Teoria dos Grafos
Algoritmo Guloso do Vizinho Mais Próximo e o Algoritmo de Força-Bruta

Hércules Teixeira 18.2.8072

[1]Universidade Federal de Ouro Preto - Campus ICEA

Resumo. *Implementar e testar os algoritmos: Vizinho Mais Próximo, e o de Força-Bruta. Além de identificar vantagens e desvantagens de abordagens exatas e heurísticas em diferentes configurações de grafos.*

1. Introdução

O problema de o Caixeiro viajante (PCV) tem como objetivo estabelecer a menor rota possível para completar um percurso sem passar duas vezes pela mesma cidade retornando a origem. Ele é um problema de otimização NP-difícil inspirado na necessidade dos vendedores em realizar entregas em diversos locais (as cidades) percorrendo o menor caminho possível, reduzindo o tempo necessário para a viagem e os possíveis custos com transporte e combustível. (CERVIERI, Alexandre; 2021)

2. Algoritmos

Apesar do Problema do Caixeiro Viajante ser NP-Difícil, há diversas estratégias para encontrar soluções satisfatórias. Estratégias que não garantem a solução ótima são chamadas heurísticas; já estratégias que dão essa garantia são as exatas.

2.1. Força-Bruta

Um exemplo de abordagem exata é o Algoritmo de Força-Bruta, que resolve o problema examinando todas as rotas possíveis. Apesar da garantia da melhor rota possível seu tempo de execução é usualmente proibitivo.

2.1.1. Pseudocódigo

Podemos ver na Figura 1, um pseudocódigo, e em sequencia temos um breve resumo sobre cada linha do código;

Algoritmo 3: FORCA BRUTA.

Entrada: (i) Um grafo $G = (V, E, w)$.
Saída: (i) O ciclo hamiltoniano de custo mínimo C^{best}.

```
1 cost^min ← inf;
2 C^best ← null;
3 para cada permutação de vértices C faça
4     C ← C ∪ {C[0]};
5     cost^C ← Calcule o custo da rota C;
6     se cost^min > cost^C então
7         cost^min ← cost^C;
8         C^best ← C;
9 retorna C^best;
```

Figura 1. Pseudocódigo: Força-Bruta

Linha 1: Armazena custo mínimo;

Linha 2: Armazena o melhor caminho;

Linha 3: Realiza a permutação da lista e percorre as Listas geradas pela permutação;

Linha 4: Adiciona o primeiro elemento para fechar o ciclo;

Linha 5: Calcula o custo de cada permutação;

Linha 6: Testa se o caminho atual é melhor do que o melhor caminho encontrado;

Linha 7: Recebe o custo da melhor rota atual;

Linha 8: Recebe a lista da melhor rota atual;

Linha 9: Retorna a Melhor rota encontrada.

2.2. Vizinho Mais Próximo

Já um exemplo de abordagem heurística é o Algoritmo Guloso do Vizinho Mais Próximo, que caminha sempre para o vizinho mais próximo ainda não visitado até fechar o ciclo. Esse algoritmo raramente encontra a rota ótima, mas executa em uma fração de segundos, mesmo para grafos com muitos vértices.

2.2.1. Pseudocódigo

Podemos ver na Figura 2, um pseudocódigo, e em sequencia temos um breve resumo sobre cada linha do código;

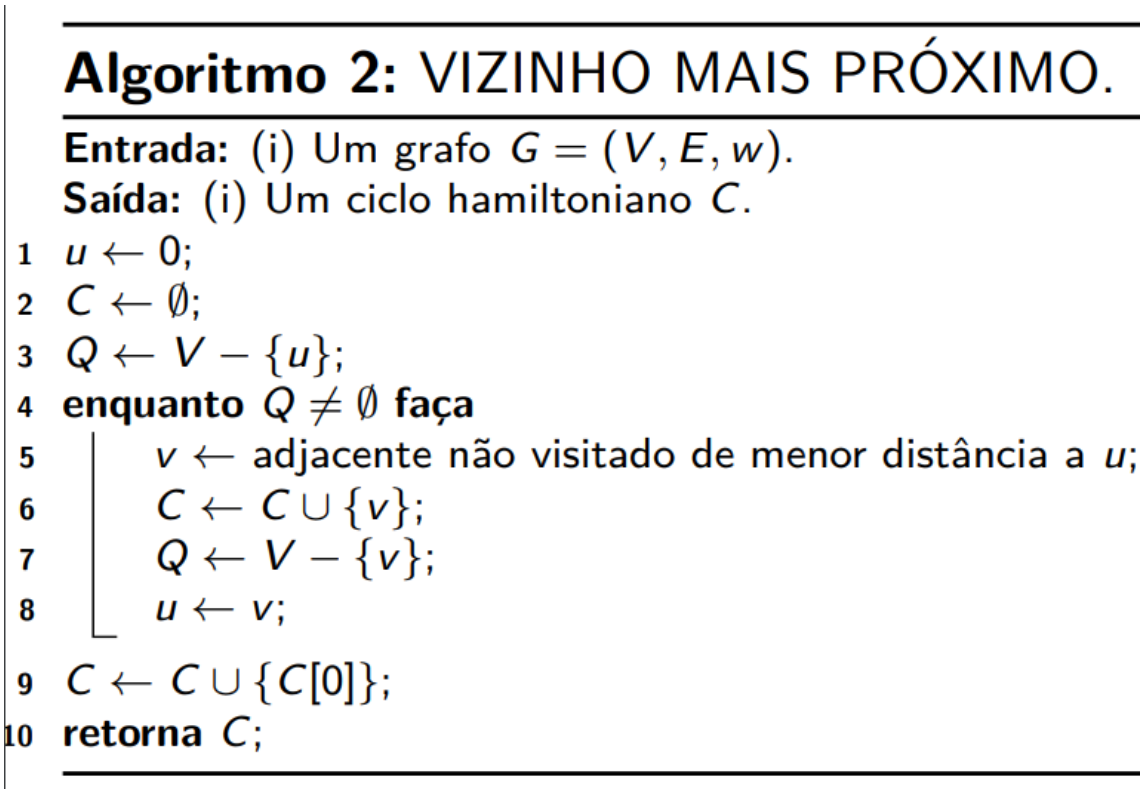

Figura 2. Pseudocódigo Vizinho Mais Próximo

Linha 1: Armazena vértice atual, sendo iniciado com zero;

Linha 2: Armazena caminho final, sendo iniciado por nulo;

Linha 3: Lista de vértices à serem processados, removendo o elemento u pois não precisamos calcular ele;

Linha 4: Enquanto a lista de vértices à serem processados for diferente de nulo;

Linha 5: Recebe o vértice não visitado com menor distancia a u;

Linha 6: Adiciona o vizinho mais próximo a lista;

Linha 7: Remove o vértice já calculado;

Linha 8: Atualiza o vértice a ser calculado;

Linha 9: Adiciona o ultimo vértice para fechar o ciclo;

Linha 10: Retorna a Melhor rota encontrada.

3. Resultados

Foram executados diversos testes embasados nos critérios propostos pelo enunciado do trabalho, na Figura 3 temos os resultados obtidos com os testes realizados. Já na Figura 4

podemos ver o gráfico resultante da conversão da tabela para meios visuais.

3.1. Tempo de Execução

Tempo de execução				
|V|	w min	w max	Guloso em s	Força-Bruta em s
5	1	5	0	0
5	1	500	0	0
8	1	5	0	0,15
8	1	500	0	0,15
10	1	5	0	14,40
10	1	500	0	16,26
20	1	5	0	E.T
20	1	500	0	E.T
50	1	5	0	E.T
50	1	500	0	E.T

Figura 3. Tabela Tempo de Execução

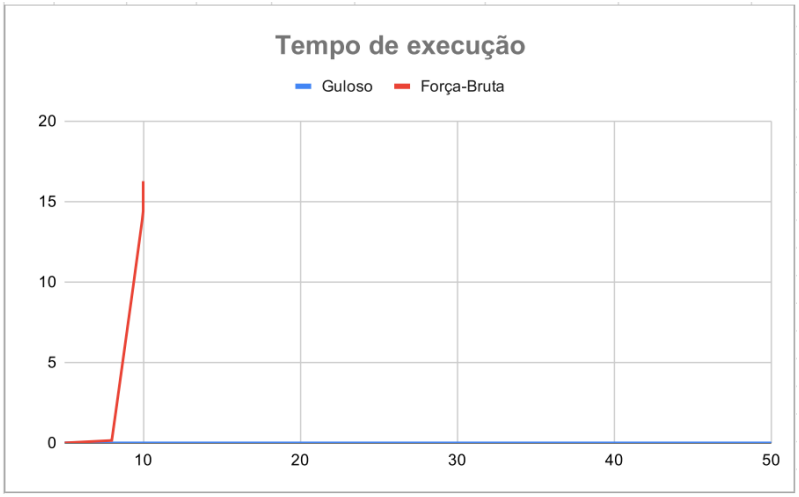

Figura 4. Gráfico Tempo de Execução

3.2. Hardware Utilizado

Processador: Intel Core i5-7200U

Memória RAM: 8 GB DDR4 2133 MHz

Placa de vídeo: Integrada Intel HD Graphics 620

Armazenamento: SSD Kingspec M2 Nvme

SO: Windows 10 x64

4. Conclusão

Analisando os dados obtidos, podemos ver a discrepância entre os dois algoritmos testados; como por exemplo o mais visível é o tempo de execução, na qual o Força-Bruta com um grafo de apenas 20 vértices já ultrapassou o tempo de execução em 10 minutos, tornando excelente em grafos pequeno e proibitivo em casos maiores. Já o algoritmo do Vizinho Mais Próximo (Guloso), tem uma execução em frações de segundo, porém o resultado obtido normalmente é pior do que o melhor caso.

Foi executado os testes levando em consideração o peso de cada aresta (w max), na qual não pode se perceber nenhuma alteração no tempo final de execução, já que o custo de uma soma é O(1).

Finalmente podemos concluir que a implementação de cada um dos dois algoritmos está diretamente ao grafo que será utilizado, pois deve analisar os prós e contras de cada caso. Ou seja, basicamente escolher entre dois fatores: tempo de execução baixo com o resultado impreciso, ou o resultado exato com um tempo de execução exponencial.

Referências

FONSECA, George; 2021. Notas de Aula. Universidade Federal de Ouro Preto. Disponível em: https://www.moodlepresencial.ufop.br/. Acesso em: 19 abr 2021.

CERVIERI, Alexandre; PY, Mónica – Algoritmo para a resolução do problema do caixeiro viajante [Em Linha]. Porto Alegre: Instituto de Informática - UFRGS, 2000. [Consult. 15 Abr. 2009].