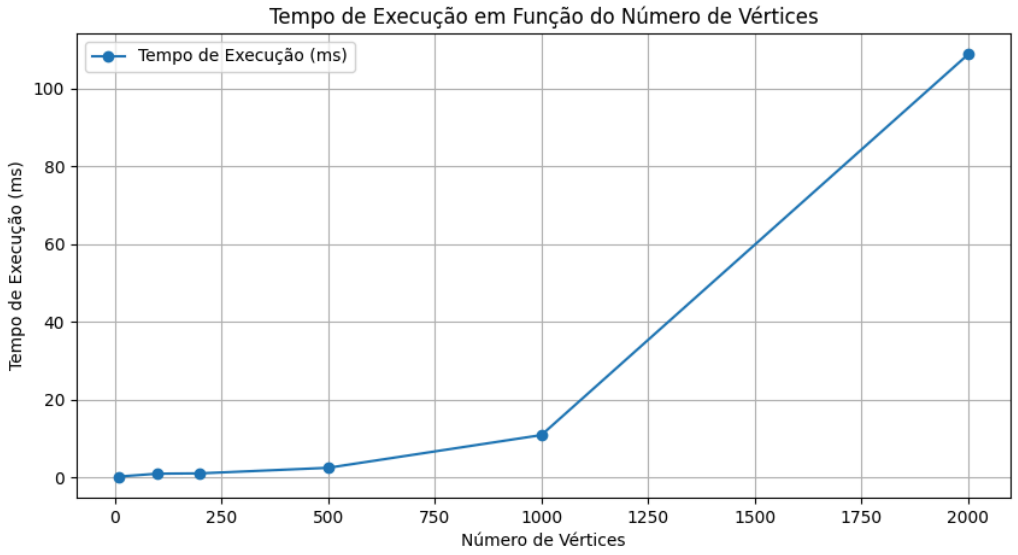
Implementação 3

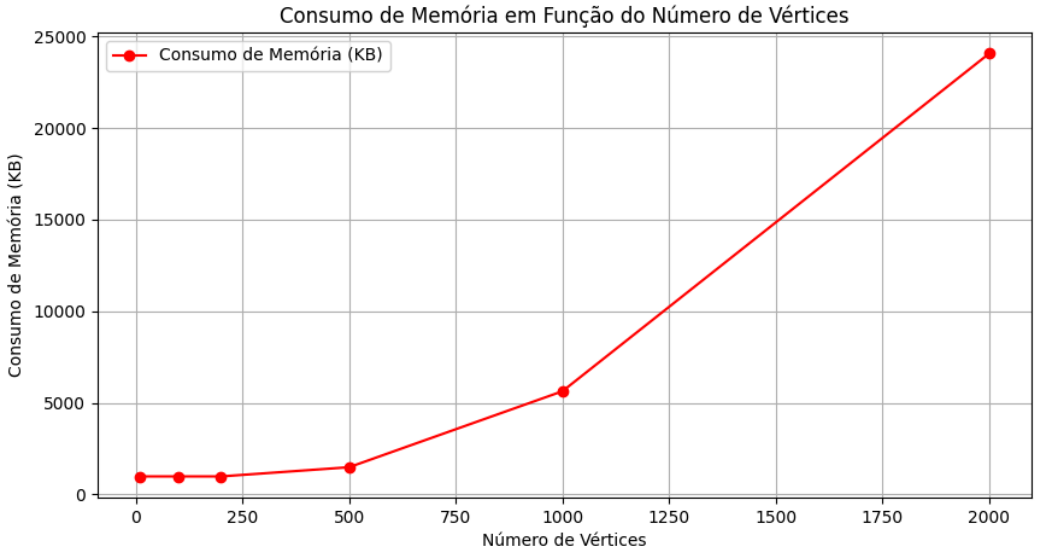
Nesta implementação 3, foi utilizado o algoritmo de Edmonds-Karp para encontrar os caminhos adjacentes nos grafos, colocando as arestas com peso binário (0 ou 1) de forma a saturar cada aresta do fluxo em que se encontra um caminho aumentante.

Para os testes, foram utilizados dois tipos de grafos, sendo um aleatório e outro bipartido, além de que para cada tipo foram utilizados grafos com 10, 100, 200, 500, 1000 e 2000 vértices cada.

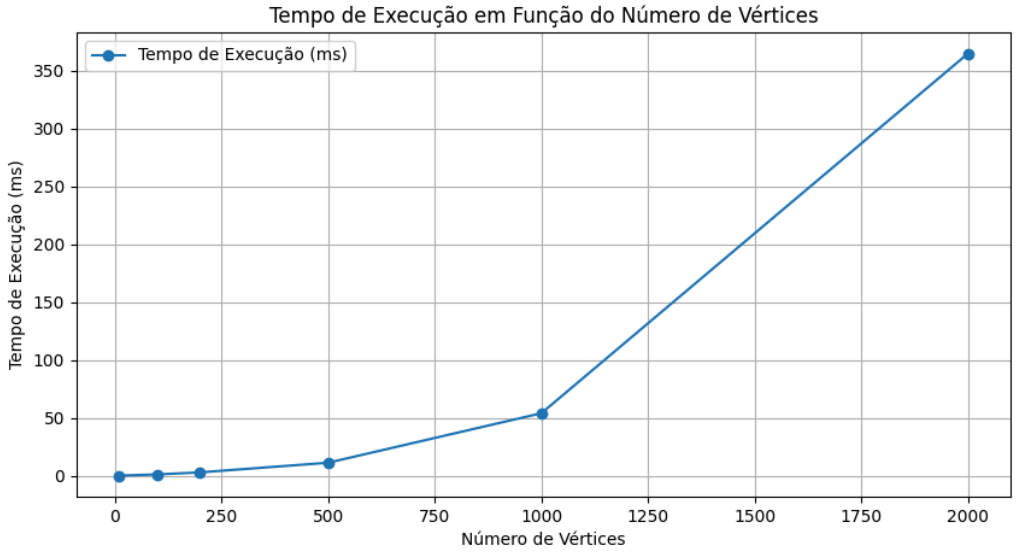
Foram medidos os tempos de execução para cada grafo utilizando o nanoTime da linguagem java e executando o mesmo 10 vezes, fazendo uma média da soma desses tempos. Para o consumo de memória foi usado o totalMemory do java também.

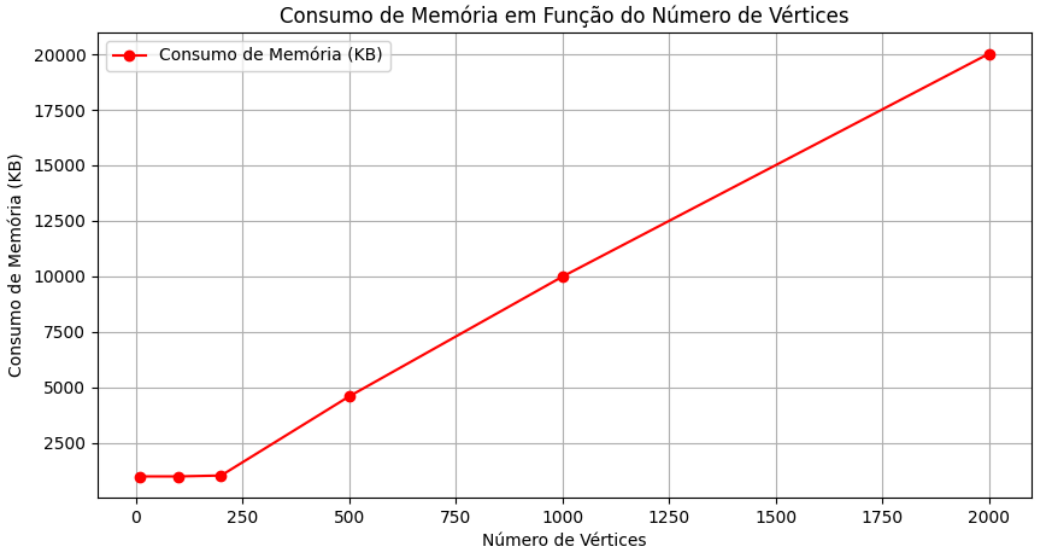
Em Grafos Aleatórios





Em Grafos Bipartidos





Análise do Tempo de Execução

1. Grafo Aleatório:
   - O tempo de execução cresce significativamente à medida que o número de vértices aumenta. Para 10 vértices, o tempo é de aproximadamente 0.21 ms, mas para 2000 vértices, ele cresce para cerca de 108.73 ms.
   - Esse crescimento indica uma complexidade crescente do algoritmo quando aplicado a grafos aleatórios maiores, o que é esperado, pois o número de caminhos possíveis e a saturação das arestas afetam a quantidade de processamento necessária.

2. Grafo Bipartido:
   - Observa-se um aumento maior no tempo de execução em comparação com grafos aleatórios, especialmente em grafos maiores. Para 2000 vértices, o tempo de execução chega a aproximadamente 364.82 ms, indicando que o algoritmo pode ser mais exigente em termos de processamento ao lidar com a estrutura bipartida.
   - Essa diferença pode ser explicada pela estrutura dos grafos bipartidos, onde o número de caminhos disjuntos pode ser maior devido à forma como os vértices são conectados entre os dois conjuntos, exigindo mais iterações do algoritmo de Edmonds-Karp.

Análise do Consumo de Memória

1. Grafo Aleatório:
   - O consumo de memória aumenta de forma relativamente gradual, começando em 983 KB para 10 vértices e chegando a 24064 KB para 2000 vértices.
   - Esse aumento está de acordo com o crescimento do grafo, pois o algoritmo precisa armazenar a estrutura de dados do grafo e as capacidades residuais para cada caminho. No entanto, o crescimento da memória parece ser mais estável e proporcional ao tamanho do grafo.

2. Grafo Bipartido:
   - Para grafos bipartidos, o consumo de memória segue uma tendência semelhante, mas com valores levemente mais altos para instâncias maiores, como 2000 vértices, onde o consumo chega a 20032 KB.
   - Isso sugere que, embora o consumo de memória cresça com o número de vértices, a estrutura bipartida não afeta significativamente o consumo de memória em relação aos grafos aleatórios.

Conclusão

- Eficiência do Algoritmo: O algoritmo de Edmonds-Karp, embora eficaz para encontrar caminhos disjuntos, mostra um aumento considerável no tempo de execução conforme o número de vértices cresce, especialmente em grafos bipartidos.
- Consumo de Memória: O consumo de memória é similar para ambos os tipos de grafos, mas os grafos bipartidos parecem consumir ligeiramente mais memória em instâncias grandes.

- Grafo Bipartido vs. Aleatório: Em geral, os grafos bipartidos têm tempos de execução mais elevados devido à sua estrutura, que gera mais caminhos possíveis e demanda mais do algoritmo.