

```

1 import matplotlib.pyplot as plt
2 import numpy as np
3 import numpy.random as npr
4 from utils import *
5
6 ''' Met à jour les phéromones : chemin plus court = plus de phéromones '''
7 def update_pheromones(path, path_length, pheromone_matrix):
8
9     for i in range(0, len(pheromone_matrix)):
10         pt1 = path[i-1]
11         pt2 = path[i]
12
13         pheromone_matrix[pt1][pt2] += 1/path_length
14
15 ''' Pour un point donné, connaissant les points qui n'ont pas encore été parcourus
16 par la fourmi, calcule la probabilité pour chaque points'''
17 def gen_probab(last_point, pt_left, pheromone_matrix, distance_matrix, a, b):
18     probas = []
19     total = 0
20     for p in pt_left:
21         ro = 1 / distance_matrix[last_point][p]
22         delta = pheromone_matrix[last_point][p]
23         proba = ro ** a * delta ** b
24
25         probas.append(proba)
26         total += proba
27     probas = [proba / total for proba in probas] # On divise par la somme des
28     probabilités pour obtenir un SCE (somme des probabilités = 1)
29     return probas
30
31 ''' Point suivant choisi par la fourmi en fonction des probabilités de chaque
32 point'''
33 def next_point(last_point, points_left, pheromone_matrix, distance_matrix, a, b):
34     batch = gen_probab(last_point, points_left, pheromone_matrix, distance_matrix, a,
35     b)
36     return npr.choice(points_left, 1, p=batch)[0]
37
38 ''' Code le comportement de la fourmi : choisit les points un par un au hasard en
39 tenant compte de la distance et des phéromones, en gardant en mémoire les points
40 restants
41 De plus, la longueur du chemin est calculée au fur et à mesure de l'avancement de
42 la fourmi, en utilisant des distances précalculées pour accélérer le programme
43 a et b sont les paramètres de réglage modifiant l'importance de la distance et des
44 phéromones dans le choix de la fourmi '''
45 def fourmi(points, pheromone_matrix, distance_matrix, a, b):
46     nb_points = len(points)
47     path = [0] # le chemin sera construit dans cette liste. exemple de chemin : [0, 6,
48     4, 2, 3, 5, 1] (remarque : les chemins sont fermés : le dernier point est relié au
49     premier)
50     points_left = list(points)
51     path_length = 0
52     for i in range(nb_points):
53         n_p = next_point(path[-1], points_left, pheromone_matrix, distance_matrix, a, b)
54
55         del points_left[points_left.index(n_p)]
56
57         path_length += distance_matrix[path[-1]][n_p]
58
59         path.append(n_p)
60     return path, path_length

```

```
51
52 ''' Simule nb_gen générations de fourmis, avec a = 4 et b = 1, et renvoie le meilleur
    chemin.
53 Remplit en plus deux graphes : longueur du chemin par génération et longueur du
    meilleur chemin trouvé jusqu'alors à la n-ième génération'''
54 def generations(points, nb_gen, distance_matrix):
55     nb_points = len(points)
56     pheromone_matrix = np.ones((nb_points, nb_points))
57
58     best_path = []
59     best_path_length = 100000000
60
61     length_graph = []
62     best_length_graph = []
63
64     for i in range(nb_gen):
65
66         path, path_length = fourmi(list(range(1,nb_points)), pheromone_matrix,
        distance_matrix, 4, 1)
67
68         if path_length < best_path_length:
69             best_path_length = path_length
70             best_path = path
71
72         length_graph.append(path_length)
73         best_length_graph.append(best_path_length)
74
75         update_pheromones(path, path_length, pheromone_matrix)
76         update_pheromones(best_path, best_path_length, pheromone_matrix)
77
78     return best_path, pheromone_matrix, distance_matrix, length_graph,
    best_length_graph
79
80 ''' Fonction principale, qui exécute generations() avec des paramètres de test, et
    affiche les graphes utiles '''
81 def main():
82     nb_points = 20
83     nb_gen = 1000
84     points = gen_points(nb_points)
85     distance_matrix = create_distance_matrix(points)
86
87     path, pheromone_matrix, _, length_graph, best_length_graph = generations(points,
    nb_gen, distance_matrix)
88
89     plt.subplot(1, 2, 1)
90
91     display_path(path, points)
92     plot_points(points)
93     plt.title("Meilleur chemin")
94
95     plt.subplot(1, 2, 2)
96
97     plt.plot(list(range(nb_gen)), length_graph, label='Longueur du chemin')
98     plt.plot(list(range(nb_gen)), best_length_graph, label='Meilleur longueur')
99
100    plt.legend()
101    plt.title("Evolution de la longueur du chemin au cours des générations")
102    plt.show()
103
104    main()
```