

## TD4

Dans ce TD, on s'intéresse à la détermination d'une partition équilibrée d'un tableau d'entiers positifs : on suppose avoir un tableau  $E$  (que l'on représentera par une liste) d'entiers naturels que l'on souhaite partitionner en deux sous-ensembles  $E_1$  et  $E_2$  (donc tels que  $E_1 \cup E_2 = E$  et  $E_1 \cap E_2 = \emptyset$ ) tels que  $|s(E_1) - s(E_2)|$  soit minimale, où  $s(E_1)$  et  $s(E_2)$  désignent la somme des éléments de  $E_1$  et  $E_2$ .

1. Justifier que le problème est équivalent à la recherche d'un sous-ensemble  $E_1$  de  $E$  tel que  $|s(E)/2 - s(E_1)|$  soit minimale.
2. Quelle serait la complexité d'une recherche de  $E_1$  exhaustive : déterminer cette valeur minimale en examinant tous les sous-ensembles  $E_1$  de  $E$  (en fonction de  $n$ , le cardinal de  $E$ ) ?

Dans la suite, on note  $S = s(E)/2$  la valeur que l'on cherche à approcher par la somme des éléments de  $E_1$  et on adopte la démarche suivante : soit  $E[0]$  n'appartient pas à  $E_1$  et dans ce cas  $E_1$  est un sous-ensemble de  $E[1:]$  qui approche au mieux  $S$ , soit  $E[0]$  appartient à  $E_1$  et dans ce cas  $E_1 \setminus \{E[0]\}$  est un sous ensemble de  $E[1:]$  qui approche au mieux  $S - E[0]$ . Puis on recommence avec l'ensemble  $E[1:]$  selon la valeur à approcher.

3. On considère la fonction récursive suivante qui suit la démarche précédente :

```
def partition1(E) :
    S = sum(E)/2
    def p(E,S) :
        if len(E) == 1 :
            r = E
        else :
            F1 = p(E[1:], S)
            s1 = sum(F1)
            F2 = p(E[1:], S-E[0])
            s2 = sum(F2)+E[0]
            if abs(s1-S) < abs(s2-S) :
                r = F1
            else :
                r = [E[0]] + F2
        return r
    return p(E,S)
```

Quelle est la complexité de cette fonction par rapport à  $n$ , la longueur de  $E$  ?

### 4. Approche ascendante

- a) On commence par remplir un tableau  $T$  (liste de listes) de taille  $(1 + \text{len}(E)) \times (1 + S)$  de sorte que, si  $0 \leq i \leq \text{len}(E)$  et  $0 \leq j \leq S$ , alors  $T[i][j] = \text{True}$  s'il est possible d'obtenir l'entier  $j$  avec la somme de certains éléments de  $E[:i]$ ,  $\text{False}$  sinon. La première ligne ( $i = 0$ ) est donc  $[\text{True}, \text{False}, \text{False}, \dots, \text{False}]$  car  $E[:0] = \emptyset$  donc la seule somme que l'on peut obtenir est  $j = 0$ .

Par exemple, si  $E = [2, 3, 2]$ , alors  $S = 7$  et  $T$  sera

	0	1	2	3	4	5	6	7
0	True	False						
1	True	False	True	False	False	False	False	False
2	True	False	True	True	False	True	False	False
3	True	False	True	True	True	True	False	True

Pour remplir ce tableau, on peut remarquer que  $T[i][j] = \text{True}$  si et seulement si on a  $T[i-1][j] = \text{True}$  ou  $T[i-1][j-E[i-1]] = \text{True}$

Écrire une fonction `tableau(E:list)->list` qui prend en argument la liste représentant le tableau  $E$  et qui renvoie une liste de liste comme remplie précédemment.

Une fois le tableau rempli, on raisonne de la façon suivante :

- commence par déterminer la somme des éléments de  $E_1$  que l'on va chercher à obtenir : on choisira la somme la plus proche de  $\lfloor S/2 \rfloor$ . Pour cela, on cherche dans les sommes utilisant tous les éléments de  $E$  (donc dans la dernière ligne) le plus grand indice  $m \leq \frac{S}{2}$  contenant la valeur  $\text{True}$ . Sur le tableau en exemple, on aurait  $m=3$ .
  - Une fois la valeur de  $m$  déterminée, reste à reconstruire l'ensemble  $E_1$  dont  $m$  est la somme.
- b) Quel renseignement sur  $E_1$  est apporté par la détermination d'un indice  $i$  pour lequel on a  $T[i][m] = \text{False}$  et  $T[i+1][m] = \text{True}$  ?
  - c) En déduire le code d'une fonction `partition(E:list)->list` qui prend en argument le tableau  $E$  et qui renvoie le tableau  $E_1$ .
  - d) Quelle est la complexité de cette fonction en fonction de  $n = \text{len}(E)$  et  $S = \text{sum}(E)$  ?

### 5. Approche récursive

Comment améliorer la fonction récursive donnée dans la question 3, en utilisant le principe de mémorisation ?