Département Informatique
Master Ingénierie Informatique

Rapport sur l'Algorithme du Plus Proche Voisin pour le TSP

Projet Réalisé par
Saber Lahcini
D136132571
PB283702

Sous la direction de
Prof : Lamia Benameur

Année Universitaire : 2024-2025

1 Introduction

Le problème du voyageur de commerce (TSP) est un problème combinatoire bien connu en informatique et en optimisation, qui consiste à trouver le plus court chemin possible pour visiter un ensemble de villes et revenir à la ville de départ. Ce rapport présente une analyse de l'algorithme du plus proche voisin (NN), une heuristique efficace et simple pour aborder ce problème. Nous discuterons de la théorie sous-jacente du TSP, de l'algorithme utilisé, de son implémentation en Python, et de ses performances.

2 Définition du problème

Le TSP peut être défini comme suit : étant donné un ensemble de n villes et les distances entre chaque paire de villes, l'objectif est de déterminer un cycle hamiltonien de coût minimal. En d'autres termes, il s'agit de trouver une séquence de villes à visiter qui minimise la distance totale parcourue, tout en s'assurant que chaque ville est visitée une seule fois et qu'on retourne à la ville de départ. Le TSP est un problème NP-difficile, ce qui signifie qu'il est peu probable qu'une solution optimale soit trouvée rapidement pour de grandes instances.

3 Description de l'algorithme : Plus proche voisin

L'algorithme du plus proche voisin est une heuristique gloutonne, qui construit la solution de manière itérative en sélectionnant à chaque étape la ville non visitée la plus proche de la ville actuelle. Bien qu'il ne garantisse pas une solution optimale, il fournit une approximation efficace dans de nombreux cas.

3.1 Étapes de l'algorithme

1. **Initialisation :** Sélectionner une ville de départ arbitraire.
2. **Itération :**
   (a) Trouver la ville non visitée la plus proche de la ville actuelle.
   (b) Ajouter cette ville à la tournée.
   (c) Marquer la ville comme visitée.
3. **Terminaison :** Répéter l'étape précédente jusqu'à ce que toutes les villes aient été visitées.
4. **Retour au départ :** Ajouter la ville de départ à la fin de la tournée pour compléter le cycle.

3.2 Pseudocode

Voici le pseudocode de l'algorithme du plus proche voisin :

```
fonction PlusProcheVoisinTSP(villes, villeDepart):
  tournee = [villeDepart]
  visitees = {villeDepart}
  villeActuelle = villeDepart
  tant que |visitees| < |villes|:
    villePlusProche = null
    distanceMin = infini
    pour ville dans villes:
      si ville n'est pas dans visitees:
        distance = distance(villeActuelle, ville)
        si distance < distanceMin:
          distanceMin = distance
          villePlusProche = ville
    tournee.ajouter(villePlusProche)
    visitees.ajouter(villePlusProche)
    villeActuelle = villePlusProche
  tournee.ajouter(villeDepart) // Retour au départ
  retourner tournee
```

4 Implémentation du code

L'implémentation en Python de l'algorithme du plus proche voisin repose sur une matrice d'adjacence représentant les distances entre les villes. Chaque ville est indexée, et la matrice contient les distances entre toutes les paires de villes.

[language=Python, caption=Code Python pour le TSP du plus proche voisin, basicstyle=, breaklines=true] def PPV$_T SP(villes, depart) : visite = [False] * len(villes) chemin = [] distance_t otale = 0 current = depart$

for $_i n range(len(villes)) : visite[current] = True chemin.append(current)$

Trouver la ville non visitée la plus proche prochain = -1 min$_d istance = float('inf') for i in range(len(villes)) : if not visite[i] and villes[current][i] < min_d istance : prochain = i min_d istance = villes[current][i]$

if prochain != -1: distance$_t otale += min_d istance current = prochain$

Retour au point de départ chemin.append(depart) distance$_t otale += villes[current][depart]$

return chemin, distance$_t otale$

5 Analyse de la complexité temporelle

La complexité temporelle de l'algorithme du plus proche voisin est de $O(n^2)$, où n est le nombre de villes. À chaque étape, l'algorithme parcourt la matrice des distances pour trouver la ville la plus proche. Puisque l'algorithme effectue une recherche dans une matrice carrée de taille $n \times n$, la complexité est quadratique.

6 Conclusion

L'algorithme du plus proche voisin est une heuristique simple et rapide pour approximativement résoudre le problème du voyageur de commerce. Bien qu'il soit efficace pour des instances de petite taille, il peut produire des solutions loin de l'optimalité pour des problèmes plus complexes. D'autres heuristiques comme l'algorithme génétique ou la recherche locale peuvent être envisagées pour améliorer les résultats dans de tels cas.