Algoritmo de Otimização de Colônia de Formigas para o Problema do Caixeiro Viajante

Vítor Rezende Silva

Agosto 2024

1 Introdução

O algoritmo de otimização de colônia de formigas basea-se no comportamento de uma colônia de formigas para buscar soluções ótimas para problemas. O objetivo desse trabalho é implementar esse algoritmo para encontrar uma solução para o problema do caixeiro viajante (PCV) usando como base uma matriz de distâncias contendo as distâncias entre cidades de um dado conjunto.

Para executar o algoritmo os seguintes valores devem ser fornecidos:

- num_iteracoes: Número de iterações do algoritmo.
- Q: Valor definido = 100
- p: Taxa de evaporação de um feromônio. Valor sugerido = 0.5
- f: Quantidade inicial de feromônios em cada aresta. Valor sugerido = 10^6.
- a: Influência do ferômonio. Valor sugerido = 1.
- b: Influência do custo do caminho. Valor sugerido = 5.
- tipo: Tipo de método para atualizar os feromônios. Pode ser "AS" e "EAS"

Além disso, foi decidido que o número de formigas utilizado vai ser igual ao número de cidades.

Como foi deixado evidente anteriormente, dois métodos de atualização de feromônios foram implementados.

O primeiro é o Ant System, que é feito depois que todas as formigas passaram sobre uma aresta (i, j), descrito pela função:

$$\tau_{ij} = (1 - \rho)\tau_{ij} + \sum_{k=1}^{m} \Delta\tau_{ij}^{k}$$

Já o segundo é o Elitism Ant System, similar ao anterior, porém com elitismo, onde a formiga com melhor percurso reforça o seus feromônios sobre eles, descrito pela função:

$$\tau_{ij} = (1-\rho)\tau_{ij} + \sum_{k=1}^{m} \Delta\tau_{ij}^{k} + \epsilon\Delta\tau_{ij}^{bs}$$

2 Testes

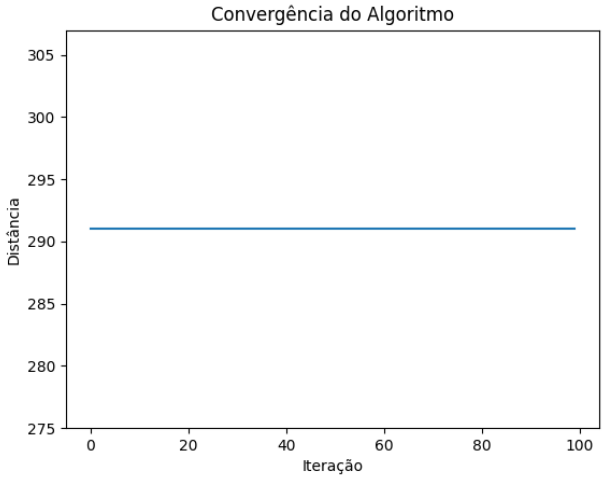
Os testes foram executados utilizando dois modelos de entrada, o arquivo sgb128_dist.txt, uma representação de 128 cidades e lau15_dist.txt, com 15 cidades. Ambos os modelos são fortemente conectados.

Realizando alguns teste com o arquivo lau15_dist.txt percebi que ele estava encontrando o melhor resultado com apenas 1 iteração do algoritmo. Para testar isso eu escrevi o seguinte script em python para executar o algoritmo 100 vezes com apenas 1 iteração e ver a variação de resultados.

```
from ColoniaFormigas import *
import matplotlib.pyplot as plt

file = open("iteracoes1.txt", 'w')

iteracoes = []
resultados = []
rotas = []
melhor_caminho = 1000000
cont = 0

filename = "sgb128_dist.txt"
for i in range(100):
    print(i)
    if i % 2 == 0:
        S, L = main(filename, 1, 100, 0.5, 10**-6, 1, 5, "AS")
    else:
        S, L = main(filename, 1, 100, 0.5, 10**-6, 1, 5, "EAS")
    file.write(str(L) + " " +str([S]) + "\n")
    iteracoes.append(i)
    resultados.append(L)
    if L < melhor_caminho:
        melhor_caminho = L
        cont += 1

print(L)
print(melhor_caminho)
print(cont)
plt.plot(iteracoes, resultados)
```

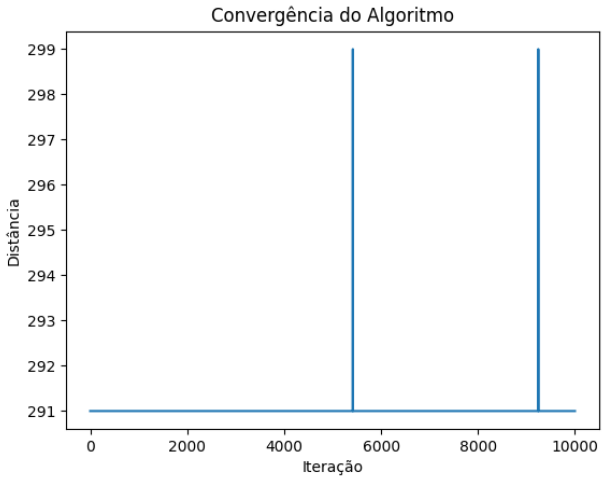
```
plt.xlabel('Iteração')
plt.ylabel('Distância')
plt.title('Convergência do Algoritmo Genético')
plt.show()
```

O gráfio gerado foi o seguinte:



Perceba que em todas as iterações o melhor caminho foi o mesmo, 291, o melhor caminho definido para esta entrada.

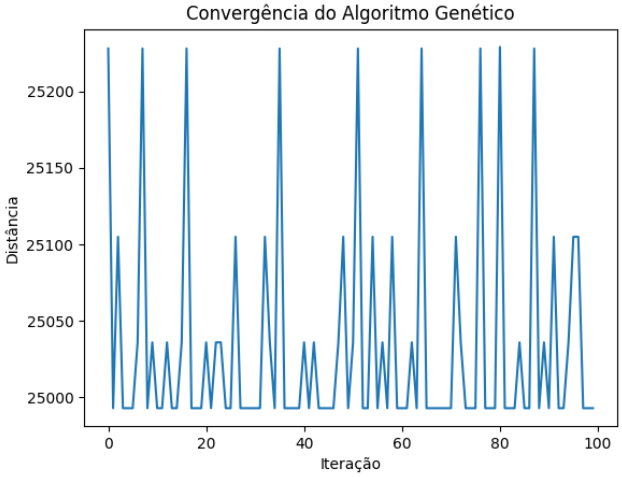
O teste foi reaplicado para ver se os resultados se repetiam, agora com 10000 iterações, e os resultados foram os seguintes:



Nesse caso ja foi possível observar dois casos onde o resultado encontrado não foi o ótimo, 299 nas duas situações.

O algoritmo aplicado à configuração lau15_dist.txt apresentou ótimos resultados, por isso resolvi aplicar o mesmo script ao conjunto sgb128_dist.txt, ja que ele apresenta um conjunto de cidades muito maior.

Os resultados obtidos foram os seguintes:



Dentre todos os testes para esta entrada o melhor resultado obtido foi 24993.