|  |
| --- |
| Compte-rendu du projet d’info |
| PROJET : LE TOUR DU MONDE LE MOINS CHER |
| Réalisé par: Robin Moussu & Jingbo Su |

2013-2014

# Plan du compte-rendu:

1- Introduction

2- Spécifications

2.1 Données : description des structures de données

2.2 Modules : rôles de chaque module (couple de fichiers .c et .h)

2.3 Fonctions : prototype et rôle des fonctions essentielles

2.4 Tests : quels sont les tests prévus

2.5 Répartition du travail et planning prévu : qui fait quoi et quand ?

3- Implantation

3.1 État du logiciel : ce qui fonctionne, ce qui ne fonctionne pas

3.2 Tests effectués

3.3 Exemple d'exécution

3.4 Les optimisations et les extensions réalisées

4- Suivi

4.1 Problèmes rencontrés

4.2 Planning effectif

4.3 Qu'avons-nous appris et que faudrait-il de plus?

4.4 Suggestion ameliorations du projet

5- Conclusion

# Introduction

Le but de faire ce projet informatique est d’appliquer la connaissance acquise pendant cette année et se familiariser avec les instructions.

Donc pour notre sujet distribué, on concerne sur le problème de faire le tour du monde le moins cher. C’est en fait le problème du voyageur de commerce qui doit visiter N villes en passant par chaque ville exactement une fois. Il commence par une ville quelconque et termine en retournant à la ville de départ. En connaissant le coût (temps, distance, ou autre chose...) des trajets possibles entre ces N villes, on doit trouver un circuit pour minimiser le coût du parcours.

Parmi les méthodes de résolution du TSP, on s’intéresse ici aux algorithmes de colonies de fourmis. Cette méthode est inspirée par le comportement de fourmis réelles dont l’action conjointe amène à trouver un chemin optimal entre deux points par la quantité de phéromone déposé sur le chemin.

Dans ce rapport, on va vous spécifier tout d’abord le structure de données, le rôle de chaque module, le prototype et rôle des fonctions essentielles, ainsi les tests prévus. On vous précise aussi la répartition des tâches entre chaque membre et l’emploi du temps d’effectuer le projet.

Ensuite, on vous indique l’état du logiciel en faisant des tests. On vous donne aussi l’exemple d’exécution. On vous introduit aussi les optimisations et les extensions réalisées.

En plus, comme tous les projets, on va raconter les problèmes rencontrés et des solutions proposées. On vous dira le planning effectif, ce qu’on a appris de ce projet et qu’est-ce qu’il faut faire en plus. On propose aussi des suggestions d’amélioration du projet.

A la fin, on vous met une petite conclusion sur ce projet en une vue globale.

# Spécifications

Avant de spécifier chaque terme, on décrit un peu en global la solution qu’on va appliquer.

Contrairement aux fourmis réelles, les fourmis virtuelles ont une mémoire qui leur permet de stocker le chemin parcouru à chaque cycle. Cette mémoire est « vide » au début de chaque cycle. Les fourmis virtuelles déposent une quantité d'information (phéromones) qui est proportionnelle à la qualité de la solution trouvée.

Elles ne mettent à jour les phéromones qu'une fois leur solution (ou chemin) complètement construite et évaluée, à la fin d'un cycle. L'évaporation est réalisée à chaque fin de cycle, juste avant le dépôt des phéromones par les fourmis.

Lorsqu'elles sont sur une ville, les fourmis doivent décider sur quelle ville adjacente se déplacer. Alors que les fourmis réelles semblent utiliser le hasard et les phéromones, les fourmis virtuelles prennent en compte les phéromones, les villes déjà visitées et la « visibilité » des villes disponibles. La visibilité est définie comme l'inverse de la distance entre 2 villes. Au cours d'un cycle, une fourmi virtuelle ne repasse jamais par une ville déjà visitée.

## 2.1 Données : description des structures de données

2.1.1 Fourmis

Pour implanter cet algorithme, on va explicitement représenter une fourmi par une structure qui contiendra:

• solution : ensemble des arcs formant la solution de la fourmi. Pour stocker l'ensemble, on choisira d’utiliser une file d’arcs. Ce choix permettra de représenter non seulement la solution de la fourmi mais aussi la liste tabu des villes déjà visitées.

• ville\_départ et ville\_courante: indice des sommets de départ et courant de la fourmi.

2.1.2 Graphe

Il existe plusieurs modes de représentation d'un graphe afin de gérer les liens entre sommets. Nous utiliserons une liste d'adjacence des voisins d'un sommet : c'est la liste des arcs sortant de ce sommet. Cette liste est stockée dans le champ voisins d'un sommet. Un arc est un quadruplet « sommet de départ », « sommet d’arrivée », « distance entre villes » et « quantité de phéromones ». Pour représenter une arête, on utilisera deux arcs orientés, un dans chaque sens.

Le graphe sera donc défini par un tableau de sommets. Chaque sommet contient ses propres caractéristiques ainsi que des informations utiles à l'algorithme :

• numéro : le numéro du sommet

• nom : le nom du sommet

• x, y : positions du sommet (pour la représentation graphique du graphe)

• voisins : la liste d'adjacence, liste de pointeurs vers les arcs sortant de ce sommet