# Structures de données et algorithmes

Répétition 8 : Résolution de problèmes

Jean-Michel Begon

22 avril 2016

## 1 Merge Sort

Dessinez l'arbre de récursion du `MergeSort` pour le problème suivant :

$$A =< 5, 2, 4, 7, 1, 3, 2, 6 >$$

Pourquoi la mémoïsation ne permet-elle pas d'améliorer ce problème ?

## 2 Couture minimale (*Checkerboard*)

Mister B. dispose d'une pièce pavée de $w + 1 \times h$ dalles. A l'exception de la première rangée, chaque dalle est occupée par une pile de dossiers, si bien que la porte de sortie n'est plus accessible. Peu enclin aux travaux domestiques, Mister B. aimerait récupérer l'usage de sa porte tout en minimisant son effort. Il souhaite donc enlever une seule pile par jour et avancer d'une rangée de case à chaque fois (il ne peut enlever que la pile de devant ou celles en diagonales d'une case déjà libérée). Partant du postulat que l'effort est proportionnel à la taille de la pile (et que celle-ci est connue), comment Mister B. peut-il minimiser son effort total ?

## 3 Plus longue sous-séquence palindromique (adapté de CLRS, 15-2)

On souhaite déterminer le plus grand palindrome qui existe au sein d'un mot. Par exemple, le mot `characters` contient un palindrome de taille 5 : `carac` (il ne s'agit pas d'une sous-séquence contigüe).

1. Proposez une solution *brute-force* à ce problème.
2. Proposez une solution par programmation dynamique pour ce problème.

## 4 Multiplications matricielles

Lorsqu'on désire multiplier plusieurs matrices $A_1 \times A_2 \times ... \times A_n$, l'ordre des multiplications peut avoir un impact important. Pour trois matrices $A[m \times n]$, $B[n \times p]$, $C[p \times q]$, le coût du produit $(A \times B) \times C$ est de $\Theta(mnp + mpq)$ alors que le coût du produit $A \times (B \times C)$ est de $\Theta(mnq + npq)$. En fonction des valeurs de $m, n, p, q$ un des produits est plus avantageux que l'autre.

1. Proposez une solution *brute-force* à ce problème.
2. Proposez une solution par programmation dynamique pour ce problème.

## 5 Le problème de la partition

On souhaite déterminer s'il est possible de partitionner un tableau d'entiers positifs $A$ en deux sous-tableaux de somme identique.

1. Proposez une solution *brute-force* à ce problème. Quelle est sa complexité ?
2. Proposez un algorithme par programmation dynamique pour ce problème. Quelle est sa complexité ?

## 6 Distance d'édition

La distance d'édition permet de calculer le nombre d'opérations nécessaires pour changer une chaîne $A$ en une chaîne $B$ avec les opérations suivantes :

— Insertion d'un caractère.
— Suppression d'un caractère.
— Substitution d'un caractère par un autre.

1. Proposez une solution *brute-force* à ce problème. Quelle est sa complexité ?
2. Proposez un algorithme par programmation dynamique pour ce problème. Quelle est sa complexité ?

## 7 Approximation gloutonne

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

1. Couture minimale
2. Plus longue sous-séquence palindromique
3. Multiplications matricielles

## 8 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.

## 9 L'ordonnancement

L'ordonnancement des tâches est un problème fréquent en informatique : on dispose d'un ensemble $S = \{s_1, s_2, ..., s_n\}$ 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 $< s_1, s_2 >$, 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'ordonnancement des tâches qui minimise la somme des temps de complétion ($C_\Sigma$).

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

## 10 Intervalles unitaires (CLRS, 16.2-5)

Etant donné un ensemble $x_1, x_2, ..., x_n$ 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.

## 11 Le produit

Soient $A$ et $B$, deux ensembles contenant chacun $n$ entiers positifs. A partir d'eux, vous pouvez créer deux séquences : $\{a\}_{i=1}^n$ à partir de $A$ et $\{b\}_{i=1}^n$ à partir de $B$. Comment pouvez-vous maximiser le gain suivant :

$$\prod_{i=1}^{n} a_i^{b_i}$$

## 12 Le bar (offline caching)

Vous êtes à la tête d'un bar qui propose $N$ boissons différentes. Votre plan de travail étant réduit, vous ne pouvez stocker que $k$ ($k < N$) bouteilles près de vous. Les autre boissons doivent être stockées plus loin et demandent donc plus de temps à servir.

En imaginant que les bouteilles sont inépuisables et que vous connaissez à l'avance l'ordre des $M$ commandes qui vont vous occuper sans relâche jusqu'à la fermeture, comment organisez-vous votre bar ?

## 13 Les fractions égyptiennes

Soit une fraction $0 < \frac{a}{b} < 1$. Trouvez un ensemble d'entiers $S = \{s_1, ...s_n\}$ différents ($s_i = s_j \iff i = j$) tel que

$$\sum_{i=1}^{n} \frac{1}{s_i} = \frac{a}{b}$$

Ce résultat est-il optimal au point de vue de la taille de $S$ ?

## 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 :

```
AGGCTATCACCTGACCTCCAGGCCGATGCCC
TAGCTATCACGACCGCGGTCGATTTGCCCGAC
```

Leur alignement consiste à apparier 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.