

## 8INF433 Algorithmique

### Devoir 3

(À remettre au plus tard mardi le 2 avril 2019)

**Justifiez et expliquez toutes vos réponses.**

**Vous serez évalué pour la rectitude et la qualité de vos réponses.**

**Travaillez en équipe de trois ou quatre étudiants.**

**Remettre dans mon casier une copie papier par équipe.**

1. Nous avons vu en classe un algorithme vorace pour résoudre le problème du retour de la monnaie en un nombre minimum de pièces. Nous avons observé que cet algorithme fonctionne correctement avec de pièces de monnaie canadiennes mais celui-ci ne trouve pas toujours la bonne réponse si, par exemple, de pièces de 12 cents sont ajoutées (e.g. il y a une erreur pour retourner 15 cents). Le problème général du retour de la monnaie peut être résolu exactement en utilisant la programmation dynamique.

Soit  $n$  le nombre de pièces distinctes et soit  $T[1 \dots n]$  un tableau donnant la valeur de ces pièces (il n'y a aucun intérêt à trier ce tableau). Supposons une quantité illimitée de chaque type de pièces. Soit  $L$ , une limite sur le montant à obtenir.

- (a) Pour  $1 \leq i \leq n$  et  $1 \leq j \leq L$ , soit  $c_{i,j}$  le nombre minimum de pièces pour obtenir le montant  $j$  si on se limite aux pièces de type  $T[1], T[2], \dots, T[i]$ . Si ce montant ne peut être obtenu alors  $c_{i,j} = +\infty$ . Donnez une équation de récurrence pour  $c_{i,j}$  incluant les conditions initiales.
- (b) Donnez un algorithme de programmation dynamique pour calculer tous les  $c_{n,j}$  où  $1 \leq j \leq L$ . Votre algorithme ne doit utiliser qu'un seul tableau (à une dimension) de longueur  $L$ .
- (c) Analysez le temps d'exécution de votre algorithme en fonction de  $n$  et de  $L$ .

2. Donnez toutes les étapes de l'algorithme de Floyd pour calculer le plus court chemin entre toutes les paires de noeuds du graphe de la figure 1. Donnez la matrice obtenue à chaque itération.

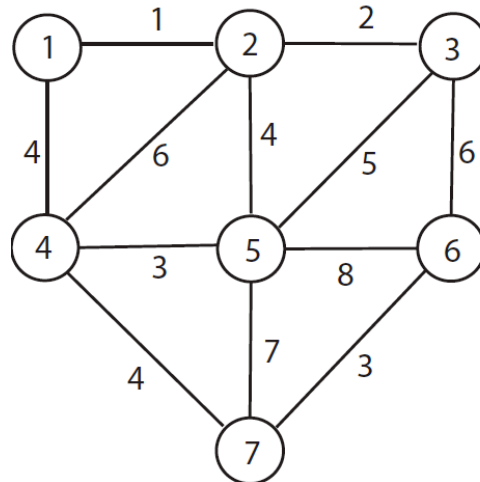


Figure 1:

3. L'algorithme de Floyd que nous avons vu en classe donne la longueur du plus court chemin entre toutes les paires de noeuds mais ne permet pas de retrouver les plus courts chemins. Montrer comment modifier l'algorithme afin qu'il mémorise l'information permettant de retrouver efficacement (temps  $O(n)$ ) le plus court chemin entre deux noeuds quelconque du graphe.
4. Supposez que vous ayez un damier  $n \times n$  et un jeton. Vous devez déplacer le jeton depuis le bord inférieur du damier vers le bord supérieur, en respectant la règle suivante. À chaque étape, vous pouvez placer le jeton sur l'un des trois carrés suivants
- le carré qui est juste au dessus,
  - le carré qui est une position plus haut et une position plus à gauche (à condition que le jeton ne soit pas déjà dans la colonne la plus à gauche),
  - le carré qui est une position plus haut et une position plus à droite (à condition que le jeton ne soit pas déjà dans la colonne la plus à droite).

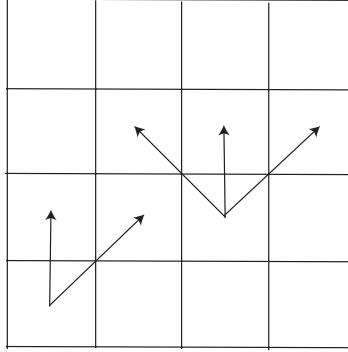


Figure 2:

La figure 2 illustre les déplacements valides à partir de deux positions dans un damier de dimension  $4 \times 4$ .

Chaque fois que vous vous déplacez du carré  $x = (i, j)$  au carré  $y \in \{(i-1, j-1), (i-1, j), (i-1, j+1)\}$ , vous recevez  $p(x, y)$  dollars. On vous donne  $p(x, y)$  pour toutes les paires  $(x, y)$  correspondant à un déplacement licite de  $x$  à  $y$ .  $p(x, y)$  **n'est pas forcément positif**.

Donner un algorithme efficace de programmation dynamique pour déterminer le montant maximum que vous pouvez empocher. Votre algorithme peut partir de n'importe quel carré du bord inférieur et arriver sur n'importe quel carré du bord supérieur pour maximiser le montant collecté au cours du trajet.

Expliquez comment vous pouvez optimiser l'espace mémoire et évaluez le temps d'exécution et l'espace mémoire de votre algorithme.