Colônia de Formigas (ACO) e o Problema do Caixeiro Viajante (PCV)

Nome: Henrique Araújo Lima

1 Introdução

Este documento descreve os resultados obtidos ao aplicar o algoritmo Colônia de Formigas (ACO) ao problema do Caixeiro Viajante (PCV). O objetivo do presente trabalho é desenvolver um ACO para resolver o problema do PCV para diversas instâncias dele.

2 Metodologia

2.1 Algoritmo de Colônia de Formiga

O ACO é um do algoritmos mais tradicionais e conhecidos dentre os algoritmos de Inteligência Coletiva. Este algoritmo, baseia-se no comportamento real das formigas para solucionar problemas de encontrar caminhos que podem ser representados em grafos.

2.2 Problema do Caixeiro Viajante

O problema do caixeiro viajante (PCV ou TSP ? *Traveling Salesman Problem*) é assim definido: dado um número finito de cidades e o custo de viagem entre cada par, deve-se encontrar o caminho que passa por todas as cidades e retorna para o ponto inicial com custo mínimo, passando por cada cidade apenas uma vez. Embora seja simples de definir, o TSP é um problema combinatório clássico comprovadamente NP-Difícil com aplicações nas áreas de logística, genética, telecomunicações e neurociência, entre outras.

O problema do caixeiro viajante pode ser definido formalmente da seguinte maneira:

Seja $G = (V, A)$ um grafo, onde V é um conjunto de N vértices e A é um conjunto de arestas. Seja C a matriz de distâncias associada com A. O TSP consiste em determinar o menor ciclo hamiltoniano, ou seja, o menor ciclo que passa por todos os vértices somente uma vez. Um caso específico do problema do caixeiro viajante é o simétrico, onde o caminho que leva da cidade i à cidade j é o mesmo que leva da cidade j à cidade i com igual distância.

2.3 Descrição Técnica

2.3.1 Input

A entrada do algoritmo pode ser dada de duas maneiras:

- Conjunto de n pares ordenados, onde o par $p_i = (x_i, y_i)$ representa a i-ésima cidade.

- Matriz de distâncias definida apenas para os valores da diagonal superior, onde a célula a_{ij} representa a distância entre a cidade i à cidade j. No entanto, essa célula pode inicialmente estar vazia, pois estamos analisando o problema do caxeiro viajante simétrico, isto é, a distância d_1 de i à j é a mesma de distância d_2 de j à i ($d_1[ij] = d_2[ji]$).

2.3.2 Output

O algoritmo deve ser capaz de processar para os dois tipos de entrada uma saída que contenha uma boa solução para o problema. Diversos parâmetros serão alterados para cada instância e contrastados para verificar a eficiência do algoritmo. São eles:

- Q = taxa de depósito de ferômonio de cada formiga;
- ρ = taxa de evaporação do ferômonio;
- k = quantidade de iterações do algoritmo;

3 Experimentos

3.1 Instância 01 (M6)

3.1.1 Descrição

Tipo de input: matriz de distância
Melhor output: 2011

Input

	Évora	Faro	Badajoz	Córdoba	Madrid
Aveiro	353	582	372	641	559
Évora		231	99	426	502
Faro			331	326	750
Badajoz				269	403
Córdoba					424

Figura 1: Input para a instância 01 do PCV

3.1.2 Experimentos

Tabela 1: Experimentos do problema m6

P	Q	K	I	Melhor Caminho	Pior Caminho	Media dos Caminhos	Tempo de Execução
0.5	1	50	10	2011	2480	2150.20	0.309s
0.5	1	50	50	2011	2511	2217.41	1.120s
0.5	1	50	100	2011	2783	2324.62	2.211s
0.8	1	50	10	2011	2641	2324.51	0.365s
0.8	1	50	50	2011	2465	2287.35	1.115s
0.8	1	50	100	2011	2454	2218.76	2.504s

3.2 Instância 02 (M15)

3.2.1 Descrição

Tipo de input: pontos euclideanos
Melhor output: -

Input

Tabela 2: Coordenadas

	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14	C15
X	200	300	300	1000	600	1700	1400	1200	200	1200	500	1000	1900	400	1600
Y	300	700	1700	1900	1400	1600	800	500	1000	1100	900	900	1000	500	200

3.2.2 Experimentos

Tabela 3: Experimentos do m15

P	Q	K	I	Melhor Caminho	Pior Caminho	Media dos Caminhos	Tempo de Execução
0.5	1	50	10	10486.247	14721.285	12445.451	4.84s
0.5	1	50	50	10354.534	14687.542	12357.533	18.351s
0.5	1	50	100	9417.354	15132.357	12221.806	38.35s
0.8	1	50	10	11442.701	13145.221	12294.522	4.004s
0.8	1	50	50	10552.254	15415.524	12554.114	17.565s
0.8	1	50	100	10441.511	14785.455	12225.415	39.551s

3.3 Instância 03 (M29)

3.3.1 Descrição

Tipo de input: pontos euclideanos
Melhor output: 27603

Input

Tabela 4: Mapa real - coordenadas

Cidade	X	Y
1	20833.3333	17100.0000
2	20900.0000	17066.6667
3	21300.0000	13016.6667
4	21600.0000	14150.0000
5	21600.0000	14966.6667
6	21300.0000	16500.0000
7	22183.3333	13133.3333
8	22583.3333	14300.0000
9	22683.3333	12716.6667
10	23616.6667	15866.6667
11	23700.0000	15933.3333
12	23883.3333	14533.3333
13	24166.6667	13250.0000
14	25149.1667	12365.8333
15	26133.3333	14500.0000
16	26150.0000	10550.0000
17	26283.0000	12766.6667
18	26433.3333	13433.3333
19	26550.0000	13850.0000
20	26733.3333	11683.3333
21	27026.1111	13051.9444
22	27096.1111	13415.8333
23	27153.6111	13203.3333
24	27166.6667	9833.3333
25	27233.3333	10450.0000
26	27233.3333	11783.3333
27	27266.6667	10383.3333
28	27433.3333	12400.0000
29	27462.5000	12992.2222

3.3.2 Experimentos

Tabela 5: Experimentos para o M29

P	Q	K	I	Melhor Caminho	Pior Caminho	Media dos Caminhos	Tempo de Execução
0.5	1	50	10	58542.114	76455.303	67154.42	41s
0.5	1	50	50	58145.267	79454.415	68420.415	3min10s
0.5	1	50	100	57428.187	82244.143	69484.245	6min57s
0.8	1	50	10	60144.673	82411.154	71654.874	37s
0.8	1	50	50	59421.116	79547.365	69412.421	3min49s
0.8	1	50	100	59510.144	83444.451	71445.114	6min50s

3.4 Instância 04 (M38)

3.4.1 Descrição

Tipo de input: pontos euclideanos
Melhor output: 27603

Input

3.4.2 Experimentos

Tabela 6: Segundo mapa real - coordenadas

Cidade	X	Y
1	11003.611100	42102.500000
2	11108.611100	42373.888900
3	11133.333300	42885.833300
4	11155.833300	42712.500000
5	11183.333300	42933.333300
6	11297.500000	42853.333300
7	11310.277800	42929.444400
8	11416.666700	42983.333300
9	11423.888900	43000.277800
10	11438.333300	42057.222200
11	11461.111100	43252.777800
12	11485.555600	43187.222200
13	11503.055600	42855.277800
14	11511.388900	42106.388900
15	11522.222200	42841.944400
16	11569.444400	43136.666700
17	11583.333300	43150.000000
18	11595.000000	43148.055600
19	11600.000000	43150.000000
20	11690.555600	42686.666700
21	11715.833300	41836.111100
22	11751.111100	42814.444400
23	11770.277800	42651.944400
24	11785.277800	42884.444400
25	11822.777800	42673.611100
26	11846.944400	42660.555600
27	11963.055600	43290.555600
28	11973.055600	43026.111100
29	12058.333300	42195.555600
30	12149.444400	42477.500000
31	12286.944400	43355.555600
32	12300.000000	42433.333300
33	12355.833300	43156.388900
34	12363.333300	43189.166700
35	12372.777800	42711.388900
36	12386.666700	43334.722200
37	12421.666700	42895.555600
38	12645.000000	42973.333300

Tabela 7: Experimentos para o M38

P	Q	K	I	Melhor Caminho	Pior Caminho	Media dos Caminhos	Tempo de Execução
0.5	1	50	10	19784.354	24784.547	21579.521	2min
0.5	1	50	50	18741.621	24887.651	21042.442	6min20s
0.5	1	50	100	17952.235	24887.411	20511.541	19min3s
0.8	1	50	10	19117.143	23874.441	21551.211	2min10s
0.8	1	50	50	19133.608	24115.349	21575.441	7min18s
0.8	1	50	100	19172.548	24354.866	21547.998	19min40s

4 Conclusão

A partir dos resultados obtidos, notou-se que se o espaço de busca é menor, isto é, a quantidade de cidades é menor e por consequência a quantidade de caminhos hamiltonianos, então neste cenário o algoritmo responde bem próximo do melhor resultado real. Além disso, nota-se que para investigar o algoritmo para diversas iterações ($I = 100$) os valores tendem a melhorar de forma geral do que os valores comparados para poucas iterações ($I = 10$). Embora o resultado melhore final do caminho melhore, o tempo cresce drasticamente.

O único problema que encontrou o melhor caminho foi o m6. E o caminho encontrado foi: $c = [4, 2, 1, 3, 0, 5, 4]$. Cujo comprimento é dado por $|c| = 2011$.