Trabalho de Implementação 1 - Heurísticas e Metaheurísticas

Francisco Teixeira Rocha Aragão - 2021031726

Data de entrega: 21 de novembro de 2024

1 Introdução

O presente trabalho busca resolver de maneira aproximada o problema do caixeiro viajante (TSP), fazendo uso de heurísticas gulosas em sua implementação. Como o problema pertence a classe NP, faz-se necessário o uso de tais estratégias, sendo feito especificamente uma heurística gulosa no trabalho. Abaixo encontra-se mais informações sobre a implementação além dos resultados obtidos.

2 Heurística utilizada

Primeiramente sobre a heurística utilizada, a estratégia implementada no trabalho refere-se a heurísticas construtivas, ou seja, heurísticas em que a solução é construída do zero, desde o início até a resolução do problema. Iniciando-se assim de uma solução vazia, obtendo-se uma solução parcial a cada iteração em que ao final transforma-se em uma solução completa válida.

Desse modo, a estratégia utilizada foi baseada em uma abordagem gulosa, em que a cada ponto (ou cidade), o próximo trajeto escolhido é aquele com a menor distância. Desse modo, inicia-se a partir de uma cidade (será explicado mais frente), e a cada iteração novas cidades são adicionadas no caminho até todas as cidades serem visitadas, voltando assim ao vértice inicial resolvendo o problema. Com isso, garante-se a validade da solução retornada, em que a cada iteração acrescenta-se uma nova cidade não visitada anteriormente, terminando o algoritmo até visitar a última cidade, retornando ao ponto inicial.

Sobre o ponto inicial escolhido, duas abordagens foram implementadas e comparadas: a primeira escolhendo o primeiro nó recebido como cidade inicial, e a segunda escolhendo o nó com a menor distância em relação a todos os outros. A segunda abordagem foi pensada para assim escolher um nó que estivesse mais próximo dos demais, podendo ser um fator que auxilie o desempenho do algoritmo.

3 Execução e Resultados

O código foi desenvolvido em C++ e os testes foram realizados em uma máquina com debian 12, 16GB de ram e processador I5-11 geração. Sua execução pode ser realizada com os seguintes comandos:

```
// compilação
make

// limpar arquivos gerados
make clean

// rodar programa
make run ARGS="<pasta com instâncias de entrada> <tipo da cidade inicial>
// <tipo de cidade inicial> = 0 para usar a primeira cidade e 1 para usar cidade central
```

Vale destacar que o arquivo de entrada foi encontrado no site TSPLIB95, presente nas referências no trabalho. Além disso, a execução foi realizada 5 vezes para cada instância, com as médias dos resultados disponível nas tabelas abaixo. O algoritmo guloso implementado em ambas as versões são exatos, não

obtendo mudanças entre as execuções, então a média dos resultados foi obtida para encontrar o valor médio do tempo de execução.

Assim, os resultados encontrados para cada abordagem foram sumarizados nas tabelas 3 e 4. É possível notar que ambas as estratégias obteram resultados semelhantes, com pouca diferença entre o erro em cada abordagem, mesmo que de maneira geral a segunda estratégia baseada no nó geral tenha obtido melhores resultados de custo na média. Sobre isso, os valores de custo foram no máximo 50% piores do que o ótimo (mais especificamente, o maior erro foi de 41.89%). Em todo caso, o ponto fundamental que pode ser verificado nos resultados refere-se ao tempo de execução, que ficou bem reduzido nas duas abordagens. Mesmo que a estratégia do nó central tenha obtidos tempo ligeiramente maiores, a diferença é praticamente irrelevante.

Sumarizando a comparação dos resultados das diferentes abordagens, temos a tabela 5 que possui a melhor abordagem para cada uma das instâncias. Percebe-se que de maneira geral, a estratégia utilizando o nó central como inicio do caminho performou melhor. Mesmo que a abordagem com o primeiro nó como inicial possuindo vantagem em alguns dos casos maiores, a hipótese é que isso foi causado por propriedades de como as instâncias estão distribuídas, embora mais testes possam ser feitos para comprovar tal fato futuramente. Em todo caso, foi testado para uma instância adicional maior ambas as abordagens, em um problema envolvendo 3795 cidades, com os resultados mostrados abaixo na tabela 1 e 2. Assim, em um primeiro momento, existe a ideía que escolher o nó central como inicial possui melhores resultados ao se extender o problema em maiores instâncias. Porém novamente, mais testes são necessários, além de otimizações na implementação para melhorar tanto o custo obtido quanto o tempo.

Instância	**Ótimo**	**Custo**	**Tempo**	**Erro Percentual (%)**
fl3795	28772	36841.4	114.198	28.04

Table 1: Heurística gulosa - Primeiro nó como inicial

Instância	**Ótimo**	**Custo**	**Tempo**	**Erro Percentual (%)**
fl3795	28772	36145.5	106.698	25.62

Table 2: Heurística gulosa - Nó central como inicial

4 Referências

Slides sobre heurísticas construtivas e heurísticas gulosas
Formato das instâncias de entrada e resultados ótimos

Instância	Ótimo	Custo	Tempo	Erro Percentual (%)
att48	10628	12861	0.001903	21.00
berlin52	7542	8980.92	0.0010034	19.08
st70	675	805.531	0.0033856	19.34
pr76	108159	153462	0.0022632	41.89
rat99	1211	1564.72	0.0043974	29.22
kroA100	21282	26856.4	0.0044204	26.14
kroB100	22141	29155	0.0043634	31.68
kroC100	20749	26327.4	0.0044004	26.89
kroD100	21294	26950.5	0.0044182	26.58
kroE100	22068	27587.2	0.0043896	25.00
lin105	14379	20362.8	0.00492	41.61
pr124	59030	69299.4	0.0072172	17.40
pr136	96772	120778	0.009604	24.81
pr144	58537	61650.7	0.010979	5.32
kroA150	26524	33609.9	0.0118756	26.72
kroB150	26130	32825.7	0.0121312	25.63
pr152	73682	85703	0.01233	16.31
rat195	2323	2761.96	0.0226642	18.90
kroA200	29368	35798.4	0.0241298	21.90
kroB200	29437	36981.6	0.0243252	25.63

Table 3: Heurística gulosa - Primeiro nó como inicial

Instância	Ótimo	Custo	Tempo (s)	Erro Percentual
att48	10628	12964	0.0019	21.97%
berlin52	7542	9140.13	0.0010	21.20%
st70	675	783.72	0.002179	16.11%
pr76	108159	151142	0.0024008	39.75%
rat99	1211	1493.92	0.0043878	23.37%
kroA100	21282	25781.6	0.0044748	21.14%
kroB100	22141	27208.5	0.0046738	22.89%
kroC100	20749	24173.9	0.0045104	16.50%
kroD100	21294	27695.5	0.004607	30.06%
kroE100	22068	26884.6	0.0045522	21.82%
lin105	14379	19182.4	0.005191	33.40%
pr107	44303	52145.5	0.0052502	17.71%
pr124	59030	68321.4	0.0074482	15.74%
pr136	96772	116045	0.0098506	20.42%
pr144	58537	64161.5	0.011302	9.61%
kroA150	26524	32786.8	0.0123896	23.62%
kroB150	26130	34583.7	0.0124628	32.37%
pr152	73682	80161.3	0.0128008	8.80%
rat195	2323	2830.1	0.0231814	21.83%
kroA200	29368	37701.5	0.0246762	28.38%
kroB200	29437	36893.6	0.024664	25.33%

Table 4: Heurística gulosa - Nó central inicial

Instância	**Menor Erro Percentual**	**Abordagem**
att48	21.01%	Primeiro nó como inicial
berlin52	19.08%	Primeiro nó como inicial
st70	16.11%	Nó central inicial
pr76	39.75%	Nó central inicial
rat99	23.37%	Nó central inicial
kroA100	21.14%	Nó central inicial
kroB100	22.89%	Nó central inicial
kroC100	16.50%	Nó central inicial
kroD100	26.57%	Primeiro nó como inicial
kroE100	21.82%	Nó central inicial
lin105	33.40%	Nó central inicial
pr107	17.71%	Nó central inicial
pr124	15.74%	Nó central inicial
pr136	20.42%	Nó central inicial
pr144	5.32%	Primeiro nó como inicial
kroA150	23.62%	Nó central inicial
kroB150	25.63%	Primeiro nó como inicial
pr152	8.80%	Nó central inicial
rat195	18.89%	Primeiro nó como inicial
kroA200	21.90%	Primeiro nó como inicial
kroB200	25.33%	Nó central inicial

Table 5: Comparação de abordagens - Menor erro percentual por instância