RÉPUBLIQUE DU CAMEROUN
Paix-Travail-Patrie
****
MINISTÈRE DE L' ENSEIGNEMENT SUPÉRIEUR
****
UNIVERSITÉ DE NGAOUNDÉRÉ
****
FACULTÉ DES SCIENCES
****
B.P.456 NGAOUNDÉRÉ

REPUBLIC OF CAMEROON
Peace-Work-Fatherland
****
MINISTRY OF EDUCATION
****
THE UNIVERSITY OF NGAOUNDÉRÉ
****
FACULTY OF SCIENCES
****
P.O.BOX . 456 NGAOUNDÉRÉ

DÉPARTEMENT DES MATHÉMATIQUES ET INFORMATIQUE(MI).

****

***** M1SLED *****

UE : INF461- COMPLEXITÉ DES ALGORITHMES AVANCÉS.

Contrôle Continu et Travail Personnel de l'Étudiant.

CHOIX DU PROBLÈME : Multiple Travelling Salesman Problem(MTSP).

NOMS ET PRÉNOMS	MATRICULES
ALBERT WEFE	18A179FS
BADAWE FILS AIME	18A176FS
DJONYANG TOBOKBE SYLVAIN	17A191FS
MOHAMADOU NOUA	17A683FS

Enseignant: Dr. NDAM NJOYA HAROUNA

Année académique 2021/2022

PLAN DETAILLÉ DU TRAVAIL.

INTRODUCTION.

I- DESCRIPTION DU PROBLÈME(MTSP).

II- FORMULATION DU PROBLÈME(MTSP).

III- RÉSOLUTION DU PROBLÈME(MTSP).

IV- STRUCTURE DES DONNÉES.

V- TEST ET DISCUSSION.

- a- Instance de test.
- b- Discussion des résultats.
- c- Analyse de complexité.

CONCLUSION.

INTRODUCTION.

Le problème du Voyageur de Commerce a été étendu au cas où *m* voyageurs de commerce doivent visiter *n* villes, défini comme le Problème de Voyageurs de Commerce Multiples ou *Multiple Traveling Salesman Problem* (MTSP). Ce problème est très proche du problème de Tournées de Véhicules ou *Vehicle Routing Problem* (VRP) pour lequel une flotte de véhicules est disponible, et chaque véhicule est limité par les distances qu'il peut parcourir.

Une des extensions les plus étudiées fut ensuite de considérer les transports de marchandises. Ces marchandises sont récoltées chez des clients au cours du trajet et ramenées à un dépôt unique. De plus, les objets ramassés possèdent un poids et une capacité maximale à ne pas dépasser est associée à chaque véhicule. L'objectif est donc d'assigner des clients à des véhicules. Cette extension est connue sous le nom de problème de Tournées de Véhicules avec Contraintes de Capacités (*Capacited Vehicle Routing Problem* ou CVRP) .

Il est donc question dans ce travail de présenter et formuler le Problème de Voyageurs de Commerce Multiples ou *Multiple Traveling Salesman Problem* (mTSP), de présenter quelques approches d' implémentation ou de résolution de ce problème, que des résultats de ce dernier; enfin, de donner un aperçu sur sa complexité.

I- DESCRIPTION DU PROBLÈME(MTSP).

Le problème du voyageur de commerce est le suivant : un voyageur de commerce souhaite faire la tournée des villages de sa région puis rentrer chez lui (certaines fois le retour chez lui est ignoré). Son objectif est de minimiser la distance totale parcourue. Un tel problème est appelé problème "d' optimisation" car le but est de trouver les paramètres qui minimisent (ou maximisent) une certaine quantité.

II- FORMULATION DU PROBLÈME(MTSP).

Formellement, le problème consiste, étant donné un graphe pondéré et un sommet initial, à trouver un cycle (ou un chemin) élémentaire passant par tous les sommets du graphe et de longueur minimale. Remarque : Pour de nombreux cas

d’ utilisation, on considère souvent un graphe complet pondéré. Ce problème est connu comme étant un exemple de problème NP-Complet.

III- RÉSOLUTION DU PROBLÈME(MTSP).

Afin de trouver la solution à ce problème, une solution simple consiste à énumérer l’ ensemble des chemins possibles, à calculer pour chacun sa longueur et à sélectionner celui ou un de ceux qui sont minimaux. Pour cela, on adapte un parcours en profondeur:

```
def travellingSalesman (graph, sourceNode) :


    bestLength = float("inf")
    bestPath = None


    def exhaustive (remainingNodes, currentNode, currentPath, currentLength, graph) :

        if not remainingNodes :
            print("Found Hamiltonian path", repr(currentPath), "of size", currentLength)
            nonlocal bestLength, bestPath
            if currentLength < bestLength :
                bestLength = currentLength
                bestPath = currentPath

        else :
            for neighbor, weight in graph[currentNode] :
                if neighbor in remainingNodes :
                    otherNodes = [x for x in remainingNodes if x != neighbor]
                    exhaustive(otherNodes, neighbor, currentPath + [neighbor], currentLength + weight, graph)


    otherNodes = [x for x in range(len(graph)) if x != sourceNode]
    exhaustive(otherNodes, sourceNode, [sourceNode], 0, graph)


    return (bestPath, bestLength)

############################################################

graph = [[(1, 1), (2, 7), (5, 3)], [(0, 1), (2, 1), (5, 1)], [(0, 7), (1, 1)], [(4, 2), (5, 2)], [(3, 2), (5, 5)], [(0, 3), (1, 1), (3, 2), (4, 5)]]
(result, length) = travellingSalesman(graph, 0)
print(repr(result))
print("La taille du chemin de tous les nœuds la plus faible obtenue est :  ",length)
```

IV- STRUCTURE DES DONNÉES.

Si le problème donné peut être divisé en sous-problèmes plus petits et que ces sous-problèmes plus petits sont à leur tour divisés en problèmes encore plus petits, et dans ce processus. En outre, les solutions optimales aux sous-problèmes contribuent à la solution optimale du problème donné.

De nos recherches, Il en ressort existence deux principales façons de le faire :

- ✓ De haut en bas : On commence à résoudre le problème en le décomposant. Si on voie que le problème a déjà été résolu, on renvoie la réponse enregistrée. Si cela n' a pas été résolu, on le résout puis on enregistre la réponse.
- ✓ De bas en haut : On analyse le problème et on voie l' ordre dans lequel les sous-problèmes sont résolus puis on commence à résoudre du sous-problème trivial jusqu' au problème donné. Dans ce processus, il est garanti que les sous-problèmes soient résolus avant de résoudre le problème. **Ceci est appelé programmation dynamique.**

Un pseudo-code serait le suivant :

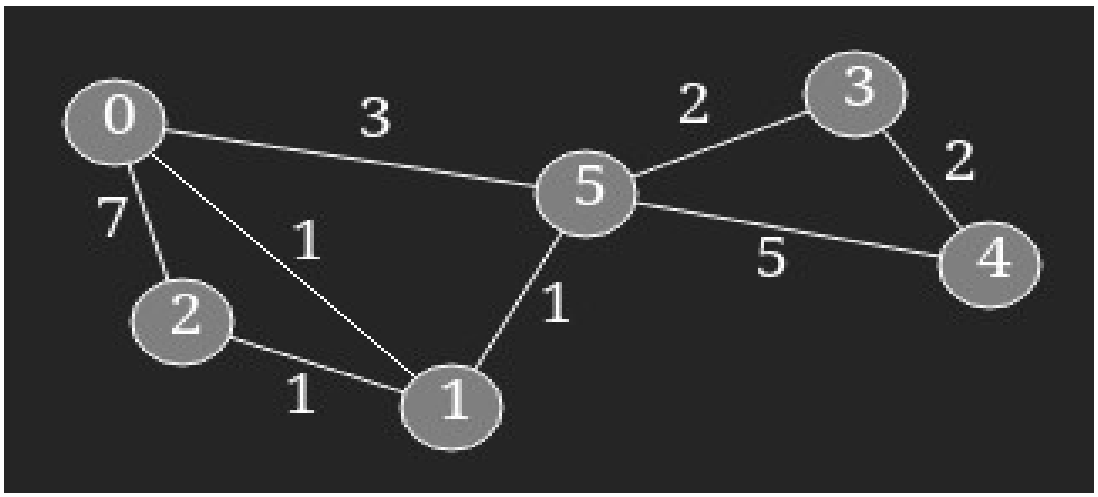
[Pseudo-code 2] Algorithme dynamique de résolution du problème du voyageur de commerce

Prérequis : Vecteur de villes et coût total, une fonction de coûts c
Résultat : le plus court chemin qui visite tous les points du vecteur V
Initialisation de la valeur maximale des coûts des distances du chemin recherché à D[] = ∞,
Initialisation d' une table P de précédence ;
Initialisation arbitraire d' un sommet **v** de l' ensemble **V.**
Pour chaque sommet w dans V faire
    D[{w}, w] = c(v, w)
    P[{w}, w] = v
Pour i de 2 à n faire
    Pour chaque sous-ensemble S de V ouu le nombre de sommet N_S = i faire
        Pour chaque sommet w de S faire
            Pour chaque sommet w2 de S faire
                z = D(S \ {w},u) + c(u, w) ;
                if z < D(S, w) alors
                    D(S, w) z ;
                    P(S, w) u ;
Retourner le chemin lecture renversée des sommets contenus dans P à partir de P[V, v]

V- TEST ET DISCUSSION.

a- Instance de test.

Dans cet exemple, le graphe utilisé est le suivant, où le sommet source est 0 (représenté sous forme d' un tableau de paires (voisin, poids)) :



b- Discussion des résultats.

Si ici on ne trouve que deux chemins hamiltoniens (0 – 2 – 1 – 5 – 3 – 4 et 0 – 2 – 1 – 5 – 4 – 3), il a quand même fallu tester tous les chemins possibles pour arriver à cette conclusion.

c- Analyse de complexité.

La complexité de cet algorithme est donc au moins celle qui consiste à énumérer tous les chemins possibles, soit O((|V|-1)!). C' est un nombre beaucoup trop important pour être calculable en pratique. Par exemple, avec un graphe complet à 30 sommets, cela représente 29!, soit 8.841.761.993.739.701.954.543.616.000.000 chemins à considérer.

CONCLUSION.

En somme, il était question pour nous de définir et présenter une approche de résolution du problème de MTSP problème du Voyageur de Commerce a été étendu au cas où *m* voyageurs de commerce doivent visiter *n* villes, défini comme le Problème de Voyageurs de Commerce Multiples ou *Multiple Traveling Salesman Problem* (MTSP). En effet, celle solution se base sur la solution du TSP particulier et en considerant que ces voyageurs prennent un meme point de depart et avec une hypothése d' une symetrie totale. Mais la résolution serait judicieuse si on essaye selon les documentations d' implementer cette approche tout en prenant en compte certaines hypothéses de pareto , methodes heuristiques qui émanent de la recherche.