Implementação de um Algoritmo Heurístico para Solucionar o Problema do Caixeiro Viajante

Letícia Capitani Trapp Sampaio[1]

[1]Departamento de Ciência da Computação
Universidade do Estado de Santa Catarina (UDESC) – Joinville, SC – Brazil

capitanileticiats@gmail.com

1. Problema do Caixeiro Viajante (TSP)

O Problema do Caixeiro Viajante (TSP) é um problema clássico em otimização combinatória, amplamente estudado no campo da ciência da computação. Este problema consiste em determinar a rota mais curta que permita visitar um conjunto de pontos e retornar ao ponto de origem.

O TSP é categorizado como um problema combinatório NP-difícil, o que implica que não se conhece um algoritmo de tempo polinomial capaz de resolver todas as suas instâncias de maneira eficiente. Consequentemente, o tempo necessário para encontrar a solução ideal aumenta exponencialmente à medida que o número de pontos a serem visitados cresce.

Determinadas instâncias do TSP podem ser tão complexas que exigiriam um tempo exorbitante para serem solucionadas de forma otimizada. Em virtude dessa complexidade, foram desenvolvidos algoritmos heurísticos que, embora não garantam soluções ótimas, fornecem resultados eficazes em um período de tempo viável.

1.1. Modelagem do Problema

O Problema do Caixeiro Viajante (TSP) é formulado como um problema de otimização combinatória em um grafo completo $G(V, A)$, onde $V = \{1, 2, ..., n\}$ representa o conjunto de vértices e $A = \{(i, j) \mid i, j \in V, i \neq j\}$ é o conjunto de arestas. Cada aresta $(i, j) \in A$ tem associado um custo c_{ij}, que representa a distância de percorrer diretamente de i a j.

O objetivo do TSP é encontrar um ciclo Hamiltoniano de custo mínimo em G, que visita cada vértice $i \in V$ exatamente uma vez e retorna ao vértice inicial. Formalmente, seja x_{ij} uma variável binária que indica se a aresta (i, j) pertence à solução do TSP:

$$x_{ij} = \begin{cases} 1, & \text{se a aresta } (i,j) \text{ está no ciclo Hamiltoniano,} \\ 0, & \text{caso contrário.} \end{cases}$$

Para garantir que a solução represente um ciclo Hamiltoniano válido e evitar *subtours*, são impostas as seguintes restrições:

1. **Visitação de todos os vértices**: Cada vértice $i \in V$ deve ser visitado exatamente uma vez.
$$\sum_{j \in V \setminus \{i\}} x_{ij} = 1, \quad \forall i \in V$$
2. **Partida de cada vértice**: Deve haver exatamente uma aresta saindo de cada vértice $i \in V$.
$$\sum_{j \in V \setminus \{i\}} x_{ji} = 1, \quad \forall i \in V$$

3. **Evitar *subtours***: Para evitar a formação de *subtours*, é necessário que as variáveis x_{ij} sejam consistentes com um único ciclo Hamiltoniano.

$$\sum_{(i,j)\in S} x_{ij} \leq |S| - 1, \quad \forall S \subset V, 2 \leq |S| \leq |V| - 1$$

4. **Conectividade do ciclo**: A solução final deve ser um único ciclo Hamiltoniano que conecta todos os vértices.

$$\sum_{(i,j)\in A} x_{ij} = |V|$$

Esta formulação matemática e as restrições são essenciais para resolver o TSP de maneira eficiente, garantindo a obtenção de uma solução ótima ou aproximada em tempo viável, dada a natureza NP-difícil do problema.

2. Heurística em ATSP

No Problema do Caixeiro Viajante Assimétrico (ATSP), um vendedor deve visitar um conjunto de cidades exatamente uma vez e retornar à cidade de origem, minimizando a distância total percorrida. Ao contrário do Problema do Caixeiro Viajante Clássico (TSP), onde a distância entre duas cidades é simétrica (ou seja, a mesma em ambas as direções), no ATSP as distâncias podem variar dependendo da direção da viagem.

Esta assimetria torna o ATSP mais desafiador computacionalmente, pois não pode ser simplificado usando as mesmas abordagens eficientes desenvolvidas para o TSP. Em vez disso, estratégias heurísticas são frequentemente empregadas para encontrar soluções aproximadas em um tempo razoável.

Para abordar esse problema,é proposto uma formulação linear relaxada do ATSP, que permite a criação de *subtours* ao não impor a restrição de que deve haver uma única rota conectando todos os vértices. Esta abordagem visa explorar soluções próximas da ótima de maneira eficiente, embora não garanta a solução exata devido à natureza NP-difícil do ATSP.

3. Exemplo e Resultados Obtidos

Com a eliminação da restrição que obriga uma única rota, agora é possível gerar várias rotas menores em vez de uma grande rota. Desse modo, é disposto as seguintes etapas do algorítmo:

1. Definida uma matriz de custos que representa as distâncias entre cada par de locais.
2. Para encontrar os *subtours*, se inicia na casa inicial e visita as casas com menor custo, acumulando o custo total.
3. Simula a combinação dos *subtours* em uma única tour, assumindo uma simplificação.
4. Após visitar todos os locais, é necessário retornar ao ponto inicial para completar o circuito.

Para ilustrar esse processo, é considerado neste exemplo uma tabela de custos para 5 locais, onde a matriz abaixo mostra o custo entre cada par de localizações. Esta matriz reflete as diferentes distâncias de ida e volta entre alguns pares de locais, exemplificando os casos onde há vias diferentes.

	0	1	2	3	4
0	0	3.5	9	2.1	12
1	6.7	0	4	13.5	8.6
2	11.9	7.8	0	9	2.5
3	4	12	6.5	0	10
4	3	11	5.8	14.2	0

Através da combinação heurística é possível fundir os *subtours* com o menor aumento de custos. Sendo assim, o resultado obtido é a seguinte sequencia, iniciada no 0 e de custo total 28.8.

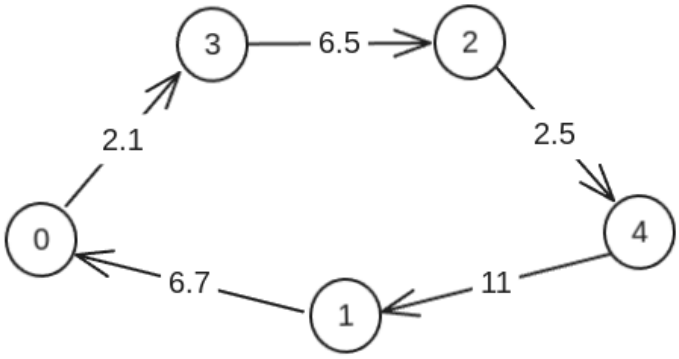

Figure 1. Primeiro exemplo aplicado no código

4. Teste com Matrizes Grandes e Complexidade

Para obter um estudo mais aprofundado, propôs-se testar o código da Heurística do ATSP com matrizes de maiores dimensões. Inicialmente, realizou-se um teste com uma matriz 5x5. No entanto, considerando que o Problema do Caixeiro Viajante envolve percorrer um grande número de cidades, é essencial avaliar o desempenho do algoritmo em escalas maiores.

Portanto, o código foi testado com matrizes de diferentes dimensões: 10x10, 100x100, 1000x1000 e 10000x10000. Foram analisados os seguintes tempos de execução:

Matriz	**Tempo (s)**
10x10	0.000035
100x100	0.000213
1000x1000	0.003038
10000x10000	0.193289

Table 1. Tempos de execução para diferentes tamanhos de matriz

Ao representar graficamente esses resultados, observa-se que a complexidade do algoritmo cresce aproximadamente em $O(n^2)$.

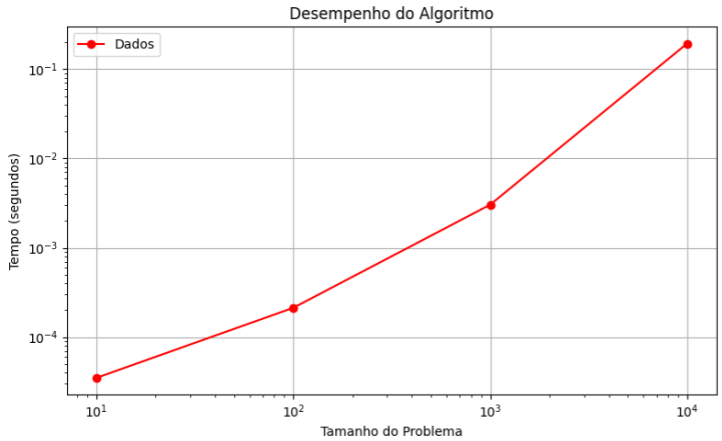

Figure 2. Crescimento do tempo de execução em função do tamanho da matriz

Os resultados demonstram que, à medida que o tamanho da matriz aumenta, o tempo de execução cresce de forma quadrática, o que é consistente com a complexidade esperada para algoritmos desse tipo.

5. Conclusão

A implementação do algoritmo heurístico ATSP para resolver o Problema do Caixeiro Viajante Assimétrico (ATSP) demonstra uma abordagem eficiente para obter soluções rapidamente em problemas de menor escala ou como uma etapa inicial em algoritmos mais complexos. No entanto, como uma heurística gulosa, ele não garante soluções ótimas, especialmente quando aplicado a problemas com grandes valores de n.

No contexto do ATSP, onde a assimetria das distâncias entre as cidades adiciona uma camada de complexidade, a formulação de soluções aproximadas através de heurísticas, como a relaxação linear, permite explorar eficazmente o espaço de soluções. Essas técnicas são fundamentais para lidar com a dificuldade NP-difícil do problema, fornecendo soluções que, embora não sejam ideais, são adequadas para muitas aplicações práticas.

6. Compilação

Compilar e executar o programa no Ubuntu:

1. **Compilar os Arquivos:** Abra o terminal e navegue até o diretório onde foi baixado os arquivos usando o comando `cd`. Em seguida, use o comando `gcc` seguido do seguinte comando:
   ```
   gcc -o atsp atsp.c main.c
   ```

2. **Executar o Programa:** Após compilar com sucesso, execute o programa usando o seguinte comando:
   ```
   ./atsp
   ```

Referências

TRUNG, Quang. **Traveling Salesman Problem: Exact Solutions vs. Heuristic vs. Approximation Algorithms**. Baeldung. Disponível em: https://www.baeldung.com/cs/tsp-exact-solutions-vs-heuristic-vs-approximation-algorithms#:\~:text=Lin-Kernighan\%20is\%20a\%20local,a\%20local\%20minimum\%20is\%20reached. Acesso em: 30 jun. 2024.

GAO, Yuan. **Heuristic Algorithms for the Traveling Salesman Problem**. Medium. Disponível em: https://medium.com/opex-analytics/heuristic-algorithms-for-the-traveling-salesman-problem-6a53d8143584. Acesso em: 30 jun. 2024.

NILSSON, Christian. **Heuristics for the Traveling Salesman Problem**. 2003.

MESTRIA, Mário. **A Hybrid Heuristic Algorithm for te Clustered Traveling Salesman Problem**. Espírito Santo, 2016.

CASTILLO, José. **A heuristic for the traveling salesman problem based on a continuous approximation**. 1999.

MUNARI, Pedro. **Heurística do Vizinho Mais Próximo, Problema do Caixeiro Viajante, Pesquisa Operacional, UFSCar**. Yoyube, 2022. Disponível em: https://www.youtube.com/watch?v=tEryMeECijE. Acesso em: 30 jun. 2024.