**Université de la Manouba**

**Ecole Nationale des Sciences de l'Informatique**



Mini-rapport du Stage d'été II1

-Stage de Programmation-

**Sujet**

|  |
| --- |
| **Simulation graphique de l’algorithme des colonies des fourmis** |

**Réalisé par**

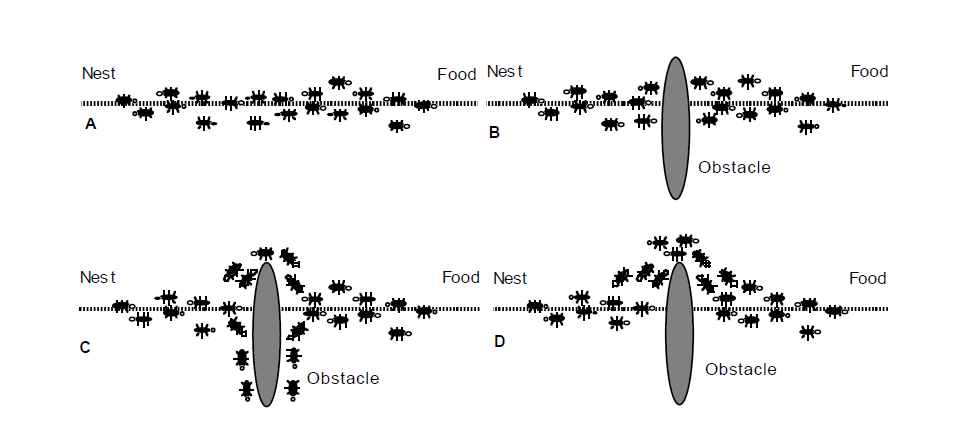
Souilmi Fayed

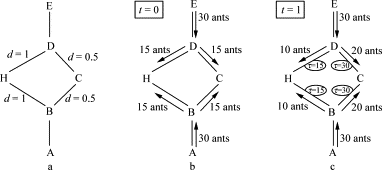
Ben Nasr Asma

Année Universitaire 2020/2021

1. **Présentation générale du sujet**

Le problème du voyageur de commerce (Traveling Salesman Problem : TSP) est un problème combinatoire d'une simplicité trompeuse. On peut le dire très simplement: un vendeur passe son temps à visiter n villes (ou nœuds). Au cours d'une tournée, il visite chaque ville une seule fois et termine là où il a commencé. Inspiré par le comportement de recherche de nourriture des colonies de fourmis, Dorigo et ses collègues ont développé l'optimisation des colonies de fourmis (ACO) qui est appliquée au TSP. L’algorithme de l’optimisation des colonies des fourmis (ACO) modélise le comportement des vraies colonies des fourmis en établissant le plus court chemin entre les sources et les nids. Les fourmis libèrent des phéromones sur le sol en marchant de leur nid à la nourriture, puis en revenant au nid. Les fourmis suivent le chemin contenant la plus grande quantité des phéromones ce qui résultera en un choix du chemin le plus court.





1. **Présentation des fonctionnalités demandées de l'application** 
   1. **Besoins fonctionnels :**

(Présentation de la structuration générale du programme : l’organisation du projet : liste des fichiers formant le projet)

-Le développement d’un algorithme d’intelligence en essaim (intelligence distribuée).

-La modélisation des villes.

-La conception d’une solution pour le problème.

* 1. **Besoins non fonctionnels :**

- La précision

-La visibilité

-L’utilisabilité

1. **Présentation de la conception de la solution proposée** 
   1. **Architecture du programme :**

Le projet comporte

* Un code Python pour la résolution du problème
* Un code Python pour la stylisation de l’interface graphique
* Un dossier contenant des fichiers texte comportant les coordonnées des villes qui vont être parcourus.
  1. **Détails de la solution**

Les classes :

* + Classe fille : Edge
    - Attributs de la classe Edge :
      * a : le point de départ
      * b : le point d’arrivée
      * weight : le poids du côté
      * initial\_pheromone : la quantité de la phéromone initiale
    - Constructeur
  + Classe fille : Ant
    - Attributs de la classe Ant :
      * alpha : paramètre qui contrôle le poids relatif de la phéromone
      * beta : paramètre qui contrôle la visibilité heuristique du côté
      * num\_nodes : le nombre des nœuds (villes)
      * edges : les côtés
    - Constructeur
    - Méthodes de la classe Ant :
      * \_select\_node : Pour cette méthode sera implémentée comme un processus de sélection à la roulette (roulette-wheel-style selection process). On organise les neouds non visités dans une liste, puis pour tous les nœuds non visités on calcule la règle de transition probabiliste:

La somme de ces probabilités est stockée dans la variable roulette\_wheel.

Puis une valeur aléatoire entre 0 et la valeur de roulette\_wheel est prise et stockée dans random\_value. Maintenant, on calcule chaque probabilité de façon cumulative et on la compare à la valeur aléatoire si elle est supérieure à cette valeur aléatoire on retourne le dernier nœud non visité qui obéit à cette condition.

* + - * find\_tour : cette méthode retourne une liste des nœuds visités (le premier nœud est choisi au hasard, les autres nœuds seront choisis selon la méthode \_select\_node).
      * get\_distance : calcule la somme des poids des côtés de la tour.
* Classe mère : SolveTSPUsingACO
  + Constructeur : Dans le constructeur il y a l’association des attributs à savoir : le mode, la taille de la colonie,, le poids du dépôt des phéromones, le nombre de pas à faire,le nombre des nœuds (villes),les labels.

On implémente aussi la création de la matrice des Edges remplie par les poids des côtés, la création des objets Ants, l’initialisation de gloabal\_best\_tour et global\_best\_distance.

* + Méthodes de la classe SolveTSPUsingACO :
    - \_add\_pheromone : augmentation de la quantité des phéromones déposées sur les côtés de la tour par une quantité de avec Q un paramètre pour ajuster la quantité de phéromone déposée, généralement il serait réglé sur 1 et *Ck* la distance totale de la tour.

On obtient donc

* + - \_acs : Cette méthode applique la méthode \_add\_pheremone à tous les côtés de la tour puis met à jour la tour en fonction de la distance, enfin elle met à jour le phéromone globalement en utilisant la formule suivante avec le paramètre d’évaporation des phéromones
    - plot : le traçage du plus court chemin
    - run : l’affichage de la distance optimale et des nœuds du chemin.
* Classe myApp :

Il s’agit d’une classe qui invoque les classes de l’interface graphique

* + Méthodes de myApp :
    - Button1Action : définit l’action du bouton 1
    - Button2Action : définit l’action du bouton 2
* Classe Notepad :
  + Constructeur : fait appel à la super classe Notepad et à évoquer les composants de l’écran qui s’affiche
  + Méthode de Notepad :
    - init\_ui : paramétrage de l’écran qui va s’afficher
    - save\_text : fonction qui sert à enregistrer les modifications
    - open\_text : fonction qui sert à parcourir les fichiers et ouvrir le fichier choisi
    - clear\_text : fonction qui sert à supprimer tout le texte du fichier

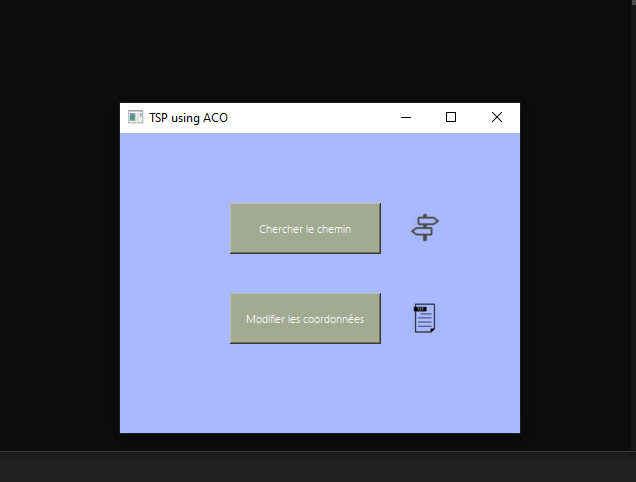
1. **Manuel d’utilisation** 
   1. **Environnement logiciel**

Langage de programmation : Python

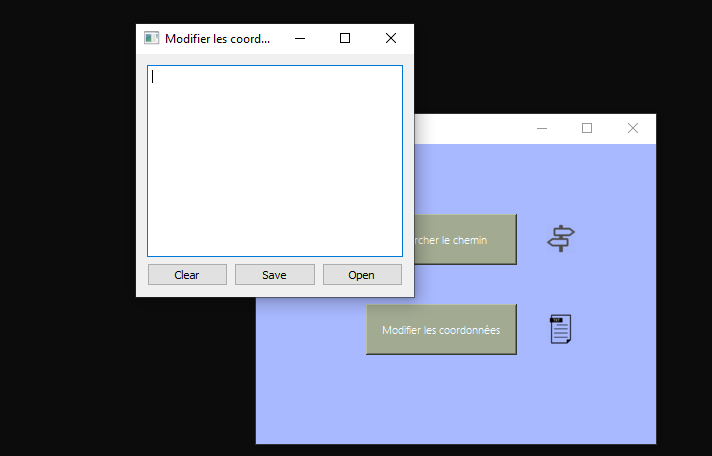
EDI : Qt Designer

Bibliothèques utilisées : math,random,matplotlib, sys, PyQt5, pathlib

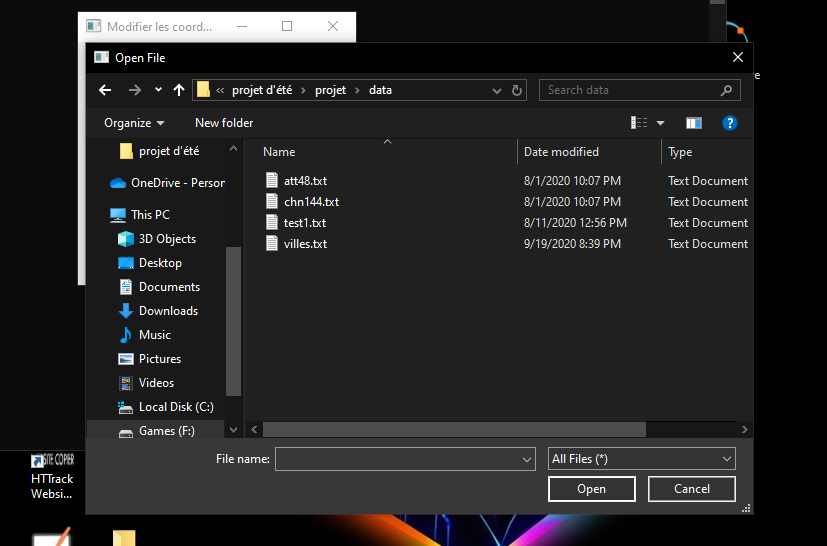
* 1. **Présentation du travail réalisé**



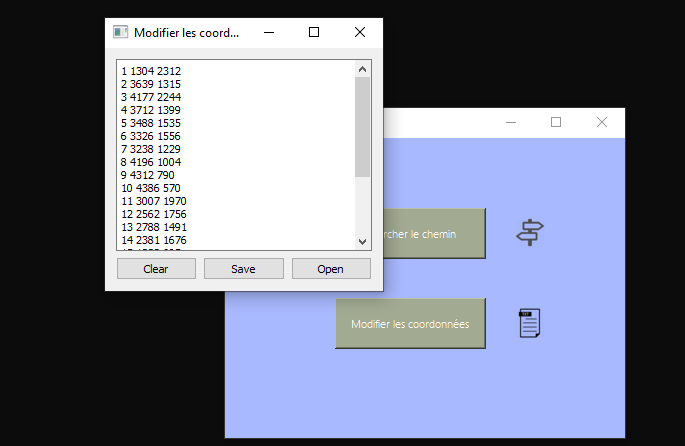
Cliquer sur le bouton « Modifier les coordonnées »



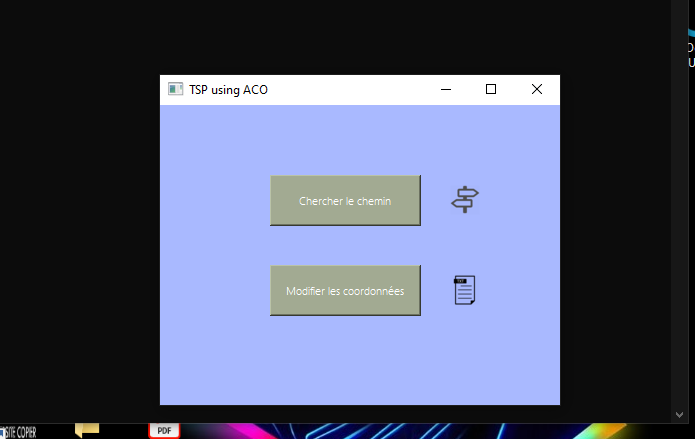
Cliquer sur « Open »



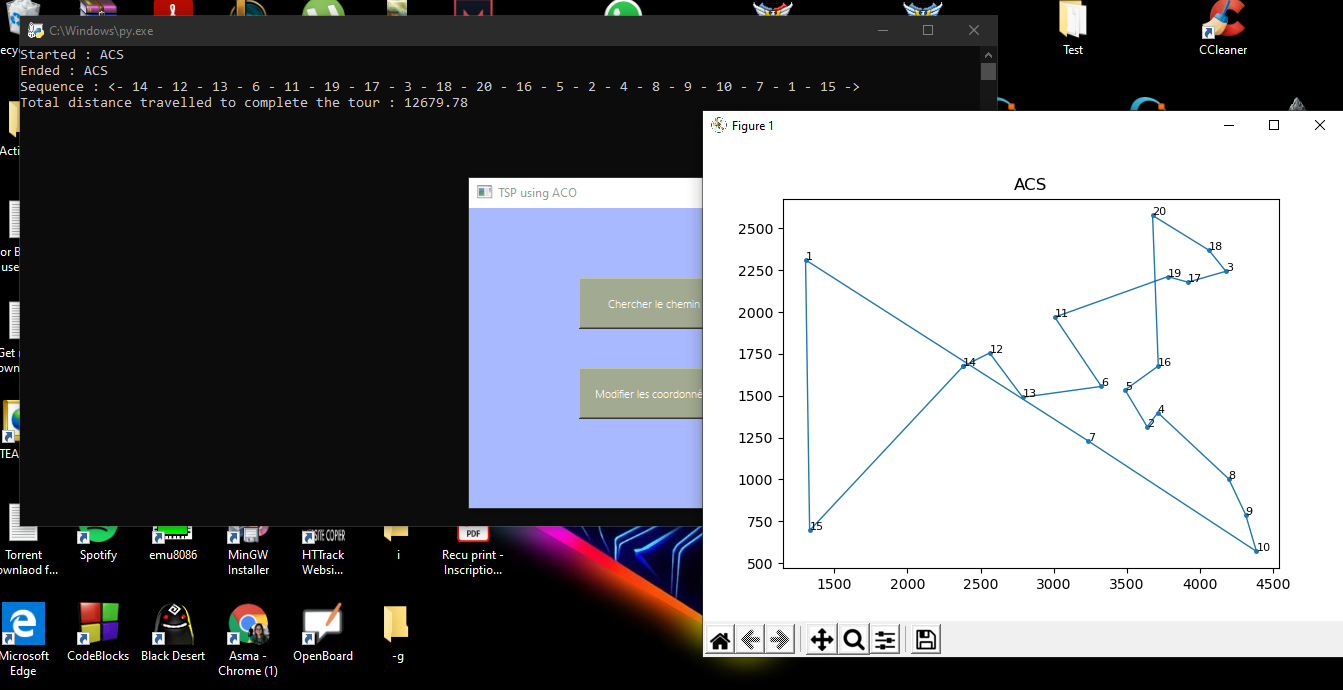
Ouvrir le fichier texte villes



Modifier les coordonnées des villes et Appuyer sur « Save » pour enregistrer les modifications puis quitter le menu de modification



Appuyer sur « Chercher le chemin »



Voir le résultat

1. **Les problèmes rencontrés et leurs solutions**

|  |  |
| --- | --- |
| Problèmes rencontrés | Solutions proposées |
| La modélisation du comportement aléatoire des fourmis | L’algorithme de la sélection de roulette-wheel |
| L’insertion des coordonnées des villes | La lecture à partir d’un fichier txt |
| La stylisation de l’interface graphique | L’utilisation de Qt Designer |

1. **Références** 
   * + Ant colony optimization: Introduction and recent trends Christian Blum, le 20 Juillet 2020
     + Ant colony optimization by Subvesh Raichand, le 18 Juillet 2020

* The Ant System: Optimization by a colony of cooperating agents by Marco Dorigo, Member IEEE, Vittorio Maniezzo , and Alberto Colorni le 1er Août 2020
* Journal of Applied & Computational Mathematics: Solving Traveling Salesmen Problem using Ant Colony Optimization Algorithm by Raghavendra BV le 15 Août 2020
* Solving the Traveling Salesman Problem with Ant Colony Optimization: A Revisit and New Efficient Algorithms Hoang Xuan Huan, Nguyen Linh-Trung, Do Duc Dong, Huu-Tue Huynh le 19 Août 2020
* Multiple Input Multiple Output Detection Using Roulette Wheel Based Ant Colony Optimization Technique B. Rebekka, B. Malarkodi le 20 Août 2020
* A modified Ant Colony Optimization algorithm to solve a dynamic traveling salesman problem: A case study with drones for wildlife surveillance le 3 Septembre 2020
* Python Qt tutorial Documentation Thomas P. Robitaille le 5 Septembre 2020
* Optimisation par colonies de fourmis COSTANZO Andrea LUONG Thé Van MARILL Guillaume le 12 Septembre 2020
* Qt Designer Manual le 15 Septembre 2020