

Programmation avancée

Répétition 9: Résolution de problèmes

Jean-Michel BEGON

23 décembre 2016

1 Approximation gloutonne

Proposez une approximation gloutonne, ainsi qu'un contre-exemple démontrant la sous-optimalité pour les problèmes suivants :

- (a) Couture minimale
- (b) Plus longue sous-séquence palindromique
- (c) Multiplications matricielles

2 Arbre couvrant minimum

Soit le graphe valué $G = (V, E, W)$, où V est l'ensemble des nœuds (*vertex*) du graphe, $E \subseteq V \times V$ est l'ensemble des arêtes (*edge*) de G et $W : E \rightarrow \mathbb{R}_+$ est la fonction de poids de chaque arête. Par exemple, les nœuds représentent des routeurs dans un réseau informatique et le poids correspond à la capacité du lien. On désire réaliser un arbre couvrant minimum (*Minimum spanning tree*), c'est-à-dire un arbre (graphe acyclique connexe) qui est tel que la somme des poids des branches est minimum. Proposez un algorithme glouton pour solutionner ce problème.

3 La course de Julien (adapté de CLRS, 16.2-4)

Julien participe à une course à pieds qui consiste à relier une ville A à une ville B . Etant donné la contenance de son bidon, il sait qu'il peut parcourir m mètres sans tomber à cours d'eau. Il a également pu obtenir des organisateurs du marathon la liste des endroits où il pourra remplir son bidon et les distances entre ces endroits.

Julien aimerait minimiser le nombre d'arrêts qu'il devra effectuer lors de sa course pour remplir son bidon. Donnez une méthode efficace pour déterminer quels arrêts il devrait faire. Montrez que votre stratégie donne une solution optimale et discutez sa complexité.

4 L'ordonnancement

L'ordonnancement des tâches est un problème fréquent en informatique : on dispose d'un ensemble $S = \{s_1, s_2, \dots, s_n\}$ de n tâches qu'on doit ordonner. A chaque tâche s_i est associé un temps d'exécution t_i . Une fois l'ordre des tâches fixé, on peut également associer à chaque tâche un temps de complétion (ou temps de réponse), c'est-à-dire, le temps qui s'est écoulé jusqu'à la fin de la tâche.

Par exemple, si on dispose des tâches s_1 et s_2 dont les temps d'exécution respectifs sont $t_1 = 3$ et $t_2 = 5$ et si on exécute les tâches dans l'ordre $\langle s_1, s_2 \rangle$, alors les temps de complétion seront :

- $C_1 = t_1 = 3$
- $C_2 = C_1 + t_2 = 3 + 5 = 8$

En outre, la somme des temps de complétion sera $C_\Sigma = \sum_i C_i = C_1 + C_2 = 11$.

On souhaite disposer d'un algorithme d'ordonnement des tâches qui minimise la somme des temps de complétion (C_Σ).

- (a) Quelle serait la complexité d'un algorithme *brute-force* pour résoudre ce problème ?
- (b) Est-il possible de faire mieux ? (*i.e.* le problème dispose-t-il de la propriété de sous-structure optimale ?)
- (c) Le cas échéant, proposez un algorithme efficace pour résoudre ce problème.

5 Intervalles unitaires (CLRS, 16.2-5)

Etant donné un ensemble x_1, x_2, \dots, x_n de points sur l'axe réel, donner un algorithme efficace pour déterminer le plus petit ensemble d'intervalles fermés de longueur unitaire qui contient tous les points. Montrez que votre algorithme est correct.

6 La peinture (adapté de l'IEEEExtreme 2016)

Vous souhaitez mettre en couleur une toile. Comme vous êtes méthodiques, vous avez déjà acheté tous les pigments nécessaires et vous avez décidé de peindre la toile de haut en bas et droite à gauche. Vous connaissez donc exactement la suite de couleurs à utiliser. Vous disposez de deux pinceaux et souhaitez minimiser le nombre de fois que vous devez nettoyer votre pinceau. Quelle stratégie allez-vous adapter ?

7 Je vais au marché

Vous devez acheter n produits. Le prix initial de chaque produit est 1 euro mais chaque jour le prix du i ème produit augmente d'un facteur $f(i)$. Développez un algorithme efficace pour trouver l'ordre optimal dans lequel les produits devraient être achetés, en supposant que vous pouvez acheter au plus un produit chaque jour.

Bonus

En bioinformatique, l'alignement de séquences est un outil qui permet de représenter deux ou plusieurs séquences d'ADN de manière à en faire ressortir les régions homologues. L'objectif de l'alignement est de disposer les composants de sorte à identifier les zones de concordance.

Par exemple, étant donné les deux séquences de bases :

```
AGGCTATCACCTGACCTCCAGGCCGATGCC
TAGCTATCACGACCGGGTCGATTTGCCCGAC
```

Leur alignement consiste à appairer une base (un caractère) de l'une soit avec une base de l'autre, soit avec un trou :

```
-AGGCTATCACCTGACCTCCAGGCCGA--TGCCC---
TAG-CTATCAC--GACCGC--GGTCGATTTGCCCGAC
```

Proposer un algorithme par programmation dynamique qui permet d'aligner deux séquences de bases en minimisant le nombre de trous. Etudier sa complexité.