Mini-projet
d'algorithmique avancée

ENSSAT SNUM 2

2019-2020

Présentation du problème

On s'intéresse à différents algorithmes que pourrait utiliser un commerçant pour rendre la monnaie avec des pièces. Son problème est d'arriver exactement à une somme N donnée en choisissant dans la caisse un multi-ensemble de pièces dont chacune possède une valeur fixée. Par exemple, dans le système numéraire de l'euro, si le client effectue un achat de 8 € 10 et donne 10 €, le problème consiste pour le commerçant à composer un multi-ensemble de pièces qui totalise 1 € 90. Il y a un grand nombre de solutions, parmi lesquelles :

- 1 pièce de 1 €, 1 pièce de 50 ¢, 2 pièces de 20 ¢
- 2 pièces de 50 ¢, 4 pièces de 20 ¢, 2 pièces de 5 ¢
- 19 pièces de 10 ¢
- ...

Pour étudier le problème de manière générale, on appelle C l'ensemble des pièces de monnaie (il y en a n différentes) et on va supposer que le commerçant dispose d'un nombre illimité de chacune d'entre elles. On note $C = \{c_1, \ldots, c_n\}$, la pièce c_i ayant pour valeur d_i. En France, on a un ensemble C de cardinalité 8, avec les valeurs :

$$d_1 = 2\,€,\ d_2 = 1\,€,\ d_3 = 50\,¢,\ d_4 = 20\,¢,\ d_5 = 10\,¢,\ d_6 = 5\,¢,\ d_7 = 2\,¢,\ d_8 = 1\,¢.$$

Pour reprendre l'exemple précédent, la première solution peut se noter par le multi-ensemble : $\{c_2, c_3, c_4, c_4\}$. Pour achever de définir le problème, on va supposer que le commerçant cherche à redonner le moins de pièces possibles. On peut donc énoncer le problème à résoudre « rendre la monnaie de manière optimale (RLMMO) » comme suit. On se donne un ensemble C tel qu'à chaque élément c_i de C est associée une valeur d_i qui est un nombre entier strictement positif. Soit un nombre entier strictement positif N, trouver un multi-ensemble S de cardinalité minimale composé d'éléments de C tel que la somme des valeurs des éléments de S vaut exactement N.

A. Essais successifs

On demande de définir les différents éléments intervenant dans ce cadre, en particulier, qu'est-ce qu'un vecteur représentant un candidat (taille, contenu) et quels sont les constituants de l'algorithme générique (*satisfaisant*, *enregistrer*, *soltrouvée*, *défaire*) ?

Il conviendra également de s'interroger sur les conditions d'élagage (incluant des estimations sur le futur) pertinentes pour ce problème. On mettra en évidence (au moins) une condition d'élagage dont on montrera l'intérêt en pratique. On donnera également un ordre de complexité maximal en nombre d'appels récursifs de la procédure proposée.

On demande de fournir le listing d'un programme commenté résolvant le problème posé ainsi que les résultats obtenus pour différents exemples suffisamment différents.

B. Programmation dynamique

On appelle $NBP(i,\ j)$ le nombre minimal de pièces nécessaires pour payer la somme j en ne s'autorisant que le sous-ensemble des pièces $\{c_1,\ \ldots,\ c_i\}$. Donner la formule de récurrence complète de calcul de $NBP(i,\ j)$.

Préciser ensuite la structure tabulaire utilisée et l'évolution du calcul permettant de la remplir afin de calculer le nombre de chacune des pièces utilisées pour rendre de façon optimale la somme N à l'aide du système $C = c_1,\ \ldots,\ c_n$. Préciser l'ordre de complexité de cet algorithme.

On demande de fournir le listing d'un programme commenté résolvant le problème posé ainsi que les résultats obtenus pour les mêmes exemples qu'en partie A.

C. Algorithme glouton

La méthode employée en général par un commerçant peut se décrire ainsi : utiliser les pièces par valeur décroissante, en prenant le plus possible de chacune d'elles. Dans l'exemple introductif, cet algorithme produit la première solution $\{c_2,\ c_3,\ c_4,\ c_4\}$.

Montrer que cet algorithme ne résout pas le problème RLMMO quand $C = \{c_1,\ c_2,\ c_3\}$, avec $d_1 = 6$, $d_2 = 4$, $d_3 = 1$ et $N = 8$.

Vérifier expérimentalement que cet algorithme résout le problème RLMMO avec le système de l'euro. Avancer une explication (on ne demande pas de démonstration formelle) de cet état de fait. Que penser du système $C' = \{50,\ 30,\ 10,\ 5,\ 3,\ 1\}$? Quel est la classe de complexité de cet algorithme ?

D. Questions complémentaires

1. Donner votre opinion sur l'effort de conception des algorithmes des parties A et B.

2. Serait-il possible de résoudre le problème RLMMO à l'aide d'une solution de type « diviser pour régner » ? (justifier votre réponse)

3. Au vu de ce qui précède, faire une recommandation argumentée pour résoudre le problème RLMMO.

Livrables attendus

Le mini-projet est effectué en binôme. Deux livrables par binôme sont attendus à l'issue du mini-projet :

1. Compte-rendu du mini-projet. Le compte-rendu contiendra *a minima* : une page de garde, une introduction, les réponses à toutes les questions de l'énoncé incluant les explications claires des solutions développées, les algorithmes écrits en **pseudo-langage** et **commentés**, une étude de la complexité de chaque algorithme, les jeux d'essai motivés (complets, avec étude expérimentale permettant d'exhiber empiriquement les gains découlant de la mise en œuvre des critères d'élagage choisis), une conclusion incluant un véritable bilan du projet (rappel et comparaison des solutions proposées, perspectives), les listings commentés en annexe. Le rapport hors annexes ne devra pas dépasser 15 pages.
ATTENTION : Le compte-rendu hors annexes doit pouvoir être lu indépendamment du code (contenir toutes les informations nécessaires à la compréhension et l'évaluation des solutions proposées).

2. Sources du mini-projet. Les fichiers sources commentés de votre code devront être fournis, incluant un fichier README (le fichier README expliquera comment installer et utiliser votre/vos programme(s)). L'utilisation du langage de développement Java vous est recommandé pour l'implantation des algorithmes (mais ce n'est pas une obligation).

Les livrables devront m'être envoyés par mail dans un mail intitulé *"[SNUM2 Algorithmique Avancée] Mini-projet du binome B_1 / B_2"* où B_1 et B_2 sont à remplacer par les noms des binômes.