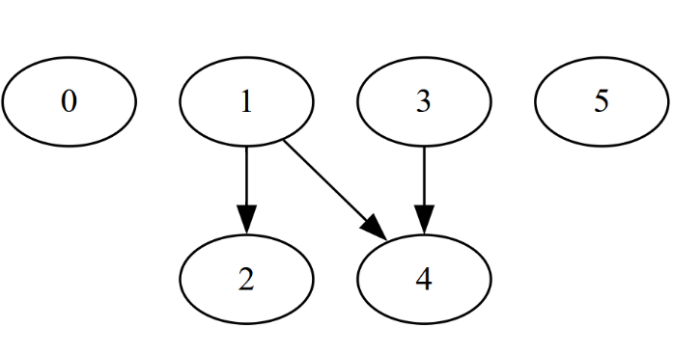
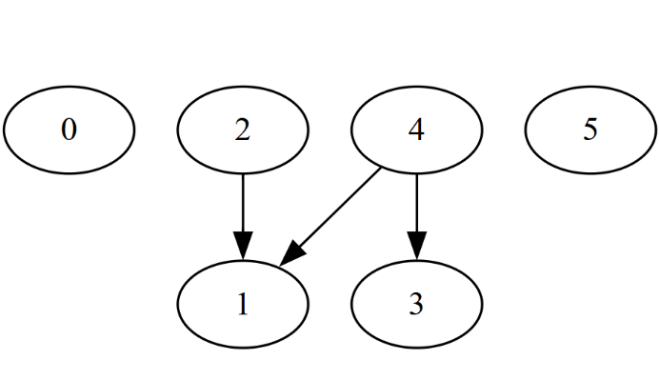
Tiago Queirós Rocha, nº 120515

**Introdução**

O presente relatório tem como objetivo caracterizar a complexidade computacional de dois algoritmos fundamentais em grafos orientados: o algoritmo de Bellman-Ford e o algoritmo de construção do fecho transitivo. Estes algoritmos foram implementados e avaliados no contexto do Tipo Abstrato de Dados (TAD) GRAPH.

O algoritmo de Bellman-Ford foi utilizado para determinar a árvore de caminhos mais curtos a partir de um vértice inicial até todos os vértices alcançáveis, enquanto o algoritmo de fecho transitivo foi responsável por gerar um grafo no qual todos os pares de vértices conectados por caminhos direcionados no grafo original são explicitamente representados.

Além disso, o relatório apresenta as métricas adotadas para medir a complexidade algorítmica, como o número de iterações e o tempo de execução, ilustradas por meio de tabelas e gráficos baseados nos resultados dos testes efetuados em grafos de diferentes tamanhos e estruturas.



Grafo Transposto

TAD Graph

**Bellman-Ford**

Path Trees (Árvores de Caminhos) por Vértice Raiz:

* Vértice 0

Não existem arestas que partem do vértice 0.

* Vértice 1

Arestas que partem do vértice

1 → 2

1 → 4

* Vértice 2

Não existem arestas que partem do vértice 2.

* Vértice 3

Aresta que parte do vértice 3:

3 → 4

* Vértice 4

Não existem arestas que partem do vértice 4.

* Vértice 5

Não existem arestas que partem do vértice 5.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Graph Root | Vertices | Arestas | Comparações | Iterações |
| 0 | 6 | 0 | 0 | 5 |
| 1 | 6 | 2 | 10 | 5 |
| 2 | 6 | 0 | 0 | 5 |
| 3 | 6 | 1 | 5 | 5 |
| 4 | 6 | 0 | 0 | 5 |
| 5 | 6 | 0 | 0 | 5 |

De forma a analisar a complexidade do algoritmo, fizemos uma tabela com as comparações e iterações dos vértices e arestas:

**Número de Iterações**

* Para todos os graph roots, o número de iterações é **5**, que corresponde a V − 1 (número de vértices menos 1).
* Isso reflete o funcionamento do Bellman-Ford, que faz V − 1 passos para calcular as menores distâncias.

**Número de Comparações**

As comparações dependem diretamente do número de arestas processadas:

* Para o graph root 1 (2 arestas): 10 comparações (5 × 2).
* Para o graph root 3 (1 aresta): 5 comparações (5 × 1)

Para outros graph roots sem arestas conectadas, não há comparações realizadas (0).

O algoritmo de Bellman-Ford apresenta uma complexidade de tempo de O(V⋅E)O(V⋅E), onde VV representa o número de vértices e EE, o número de arestas no grafo. No pior caso, o algoritmo percorre todas as arestas para cada vértice, o que resulta nessa complexidade temporal.

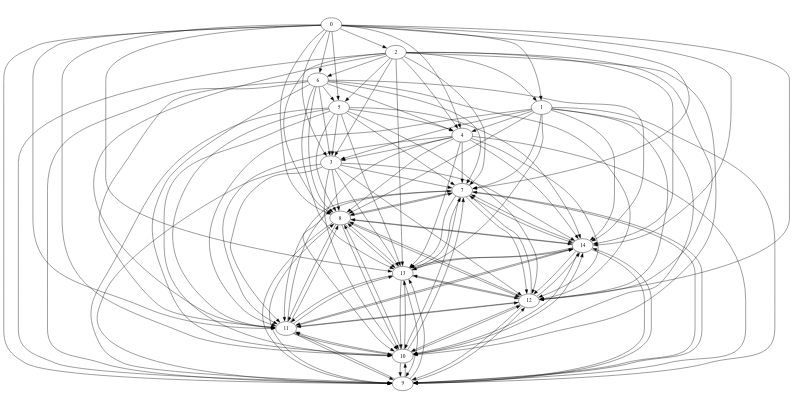
Já a complexidade espacial do algoritmo é O(V)O(V), uma vez que é necessário armazenar as distâncias entre o vértice de origem e todos os outros vértices do grafo. Esse armazenamento é o principal fator que determina o uso de memória pelo algoritmo.

No melhor caso, o algoritmo de Bellman-Ford possui uma complexidade de tempo de O(E). Nesse cenário, as distâncias iniciais já representam os caminhos mais curtos, e o algoritmo faz apenas uma passagem eficiente por todas as arestas para verificar sua validade. Essa eficiência é particularmente vantajosa em grafos onde o número de arestas é reduzido.

No caso médio, a complexidade de tempo do algoritmo de Bellman-Ford é O(V⋅E). Esta complexidade aplica-se a diversas estruturas e densidades de grafos, já que o desempenho do algoritmo é influenciado principalmente pelo número de vértices e arestas. Em situações práticas, o comportamento do caso médio geralmente assemelha-se ao pior caso, especialmente em grafos densos com um grande número de arestas. (editado)

No pior caso, o algoritmo de Bellman-Ford apresenta uma complexidade de tempo de O(V⋅E). Isso acontece quando o algoritmo realiza ∣V∣−1∣ iterações completas sobre todas as arestas e vértices, além de uma iteração adicional para verificar a existência de ciclos de peso negativo.

**Transitive Closure**



No cálculo do fecho transitivo de um grafo, as comparações ocorrem em dois momentos principais: durante a execução do algoritmo de Bellman-Ford, para determinar se ocorreram relaxamentos, e na construção do grafo transitivo, para verificar a existência de caminhos entre os vértices. O número de iterações é diretamente proporcional ao número de vértices (V).

A complexidade do algoritmo de Bellman-Ford é O(V⋅E), onde E representa o número de arestas. No cálculo do fecho transitivo, esse algoritmo é executado V vezes, resultando numa complexidade global de O(V⋅(V⋅E)) = O(V²⋅E).

No melhor caso, o número de arestas é reduzido, o que minimiza as operações realizadas. Embora o Bellman-Ford precise ser executado V vezes, o tempo de execução é reduzido devido ao baixo número de arestas por vértice. Nesse caso, a complexidade pode ser expressa como O(V² ⋅ X), onde X é o número médio de arestas por vértice.

No pior caso, o grafo possui o número máximo de arestas, ou seja, é denso, com E≈V². Como cada vértice está conectado a praticamente todos os outros, a complexidade do algoritmo torna-se O(V²⋅V²) = O(V⁴).