Projet : Voyageur de commerce

Alexandre Talon

3 novembre 2012

Table des matières

Introduction

On a ici réalisé un projet de programmation dans le langage C, qui a duré 7 semaines. Ce projet a été en très grande partie réalisée en monôme par Alexandre Talon.

On étudie un problème d'optimisation. En effet, le projet porte sur le problème du voyageur de commerce : on veut visiter un certain nombre de villes, dans le temps le plus court possible, et revenir à son point de départ. D'un point de vue plus formel, on peut voir les villes comme étant les sommets d'un graphe, la distance entre deux villes le poids de l'arête reliant les sommets correspondant, le problème étant de trouver un cycle hamiltonien de longueur minimale.

Il s'agit d'un problème NP-complet, c'est-à-dire qu'on ne dispose pas d'algorithme résolvant exactement le problème en un tems polynomial. Ainsi, l'objectif ici est de programmer un algorithme permettant de trouver une tournée passant par toutes les villes imposées par l'utilisateur, le tout en un temps raisonnable.

Pour y parvenir, il est nécessaire de s'occuper d'abord des structures de données à utiliser : quelles sont elles, à quoi serviront elles ici, quelles sont leurs avantages. Ensuite, nous verrons l'algorithme à proprement parler, découpé en fait en deux algorithmes. Enfin, nous traiterons l'aspect pratique et l'utilisation de notre programme.

1 Structures de données

On utilise dans ce projet deux structures non triviales : d'une part des tas, d'autre part des arbres AVL, qui appartiennent à la classe des arbres binaires de recherche.

1.1 Tas

Un tas est un arbre binaire complet vérifiant la propriété suivante :

1.2 Arbres AVL

Les arbres AVL appartiennent à la classe des arbres binaires de recherche. Ils présentent en outre la particularité d'être automatiquement équilibrés, c'est-à-dire :

Définition (Arbre équilibré)**.** *Un arbre équilibré est un arbre dans lequel la hauteur des deux sous-arbres d'un noeud diffère d'au plus un.*

Pour pouvoir définir plus aisément la notion de "différence de hauteur de sous-arbre", on va définir le facteur d'équilibrage :

Définition (Facteur d'équilibrage d'un noeud)**.** *Le facteur d'équilibrage d'un noeud est la différence entre la hauteur de son sous-arbre droit et celle de son sous-arbre gauche.*

Ainsi un arbre équilibré est simplement un arbre dont le facteur d'équilibrage de la racine est compris entre -1 et 1.

Les arbres AVL s'utilisent comme les arbres binaires de recherche, mais on peut également effectuer sur eux les opérations de rééquilibrage. Pour rééquilibrer un arbre déséquilibré, on effectue ce qu'on appelle des rotations. Il convient tout d'abord de remarquer qu'un arbre peut être déséquilibré de deux façons : il peut être "left heavy", c'est-à-dire que sa racine possède un facteur d'équilibrage plus petit que -2, ou "right heavy", c'est-à-dire que sa racine possède un facteur d'équilibrage plus grand que 2. Nous allons commencer par voir comment résoudre ces problèmes par des rotations simples, puis remarquer qu'elles ne suffisent pas dans certains cas où il est obligatoire d'utiliser des rotations doubles.