L3-INFO Algorithmique et Modélisation

Examen session 1 – 19 avril 2023 – durée 3 h

Sont interdits : les documents, les ordinateurs, les téléphones (incluant smartphone, tablettes,... tout ce qui contient un dispositif électronique).
Seuls les dictionnaires papier pour les personnes de langue étrangère sont autorisés.
En cas de doutes sur l'énoncé, préciser les choix que vous faites sur votre copie et continuer.
Il sera tenu compte de la qualité de la rédaction et de la clarté de la présentation (2 pts).
Le barème indicatif : Exercice I : 5 points ; Exercice II : 5 points ; Problème III : 8 points
Les exercices et le problème peuvent être traités indépendamment.
Les durées sont indicatives, penser à se relire.

Exercice : Des chiffres... *($\sim$ 45 mn)*

En cryptographie, des énigmes classiques associent des chiffres à des lettres.

Extrait de Jeux et Stratégie n 26

Voici deux cryptarithmes très « sympathiques », que nous devons à un lecteur François Brassaud, de Talant (qui gagne un abonnement d'un an à J. & S.). Tous deux à solution unique.

a. Commençons par le plus simple :

$$\begin{array}{r} \text{T I G R E} \\ +\ \text{L I O N N E} \\ \hline =\ \text{T I G R O N} \end{array}$$

b. Pas aussi facile qu'on pourrait le croire :

$$\begin{array}{r} \text{O N Z E} \\ \times\ \ \text{D I X} \\ \hline =\ \text{C E N T D I X} \end{array}$$

Brièvement le rappel du principe :
chaque lettre représente un chiffre toujours le même ;
un chiffre est toujours représenté par la même lettre.
Aucun nombre ne commence par 0 (zéro).

Il ne vous est pas demandé de résoudre ce problème à la main pendant l'examen (activité personnelle à faire à la maison).

Autre exemple :

$$\begin{array}{rcccc} & \text{N} & \text{E} & \text{U} & \text{F} \\ + & \text{D} & \text{E} & \text{U} & \text{X} \\ \hline & \text{O} & \text{N} & \text{Z} & \text{E} \end{array}$$

admet comme solution (il y en a plusieurs), huit chiffres sont utilisés $5283 + 1289 = 6572$

0	1	2	3	4	5	6	7	8	9
/	D	E	F	/	N	O	Z	U	X

On souhaite résoudre ce problème en force brute, pour cela on énumère toutes les possibilités d'association entre les chiffres et les lettres du problème.

I.1. Donner, en justifiant votre réponse, le nombre de possibilités à tester dans les trois exemples ci-dessus.

I.2. Combien y-en-a-t'il dans le cas général de 10 chiffres à associer ? Donner un ordre de grandeur de ce nombre. Justifier s'il est raisonnable ou non d'énumérer toutes les possibilités en utilisant un ordinateur actuel ?

I.3. Algorithme d'énumération de toutes les possibilités.

(a) Écrire un algorithme récursif d'énumération de toutes les possibilités. Spécifier les paramètres d'appel et la structure de donnée utilisée.

(b) Prouver la correction de cet algorithme et évaluer sa complexité

Exercice : Emploi du temps... *(∼ 45 mn)*

Un étudiant souhaitant planifier sa formation doit choisir un ensemble de cours compatibles dans son emploi du temps. L'ensemble des n cours proposés définissent des plages horaires que l'on note $[d_i, f_i]$ pour le cours i avec d_i la date et l'heure de début du cours i et f_i la date et l'heure de fin (on a $d_i < f_i$). Évidemment un étudiant ne peut pas suivre 2 cours dont les plages se chevauchent.

Supposons, dans un premier temps, que l'étudiant souhaite suivre un maximum de cours, il propose un algorithme glouton qui consiste à classer les cours par date de fin croissante et choisir les cours dans cet ordre en respectant la contrainte que 2 cours ne se chevauchent pas.

II.1. Donner rapidement les éléments de preuve qui montrent que cet algorithme glouton produit une solution optimale.

En fait, à chaque cours on associe un nombre de crédits ECTS, on note c_i le nombre d'ECTS pour le cours i, l'objectif est alors de maximiser le nombre d'ECTS. On suppose que les cours ont été triés par date de fin croissante.

II.2. (a) Montrer par un contre-exemple (dessin) que l'algorithme glouton proposé ci-dessus ne produit pas nécessairement une solution optimale.

(b) Formaliser le problème en exprimant une solution optimale pour les cours $\{1, \cdots, k\}$ à partir des solutions optimales trouvées pour les cours $\{1, \cdots, i\}$ avec $1 \leqslant i \leqslant k-1$.

(c) Illustrer par un dessin les différentes situations possibles.

(d) Écrire l'algorithme qui construit une solution optimale, on justifiera les structures de données auxiliaires utilisées.

(e) Donner une trace de votre algorithme sur l'exemple :

cours	a	b	c	d	e	f	g	h	i	j	k	l	m	n
début	1	1	2	2	3	5	6	4	8	6	8	9	10	9
fin	2	3	4	5	5	7	8	8	9	10	11	11	12	12
crédits	1	2	3	4	1	4	3	2	1	2	1	4	2	3

(f) Évaluer le coût de l'algorithme.

Un autre étudiant suggère que ce problème est simplement un problème de recherche d'un chemin optimal dans un graphe associé au cours

II.3. Reformuler ce problème en terme de graphe orienté sans circuit et adapter un algorithme classique pour le résoudre.

Problème : Des chemins... (∼ *1h30*)

Dans un graphe orienté dont les arcs sont valués (valuation positive), l'algorithme de Dijkstra permet de calculer une arborescence des chemins de plus petite valeur [1] d'une source s à chacun des sommets du graphe.

La structure de l'algorithme est donnée ci-après.

`Algorithme_Dijkstra` ($\mathcal{G}$,s)

Données : $\mathcal{G} = (\mathcal{X}, \mathcal{A}, \mathcal{V})$ graphe orienté valué positif [a], s sommet de $\mathcal{G}$

Résultat : Visite tous les sommets accessibles x du graphe à partir de s et calcule la valeur minimale des chemins de s à x

foreach $x \in \mathcal{X}$ **do** $d(x) = +\infty$
$S = \{s\}$
$d(s) = 0$
repeat
  `Choisir` $y \notin S$ minimisant

$$\pi(y) = \min_{x \in S, (x,y) \in \mathcal{A}} d(x) + v(x, y)$$

  $S \leftarrow S \cup \{y\}$
  $d(y) = \pi(y)$
until *il existe* $y \notin S$ *et* $x \in S$ *tels que* $(x, y) \in \mathcal{A}$

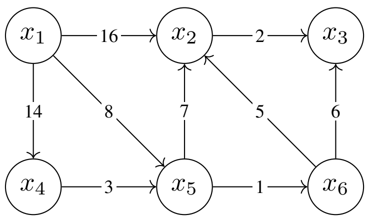
a. $\mathcal{A}$ ensemble des arcs, $v(x, x) = 0$, $v(x, y) \geqslant 0$ si $(x, y) \in \mathcal{A}$, $v(x, y) = +\infty$ si $(x, y) \notin \mathcal{A}$

III.1. Proposer une modification de l'algorithme ci-dessus et proposer un algorithme `chemin` (s, y) qui renvoie la liste des sommets d'un chemin de valeur minimale de la source s au sommet y. Quel est le coût de l'algorithme `chemin` ?

On souhaite maintenant construire les valeurs minimales des chemins pour tout sommet x à un puits p donné.

III.2. Proposer un algorithme qui résolve ce problème et après avoir exécuté cet algorithme, proposer un algorithme `chemin` (x, p) qui renvoie la liste des sommets d'un chemin de valeur minimale du sommet x au puits p.

III.3. Donner une trace de l'exécution de l'algorithme proposé sur le graphe suivant en prenant le sommet x_3 comme puits.



Pour chercher un chemin de valeur minimale de s à p, on pourrait simultanément exécuter l'algorithme de recherche du plus court chemin de la source s et l'algorithme construit dans la question III.2 vers le puits p. On cherche un chemin minimal en commençant par les 2 bouts. On appelle cet algorithme l'algorithme de Dikstra bi-directionnel.

III.4. Faire un dessin explicatif de cette idée.

III.5. Écrire un algorithme basé sur ce principe, dans un premier temps on ne cherchera que la valeur minimale d'un chemin de s à p

III.6. Écrire la trace de cet algorithme lorsque on l'exécute sur l'exemple de la question III.3.

III.7. Analyse de l'algorithme :

(a) Donner un invariant de l'itération et justifier la condition d'arrêt de l'itération.

(b) En supposant l'algorithme de Dijkstra prouvé, faire la preuve de l'algorithme de Dijkstra bi-directionnel.

(c) Comparer la complexité de cet algorithme à la complexité de l'algorithme de Dijkstra, commenter.

III.8. L'utilisation d'une heuristique admissible comme dans l'algorithme A^* peut-elle s'appliquer dans le cas bi-directionnel ? Si oui comment la mettre en place, si non expliquer pourquoi.

1. la valeur d'un chemin est la somme des valeurs des arcs composant le chemin