Projet de programation avancée
S6

NEUS Maxence

May 30, 2021

Contents

1 Introduction

Dans le cadre du module de programmation avancée, nous avons eu à plusierus reprises l'occasion de travailler avec la notion de structures de donnée en C. Lors de ce projet, nous avons eu à mettre à profit les connaissances acquisent à ce sujet au cours du module pour traiter le plus efficacement possible un enseble de données concernant des vols entre un grand nombre d'aéroports aux Etats-Unis. Il nous est proposé de réaliser un certain nombre de fonctions pour obtenir des informations intéressantes sur les vols présentés.
La mise en oeuvre de ce projet peut être divisée en trois parties:

- Lecture des données et gestion des structures.
- Algorithmes permettant d'implémenter les différtentes fonctions proposées dans le sujet.
- Mise en place d'une interface en ligne de commande pour acceder aux données.

2 Lecture des données

2.1 Structures de données

Tout d'abord, nous allons nous pencher sur les choix qui ont été fait concernant le stockage des données.

2.1.1 Aéroports

Pour stocker les informations sur les différents aéroports, l'approche evidente est d'utiliser une structure de type Table de Hashage.
En effet la présence des $IATA\_code$,nous offre un choix simple de clé de hashage car ils sont unique à chaque aéroport.
Cette approche nous permet d'acceder directement à toutes les informations relatives à un aéroport en aillant simplement son $IATA\_code$.
Le choix de la fonction de hashage est détaillé avec celui des companies en 2.1.3.

2.1.2 Companies

Pour les mêmes raisons que pour les aéroports, la présence de $IATA\_code$ nous invite à utiliser ici une table de hashage pour stocker les données relatives aux companies. Le choix de la fonction de hashage est ici aussi détaillé en 2.1.3.

2.1.3 Fonctions de hashage

Se pose alors la question de la fonction de hashage, pour résoudre ce problème, quelques recherches ont menés à la découverte d'un projet appelé "$perfect - hash$" (projet github par ilanschnell). Cet outil consiste en un script python qui prends un fichier de type csv et génere à partir d'une colone une fonction de hashage "parfaite" (c'est à dire qui est parfaitement bijective). Ce script a été ici utilisé pour générer les fonctions de hashage pour les aéroports et pour les companies.

Une limite de ce choix de fonction de hashage est que le code ne pourra traiter que les aéroports et companies qui sont mises à disposition ici. Mais les ajout de companies ou la construction de nouveaux aéroports étants aprioris peu fréquent, ce choix semble raisonnable. Un problème majeur de cette aproche reste lors d'une éventuelle extension du système à l'exterieur des USA, il faudrait regénerer de nouvelles fonctions de hashage.

2.1.4 Vols

Un certain nombre de fonctions utilisent la date des vols comme argument, il semble donc utile de rendre l'acces aux vols à un jour précis le plus rapide possible. Pour ce faire, le choix a été fait ici d'utiliser encore une fois une table de hashage dont la clé d'acces est cette fois-ci la date du vol. Avec cette approche, la structure consistera en n listes de vols qui ont eu lieu le même jour. Le nombre de vols par jour etant variable, et le parcours de chaque liste fréquent, nous choisirons ici de représenter ces listes de vols par des listes contigües de vols.
La structure globale de stockage des vols sera donc une Table de hashage de ces listes contigües où l'indice dans la table correspond à la date à laquelle les vols de la liste ont eu lieu.
Pour déterminer cet indice, la fonction de hashage suivante :

$$f(m, j) = (1 - m) * 31 + j - 1 \tag{1}$$

permets d'obtenir des indices entre 0 pour le 1^{er} janvier et 371 pour le 31 decembre. soit 372 valeurs ce qui correspond à seulement 6 jours de plus qu'une année bissextile, La consommation mémoire de cette solution est donc plutôt optimale tant que la longueure maximale des listes contigües reste proche du nombre réel de vols dans la journée.

2.2 Lecture