Identificação de Caminhos Disjuntos em Arestas Utilizando o Algoritmo de Ford-Fulkerson

Luis Phillip

Outubro de 2024

1 Metodologia

Foi utilizado o algoritmo de Ford-Fulkerson, configurando todas as arestas do grafo com capacidade igual a 1. Dessa maneira, conseguimos determinar o fluxo máximo do vértice origem s até o vértice destino t, correspondendo ao número de caminhos disjuntos por aresta.

1.1 Implementação do Algoritmo para Identificação de Caminhos Disjuntos

A implementação do algoritmo para identificar caminhos disjuntos em arestas foi realizada seguindo os seguintes passos:

1. **Definição das Capacidades das Arestas:** Todas as arestas do grafo foram configuradas com capacidade igual a 1. Essa configuração garante que cada aresta possa transportar no máximo um fluxo, permitindo que o fluxo máximo calculado corresponda diretamente ao número de caminhos disjuntos por aresta entre os vértices origem s e destino t.

2. **Cálculo do Fluxo Máximo:** O algoritmo de Ford-Fulkerson calcula o fluxo máximo do vértice s até o vértice t. Devido às capacidades iguais das arestas, o valor do fluxo máximo representa exatamente o número de trajetos disjuntos (caminhos disjuntos por arestas) entre esses vértices.

3. **Extração das Arestas Utilizadas no Fluxo:** Após o cálculo do fluxo máximo, extraímos todas as arestas que transportaram fluxo (ou seja, arestas com fluxo maior que zero). Essas arestas representam as conexões utilizadas nos caminhos disjuntos identificados pelo algoritmo.

4. **Criação de um Novo Grafo com as Arestas Utilizadas:** Com base nas arestas extraídas, criamos um novo grafo que contém apenas essas arestas com fluxo positivo.

5. **Identificação Iterativa de Caminhos Disjuntos:** No novo grafo, realizamos uma busca iterativa por caminhos do vértice s ao vértice t. Para cada

caminho encontrado, imprimimos a sequência de vértices que formam o caminho e removemos as arestas utilizadas nesse caminho do grafo. Esse processo é repetido até que não seja mais possível encontrar caminhos adicionais entre s e t.

Essa abordagem permite a identificação de todos os caminhos disjuntos por aresta entre s e t, garantindo que cada caminho encontrado seja independente dos demais em termos de compartilhamento de arestas.

2 Resultados

Nesta seção, apresentamos os resultados dos testes realizados com dois tipos de grafos: **Grafo Aleatório** e **Grafo Malha (Grid)**. Para cada tipo de grafo, foram realizados experimentos com instâncias de tamanhos diferentes.

2.1 Grafo Aleatório

Apresentamos os resultados obtidos para o Grafo Aleatório na Tabela 1, na Figura 1 e no Gráfico **??**.

Table 1: Resultados para o Grafo Aleatório

Tamanho	**Média Arestas**	**Média Caminhos**	**Máx. Caminhos**	**Tempo Médio (ms)**
100	513	5.3	9	34
500	7205	13.7	21	838
1,000	10547	12.7	21	1,603
10,000	108380	8.3	15	10,846

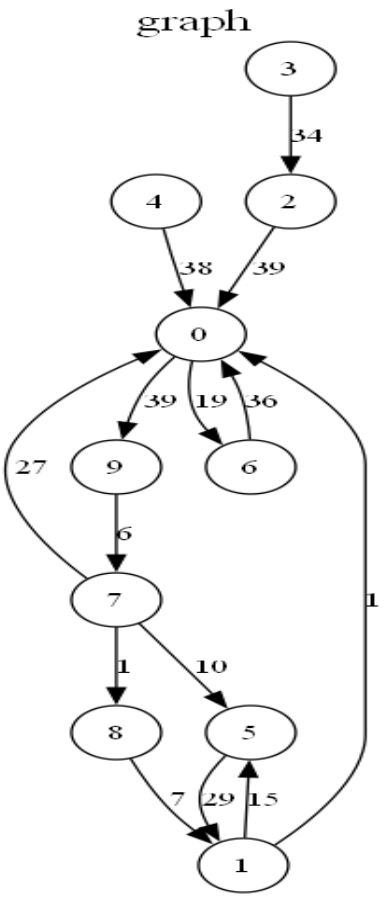

Figure 1: Exemplo de grafo gerado pelo gerador aleatório

2.2 Grafo Malha (Grid)

Apresentamos os resultados obtidos para o Grafo Malha nas Tabelas 2, na Figura 2 e no Gráfico 4.

Table 2: Resultados para o Grafo Malha

Tamanho do Grafo	**Número de Caminhos Disjuntos**	**Tempo de Execução (ms)**
1,000	1	5
50,000	2	1,776
300,000	2	13,357
1,600,000	2	90,990

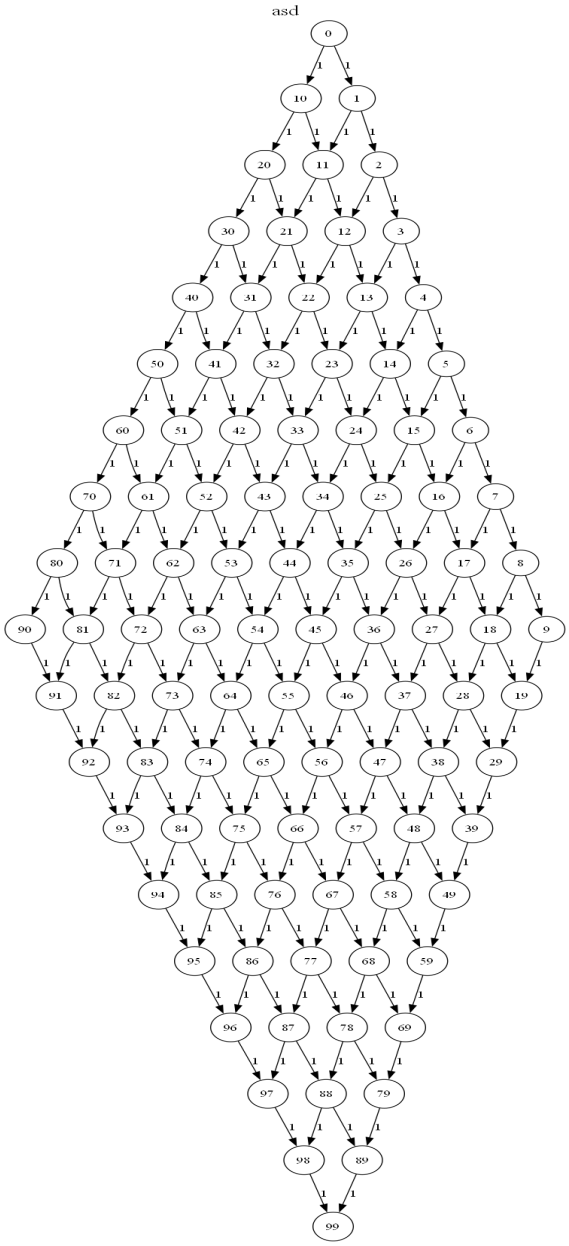

Figure 2: Exemplo do grafo Malha (Grid Graph)

2.2.1 Detalhes do Grafo Malha e Função de Controle de Crescimento

O Grafo Malha (Grid Graph) é uma estrutura que, naturalmente, apresenta um crescimento quadrático em relação ao número de vértices, devido à sua configuração de matriz bidimensional onde cada vértice está conectado aos seus vizinhos adjacentes. Para evitar que o número de arestas se torne excessivamente alto, adotei um abate do valor calculado total.

Criei uma função que determina a largura (Pn) do grafo com base na altura (Pm) e no índice da iteração atual (idx). A fórmula aplicada foi a seguinte:

$$Pn = \text{sizes}[idx]$$

$$Pm = \frac{\text{sizes}[idx]}{10 \times (idx + 1)}$$

Onde:

- **sizes** é um vetor que contém os diferentes tamanhos utilizados para altura e largura, definido como `vec![100, 500, 1_000, 2_000]`.
- **altura** (Pm) é definida diretamente pelo valor atual de `sizes[idx]`.
- **largura** (Pn) é calculada para limitar o crescimento do número de arestas, reduzindo a taxa de expansão conforme o índice da iteração aumenta.

Essa função permite que, à medida que aumentamos o tamanho do grafo, a largura não cresça de forma linear ou exponencial, mas de acordo com uma taxa controlada.

Além disso, observou-se que, mesmo com as restrições impostas pela função de controle de crescimento, o número máximo de caminhos disjuntos no Grafo Malha direcionado permaneceu consistentemente em dois. Este comportamento reflete a estrutura intrínseca do grafo, onde a conectividade entre os vértices impede a existência de múltiplos caminhos independentes.

3 Gráficos

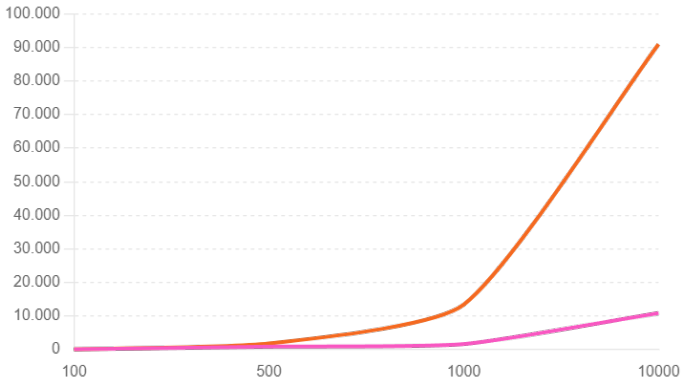

Figure 3: Crescimento do Tempo de Execução em Função do Número de Vértices proposto

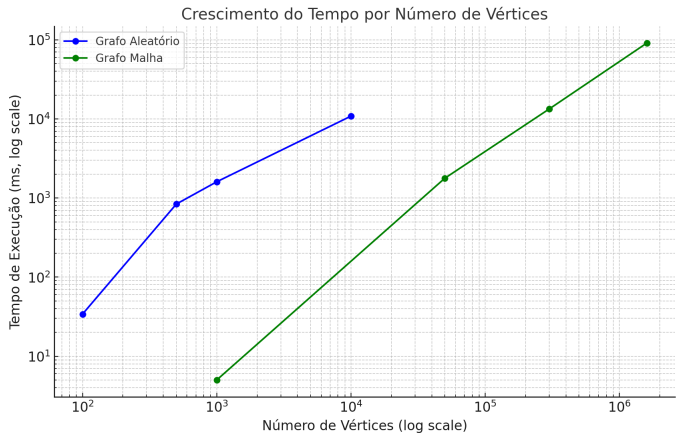

Figure 4: Crescimento do Tempo de Execução em Função do Número de Vértices ntotal no grafo

4 Referências

References

[1] **Grid Graph**. Disponível em: https://mathworld.wolfram.com/GridGraph.html

[2] **Aplicações de fluxo máximo para achar caminhos disjuntos**. Disponível em: https://courses.grainger.illinois.edu/cs473/sp2010/notes/17-maxflowapps.pdf