TD 9 et 10 Révisions

1 Diviser pour régner

Exercice 1. On nous donne une fonction croissante $f : \mathbb{Z} \to \mathbb{Z}$, on peut appeler f en $\mathcal{O}(1)$. On souhaite trouver le premier N tel que $f(N) \geq 0$. Donnez un algorithme en $\mathcal{O}(\log N)$ permettant de trouver N.

2 Backtracking

Exercice 2. Étant donné un nombre N et un chiffre C, on souhaite exprimer N à l'aide des opérations $+C$, $-C$, $*C$, $/C$ en partant de C (les opérations seront toujours parenthésées à gauche quand il y a une ambigüité). Par exemple :

— pour $N = 100$ et $C = 7$: $(((7 + 7) * 7) * 7 + 7 + 7)/7$
— pour $N = 7$ et $C = 4$: $(4 + 4 + 4)/4 + 4$

Le but va être d'écrire un algorithme permettant de calculer le nombre minimal d'opérations permettant d'exprimer N en fonction de C.

1. Montrer que qu'il existe toujours une expression permettant d'exprimer N en fonction de C, déduisez-en une borne supérieure sur la longueur de l'expression minimale.
2. Écrivez une fonction `minlen(N,C,longueurmax)` qui étant donné N et C permet de trouver l'expression la plus courte, de longueur maximale `longueurmax`, permettant d'exprimer `N`.
3. Quelle est la complexité de `minlen` ?
4. Faites en sorte que votre fonction adapte `longueurmax` en fonction des résultats déjà trouvés pour les appels récursifs suivants.
5. En utilisant de la mémoization, quelle est la complexité maximale de cet algorithme de backtracking ?

3 Programmation Dynamique

Exercice 3. Étant donné un nombre n, on souhaite trouver le nombre minimal d'opérations permettant d'arriver à un. Les opérations autorisées sont :

— Diviser n par 3 si celui-ci est divisible par 3.
— Diviser n par 2 si celui-ci est divisible par 2.
— Décrémenter n de 1.

Donnez une formule de récurrence puis un algorithme permettant de trouver ce nombre d'étapes ainsi que sa complexité en fonction de n.

Exercice 4. Étant donné un tableau d'entiers positifs, on souhaite compter le nombre de sous-suite croissantes qu'il contient.

Après avoir donné une formule de récurrence et l'avoir expliquée, vous donnerez l'algorithme correspondant ainsi que sa complexité.

Exercice 5. Un palindrome est une chaîne de caractère qui se lit de la même manière dans les deux sens, par exemple : `abba`, `kayak`, `a`. Étant donné une chaîne de caractères, on souhaite trouver le plus petit nombre de caractères à supprimer pour la transformer en palindrome. Vous devez trouver un algorithme en $\mathcal{O}(n^2)$ permettant de résoudre ce problème.

Exercice 6. On dispose initialement de N œufs, et d'un immeuble haut de H étages - l'étage 0 est le rez-de-chaussée, le dernier étage étant donc le $H-1$ème.

On suppose, pour se faciliter la tâche, que les œufs sont identiques et se comportent de la façon suivante :

— Un œuf qui se casse en chutant ne peux pas être gardé, mais on peut réutiliser sans risque un œuf qui y a résisté.

— Un œuf qui ne se brise pas en chutant de l'étage k ne se serait pas non plus brisé en chutant de l'étage $k-1$; réciproquement, s'il se casse en chutant de l'étage k, il n'aurait pas non plus résisté à une chute plus haute.

— Avant l'expérience, on ne sait rien : il est possible que les œufs se brisent même en chutant du rez-de-chaussée, ou à l'inverse qu'ils ne se cassent pas même en tombant du sommet de l'immeuble

Le but est de déterminer *exactement* l'étage à partir duquel la chute devient trop haute pour nos œufs, en minimisant le nombre de tentatives effectuées $T(N,H)$ — en particulier, on ne se préoccupe pas de savoir combien d'œufs vont être cassés, pourvu que l'on détermine l'étage en question en un minimum de lancers. Autrement dit, $T(N,H)$ est le nombre de nombre minimal de jeters d'œufs qui permette dans tous les cas de détérminer l'étage limite.

1. Déterminer un algorithme optimal quand on ne dispose que d'un œuf ($N=1$). Combien de tentatives doit-on faire dans le pire des cas ?
2. Considérez les deux situations suivantes :
   — Il ne vous reste plus que n œufs mais après plusieurs lâchers, vous avez déterminé qu'ils résistent à la chute de l'étage 0 mais pas à celle de l'étage k
   — Il vous reste n œufs, et vous savez qu'ils résistent à la chute de l'étage k' mais pas celle de l'étage $k'+k$

   Que dire du nombre optimal de tentatives restantes dans ces deux scénarii ?
3. En déduire une relation de récurrence sur T, puis un algorithme calculant $T(N,H)$. Quelle est sa complexité ?
4. (Bonus) Donner un algorithme en $\mathcal{O}(NH\log(H))$
5. (Bonus difficile) Donner un algorithme en $\mathcal{O}(N\log H)$. Indication : Considérer le problème sous l'angle suivant : si on note m l'étage limite cherché, poser $F(J,N)$ le nombre maximal de m pouvant être distingués en J jeters et avec N œufs.