

Correction TD 04 – Subset-sum et Partition

Exercice 3.

Subset-sum et Partition

3. Montrer que **Partition** $\leq_m^P$ **Subset-sum**.

Intuition : il faut expliquer comment, si vous aviez accès à un algorithme efficace pour **Subset-sum**, vous pourriez l'utiliser pour résoudre efficacement **Partition**.

Il s'agit d'une réduction de **Partition** vers **Subset-sum** montrant que **Partition** $\leq_m^P$ **Subset-sum**,

Pour transformer une instance de **Partition** en une instance de **Subset-sum**, on commence par calculer le total $t = \sum_{i=1}^n e_i$. Si t est impair, avoir une bi-partition de même somme est impossible et on produit une instance négative de **Subset-sum** (aucun élément et somme à 1). Si t est pair alors chacune des deux parties doit avoir somme $\frac{t}{2}$, ce qui correspond à demander s'il existe un sous-ensemble *parmi les mêmes éléments* dont la somme est $\frac{t}{2}$ (s'il existe un sous-ensemble de somme $\frac{t}{2}$ alors le complémentaire a également somme $\frac{t}{2}$).

```

algorithme Partition(e) :
    n = longueur(e)
    t = 0
    pour i de 1 à n faire :
        t = t + e[i]
    si t%2 == 1 alors :
        retourner ([], 1)
    sinon :
        retourner (e, t/2)

```

Temps linéaire $\Theta(n)$.

4. Montrer que **Subset-sum** $\leq_m^P$ **Partition**.

Intuition : il faut expliquer comment, si vous aviez accès à un algorithme efficace pour **Partition**, vous pourriez l'utiliser pour résoudre efficacement **Subset-sum**.

Nous allons présenter une réduction many-one polynomiale, de **Subset-sum** vers **Partition**, montrant **Subset-sum** $\leq_m^P$ **Partition**.

Soit une instance $e_1, \dots, e_n \in \mathbb{N}$ avec objectif s du problème **Subset-sum**, et $t = \sum_{i=1}^n e_i$. L'idée consiste à ajouter deux entiers x et y très grands tels que $x + s = y + (t - s)$, qui seront forcés d'être dans des parties différentes, et qui nécessiteront d'être complétés avec les entiers $e_1, \dots, e_n$ en deux paquets de somme respective s et $t - s$ pour arriver à l'égalité $x + s = y + (t - s)$. On ajoute donc les entiers $x = 2t - s$ et $y = 2t - (t - s) = t + s$ à la liste, pour obtenir une instance de **Partition** qui est positive si et seulement si l'instance de **Subset-sum** avec objectif s est positive.

$\Leftarrow$ Supposons que **Subset-sum** a une solution (sur les éléments $e_1, \dots, e_n$ avec l'objectif s). On met x et y chacun dans une partie, puis on complète la partie contenant x avec la solution de **Subset-sum** pour sommer à $x + s = 2t$, et on met tous les autres éléments avec y qui vont sommer à $y + (t - s) = 2t$. Donc on a une solution à **Partition**.

$\Rightarrow$ Supposons que **Partition** a une solution (sur les éléments $e_1, \dots, e_n, x, y$). On ne peut pas avoir x et y dans la même partie car $x + y = 3t$ et le reste qui vaut seulement t ne peut pas compenser pour obtenir une seconde partie de même somme. Donc x et y sont chacun dans une partie, puis pour arriver à la même somme qui doit être $\frac{t+x+y}{2} = 2t$, on

doit compléter la partie contenant x avec une sélection d'éléments de somme $2t - x = s$.
Donc on a une solution à **Subset-sum**.

Remarquons que l'on aurait pu ajouter n'importe quel $z \geq 0$ en plus à x et à y (par exemple $z = 42t$ donnerait $x = 44t - s$ et $y = 43t + s$), le raisonnement resterait valide.

```
algorithme Subset-sum(e, s) :  
    n = longueur(t)  
    f = tableau de longueur n + 2  
    t = 0  
    pour i de 1 à n faire :  
        f[i] = e[i]  
        t = t + e[i]  
    f[n+1] = 2t-s  
    f[n+2] = t+s  
    retourner f
```

Temps linéaire $\Theta(n)$.