TP2 Recuit simulé, algorithmes génétiques et colonies de fourmis appliqués au TSP

Consignes

A l'issu du TP, l'étudiant produit un compte rendu en pdf accompagné des fichiers .py correspondant aux questions posées dans le sujet. L'ensemble est compressé et envoyé dans un fichier NOM_PRENOM.zip. Mettez votre nom et prénom dans le pdf et aussi en commentaire des programmes.

Objectif

Ce TP vise à comparer trois approches pour résoudre le problème du voyageur de commerce connu sous le nom TSP (travelling salesperson problem). Les trois approches sont respectivement le recuit simulé, l'algorithme génétique et les colonies de fournis. Plusieurs variantes du problème du TSP existent. Aussi, le TSP trouve plusieurs applications pratiques comme dans l'agencement, l'ordonnancement et la desserte. Ce TP, se focalise sur la version de base du TSP.

Description

Considérons le problème du voyageur de commerce dans sa version originale. Nous souhaitons calculer le trajet du voyageur qui doit visiter l'ensemble des villes et retourner chez lui. L'objectif du voyageur est de visiter chaque ville une seule fois en parcourant le minimum de kilomètres possibles. Dans ce TP une première solution à base du recuit simulé est proposée (voir le fichier TSPexemple.py).

L'objectif est d'évaluer les performances des trois heuristiques pour résoudre les problèmes de TSP d'une manière générale.

Questions

Après familiarisation avec le code proposé, répondre aux questions suivantes :

1- Visualisation : Le programme proposé génère des instances du problème sous la forme d'une matrice des distances. Afin d'améliorer la visualisation des solutions proposés, procédez de la manière suivante :
   a. Chaque ville à visiter a des coordonnées cartésiennes (X,Y) générées aléatoirement
   b. La matrice des distances entre les villes est calculée en utilisant la distance euclidienne
   c. Afficher les points et la tournée proposée (droite liant chaque paire de points successifs)

2- Faites varier les paramètres du programme de telle sorte à traiter des problèmes de tailles différentes (4 à 20 villes). Faites aussi varier les paramètres du recuit simulé, et représenter graphiquement l'évolution du coût de la solution en fonction du nombre d'itérations.

3- Programmer l'algorithme génétique permettant de résoudre le TSP (Vous pouvez vous inspirer du TP précédent et des éléments présentés dans TSPexemple.py).
   a. Choisir les paramètres du croisement et de la mutation en fonction des résultats observés

4- Programmer l'algorithme de colonies de fourmis adapté au TSP :
   a. Choisir les paramètres de l'algorithme en fonction des résultats observés

5- Comparer les trois approches du point de vue de la qualité des solutions proposées et du temps de calcul. Pour ce faire, générer des instances représentatives (minimum 4) du problème TSP.