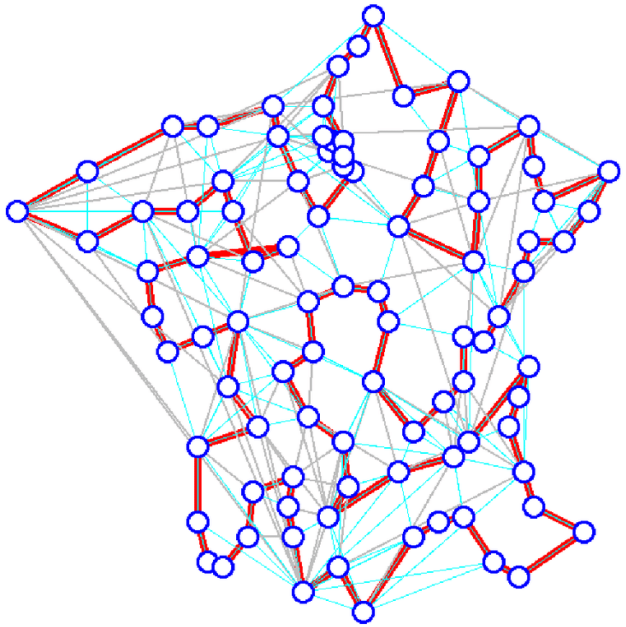
Problème de voyageur de commerce.

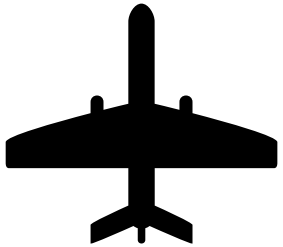


Réalisé par: ***Saif henzili***
Iheb oueslati
Maram ben ameur
Aziz Bannour

esprit
Se former autrement

Les algorithmes proposés:

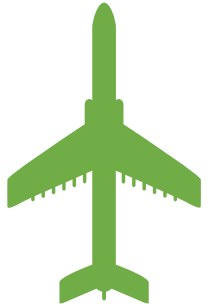
L'algorithme du plus proche voisin (Méta-Heuristiques)



La méthode de recuit simulé (Heuristiques)

L'algorithme de Génétique(Meta-heuristique)

Colonie de fourmis(ACO)



1.Complexité:

Les algorithmes proposés:

L'algorithme du plus proche voisin (Complexité):

Complexité :
- chercher permis les villes à visiter, laquelle est la plus proche de la dernière ville du cycle.
-Comme on parcourt entièrement la matrice des distances des villes, on obtient une complexité $O(n^2)$.
-grâce à l'utilisation de vecteurs, on ne revisite pas une ville déjà insérée, ce qui diminue un peu la complexité de cet algorithme

Facile à implémenter. Complexité en O(n 2).
-En moyenne, NN donne des tours de coût 1.26 fois plus élevé que la valeur optimale, noté NN/OPT= 1.26.

- Pire comportement connu : NN/OPT $\in$(log n/ 3 log log n) (environ 3 pour n = 100, 4 pour n = 10, 000, etc.)

- Borne garantie sur la solution optimale (Rosenkrantz, Stearns, Lewis, 1977) :
Si les distances sont positives et respectent l'inégalité triangulaire, alors
NN/OPT ≤ 1 2 [log2 n] + 1 /2
(4.5 pour n = 100, 7.5 pour n = 10, 000, etc.)

Les algorithmes proposés:

Colonie de fourmis(ACO) (Complexité):

Pour calculer la complexité on ***divise l'algorithme en 5 sections:***

1/ Initialisation
2/ Un cycle de l'algorithme
3/ Fin du cycle et calcul des dépots de phéromone.
4/ Evaporation des phéromones.
5/ Boucle de l'algorithme.

1/ On a une complexité O (|L|+m)=O(n^2+ m) , puisque l'on a supposé une interconnexion totale entre les ville. n= nombre de ville / m= nombre de fourmis
2. La complexité est O(n^2 X m) , puisque les opérations de calcul de la ville suivante nécessite un balayage de l'intégralité des villes.
3/ La complexité est O(|L| + m X |L|)= O(m X|L|)= O(n^2 X m)
4/ La complexité est O(|L|)=O(n^2).
5/ Le test de stagnation est de complexité O(n x m) (on doit comparer les tours de m fourmis, chque tour ayant une longeur de n éléments).

- La complexité générale de l'algorithme devient donc:
O(n^2 + m + NCmax X n^2 X m)

Soit: AS complexity= O(NCmax x n^2 x m)=O(n^2) avec Nmax= le nbre max d'iteration
et m le nbre de fourmis.

Les algorithmes proposés:

L'algorithme génétique (Complexité):

Complexité** via une **méthode de résolution avec algorithme génétiques:

Détail de calcul :

$O(\text{Algo Gene}) = O(n^2) + O(\text{Muter}) + O(\text{Croiser}) + O(\text{SelectionNaturelle})$

$= O(n^2) + O(n) + O(2n) + O(n)$

$= O(n^2)$

NB VILLES	NB POSSIBILITES	TEMPS DE CALCUL
5	25	25 µs
10	100	100 µs
15	225	225 µs
20	400	400 µs

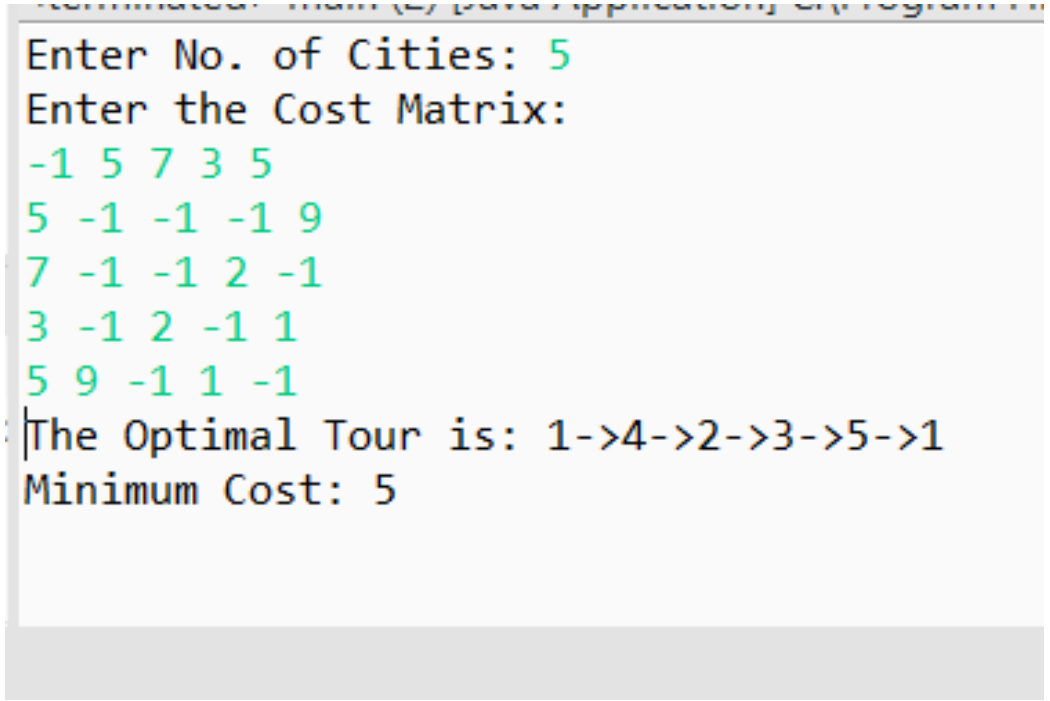
Les algorithmes proposés:

La méthode de recuit simulé (Complexité):

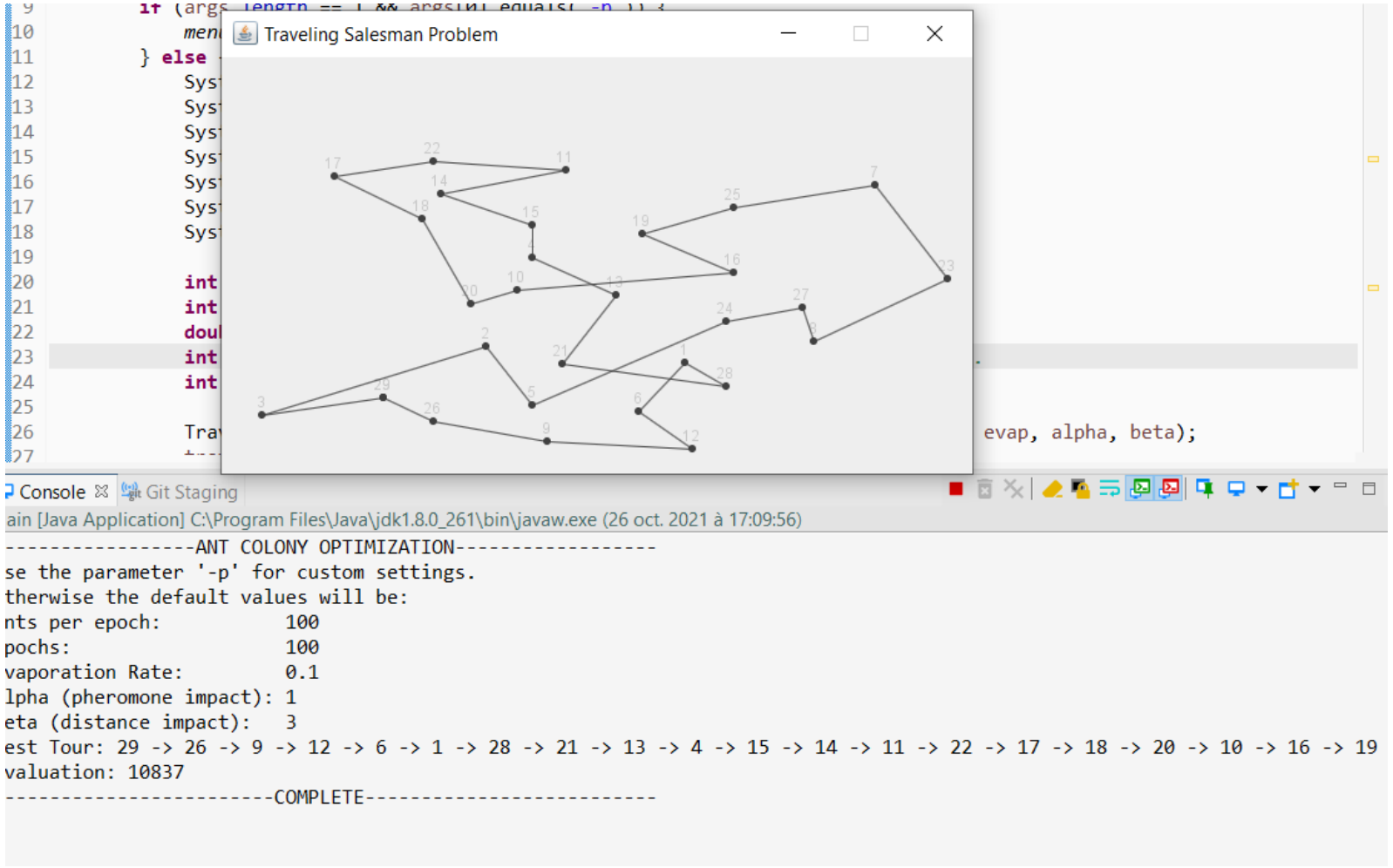
Soit E l'ensemble de n villes numérotées de 0 à n-1. J'appelle $\delta(vn+1,vi)$ la distance connue entre deux villes voisines.
e voyageur de commerce en faisant son tour sera donc égale à $d=\sum i=0\ a\ n-1\delta(vn+1,vi)$
Un petit exemple :
Raisonnons sur un ensemble E avec peu d'élements, par exemple 3 villes, numérotées 1, 2 et 3
Les chemins possibles, sachant qu'il faut revenir au point de départ, sont : (1,2,3,1), (1,3,2,1), (2,3,1,2), (2,1,3,2), (3,2,1,3) et (3,1,2,3). Il y a 6 combinaisons de chemins possibles. En fait, le nombre de possibilités s'élève à n! (n factorielle). Pour n=3, cela fait 6 possibilités. Pour 10 villes, cela fait 3,6 millions de combinaisons possibles et pour 20 villes $2,4.10^{18}$ combinaisons possibles

2. Benchmarking entre les différents algorithmes

1.Exécution du algorithme du plus proche voisin :



2. Exécution du algorithme Colonie de fourmis



3.Exécution du algorithme de recuit simulé

```
76          TourManager.addCity(city23);
77          City city24 = new City("24",5, 5);
78          TourManager.addCity(city24);
79
80
```

Console Git Staging

terminated> SimulatedAnnealing [Java Application] C:\Program Files\Java\jdk1.8.0_261\bin\javaw.exe (26 oct. 2021 à 17:38:26)

```
otal distance of initial solution: 64
our: 8 -> 1 -> 18 -> 14 -> 6 -> 4 -> 23 -> 0 -> 16 -> 9 -> 3 -> 10 -> 11 -> 22 -> 21 -> 13 -> 20 -> 24 -> 15 -> 19 -> 7 -> 5 -> 17 -> 2 -> 12
inal solution distance: 40
our: 24 -> 14 -> 7 -> 11 -> 12 -> 17 -> 21 -> 16 -> 22 -> 20 -> 18 -> 23 -> 15 -> 2 -> 0 -> 5 -> 6 -> 8 -> 4 -> 9 -> 3 -> 1 -> 10 -> 13 -> 19
```

Writable Smart Insert 80 : 9 : 2747 239M of 389M

4. Exécution du méthode génétique

Représentation d'une solution(Génétique) :

Comme nous l'avons déjà dit le voyageur de commerce doit revenir à son point de départ et passer par toutes les villes une fois et une seule. Nous avons donc codé une solution par une structure de données comptant autant d'éléments qu'il y a de villes, c'est à dire une permutation. Chaque ville y apparaît une et une seule fois. Il est alors évident que selon la ville de départ que l'on choisit on peut avoir plusieurs représentations différentes du même parcours

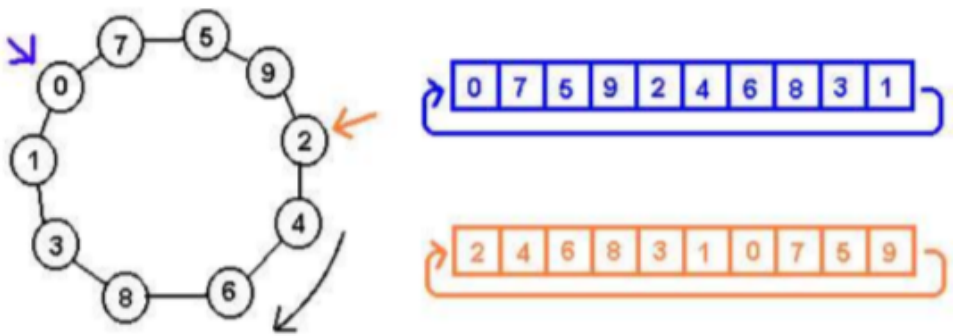


Figure 16 : codage d'une solution (ensemble de villes) dans un tableau .

4. Exécution du méthode génétique

```
Starting grade: 1.71 / 110
Current grade: 17.00 / 110 (255 generation)
Current grade: 23.21 / 110 (511 generation)
Current grade: 36.98 / 110 (767 generation)
Current grade: 43.91 / 110 (1023 generation)
Current grade: 50.95 / 110 (1279 generation)
Current grade: 57.00 / 110 (1535 generation)
Current grade: 61.95 / 110 (1791 generation)
Current grade: 65.90 / 110 (2047 generation)
Current grade: 68.76 / 110 (2303 generation)
Current grade: 77.87 / 110 (2559 generation)
Current grade: 82.89 / 110 (2815 generation)
Current grade: 85.91 / 110 (3071 generation)
Current grade: 86.81 / 110 (3327 generation)
Current grade: 86.80 / 110 (3583 generation)
Current grade: 88.94 / 110 (3839 generation)
Current grade: 89.95 / 110 (4095 generation)
Current grade: 90.92 / 110 (4351 generation)
Current grade: 93.83 / 110 (4607 generation)
Current grade: 94.83 / 110 (4863 generation)
Current grade: 95.95 / 110 (5119 generation)
Current grade: 97.97 / 110 (5375 generation)
Current grade: 98.80 / 110 (5631 generation)
Current grade: 99.00 / 110 (5887 generation)
Current grade: 101.76 / 110 (6143 generation)
Current grade: 101.84 / 110 (6399 generation)
Current grade: 102.90 / 110 (6655 generation)
Current grade: 102.88 / 110 (6911 generation)
Current grade: 103.75 / 110 (7167 generation)
Current grade: 103.93 / 110 (7423 generation)
Current grade: 103.95 / 110 (7679 generation)
```

```
Current grade: 105.90 / 110 (7935 generation)
Current grade: 105.78 / 110 (8191 generation)
Current grade: 105.85 / 110 (8447 generation)
Current grade: 106.88 / 110 (8703 generation)
Current grade: 107.00 / 110 (8959 generation)
Current grade: 106.85 / 110 (9215 generation)
Current grade: 106.85 / 110 (9471 generation)
Current grade: 107.00 / 110 (9727 generation)
Current grade: 107.95 / 110 (9983 generation)
Current grade: 107.85 / 110 (10239 generation)
Current grade: 107.85 / 110 (10495 generation)
Current grade: 107.83 / 110 (10751 generation)
Current grade: 108.97 / 110 (11007 generation)
Current grade: 108.92 / 110 (11263 generation)
Current grade: 108.94 / 110 (11519 generation)
Current grade: 108.84 / 110 (11775 generation)
Final grade: 109.02 / 110
Solution found (7 times) after 11962 generations.
```

Merci pour votre attention

esprit
Se former autrement