**Méthodes d’optimisation pour l’aide à la décision**

**TD3**

**Exercice 1 : Problème d’affectation**

Le responsable du personnel d'une entreprise est confronté au problème suivant : afin de réaliser un certain nombre de tâches différentes bien identifiées, il dispose de personnels dont la compétence est diverse et pour lesquels il faut envisager une formation.

Il y a autant de personnes que de tâches et chaque personne se verra confier une et une seule tâche.

Le coût de formation dépend de la personne et de la tâche.

Dans le cas de 5 tâches, il a constitué le tableau suivant donnant le coût de formation pour chaque personne :

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Tâche 1 | Tâche 2 | Tâche 3 | Tâche 4 | Tâche 5 |
| Personne 1 | 15 | 40 | 5 | 20 | 20 |
| Personne 2 | 22 | 33 | 9 | 16 | 20 |
| Personne 3 | 40 | 6 | 28 | 0 | 26 |
| Personne 4 | 8 | 0 | 7 | 25 | 60 |
| Personne 5 | 10 | 10 | 60 | 15 | 5 |

Il voudrait connaître le nombre de solutions possibles à ce problème.

Pour déterminer l'affectation des 5 personnes aux tâches qui conduira au coût total de formation le moins élevé possible, le responsable du personnel propose de considérer les personnes dans un ordre donné, personne 1 puis 2, etc. et de leur affecter la tâche de moindre coût encore non affectée.

1. A quelle solution cette politique conduit-elle ici ? Quel est son coût ?
2. Pensez-vous que cette méthode fournisse la "meilleure" solution ?
3. Formuler le problème à l’aide d’un programme linéaire. Combien de solutions réalisables possède ce problème ?
4. Le code Python permettant de résoudre le problème linéaire correspondant à ce problème est donné par le fichier « Simplex-5.py ». Exécuter ce code sur Google Colab et comparer le coût de cette solution avec celle de la question 1.
5. Une deuxième solution à ce problème est donnée par le code python du fichier GA-5.py utilisant l’algorithme génétique pour résoudre ce même problème. Exécuter ce code sur Google Colab et comparer le coût de cette solution avec les deux précédentes.
6. Modifier les codes des deux algorithmes fournis (Simplex et GA) pour résoudre le même problème si on a 100 tâches. Déterminer le coût de la solution optimale pour chaque algorithme et comparer les performances des deux algorithmes (en termes de temps d’exécution).

Voici le code Pyhon permettant de générer aléatoirement un tableau de coût de tâches pour 100 personnes :

import numpy as np

from pulp import LpProblem, LpVariable, lpSum, LpMinimize

# Générer un tableau aléatoire de 100 personnes et 100 tâches avec des coûts aléatoires

num\_people = 100

num\_tasks = 100

cost\_matrix = np.random.randint(1, 101, size=(num\_people, num\_tasks))

# Imprimer le tableau généré

print("Tableau aléatoire de coûts de formation :")

print(cost\_matrix)

1. Même question si le nombre de tâches est égal à 1000 et puis 10000. Pouvez-vous fournir la solution optimale donnée par chaque algorithme ?