IFT 2121 H2000 **Pierre McKenzie**
Introduction à l'algorithmique

Devoir 3

à remettre *au plus tard* le mercredi 16 février à 11:31

1. Soit $m \geq 0$ un montant entier à remettre et soit $k \geq 0$ un entier. Démontrez que si l'on dispose de $n_0 \geq m$ pièces de 1 unité, de $n_1 \geq 0$ pièces de 2 unités, de $n_2 \geq 0$ pièces de 4 unités, et ainsi de suite jusqu'à $n_k \geq 0$ pièces de 2^k unités, alors l'algorithme vorace bien connu remet le montant m en utilisant le nombre minimum de pièces. (Suggestion: une façon de procéder consiste à démontrer d'abord le

   Lemme. Pour tout entier $l \geq 1$, si $a_0, a_1, \ldots, a_{l-1}$ sont des naturels qui vérifient $\sum_{i=0}^{l-1} a_i 2^i \geq 2^l$, alors il existe des naturels $b_0 \leq a_0, b_1 \leq a_1, \ldots, b_{l-1} \leq a_{l-1}$ tels que $\sum_{i=0}^{l-1} b_i 2^i = 2^l$.)

2. Appelons C_i, $1 \leq i \leq n$, la colonne i d'une matrice $A \in \mathbb{Z}^{n \times n}$, et définissons $\mathit{poids}(C_i)$ comme la somme des valeurs absolues des entrées de C_i. Donnez sans justification un algorithme vorace résolvant le problème suivant:

   Donnée: une matrice $A \in \mathbb{Z}^{n \times n}$.
   Calculez: un ensemble $S \subseteq \{1, 2, \ldots, n\}$ tel que $\{C_i \mid i \in S\}$ est un ensemble linéairement indépendant et $\sum_{i \in S} \mathit{poids}(C_i)$ est maximal.

   Vous pouvez faire appel à une fonction booléenne *Indép*(S) qui retourne `vrai` ssi $\{C_i \mid i \in S\}$ est linéairement indépendant.

3. Faites le problème 7.34 de la page 255 du livre d'algorithmique. [Précisions: 1) les entiers n_i sont distincts, et 2) il vous suffit de supposer l'existence d'un algorithme $\mathit{Mult}(q_1, q_2)$ capable de multiplier deux polynômes de degré $\leq k+1$ en un temps dans $O(k \log k)$, et 3) par analyse on entend la justification de l'ordre de votre algorithme, et 4) cet ordre doit être strictement inclus dans $O(n^2)$.]

4. Le but de ce problème est de construire un circuit qui accepte en entrée n entiers et qui produit à la sortie la somme de ses entrées. Chaque entier est représenté en binaire et possède n bits.

   La *profondeur* d'un circuit est la longueur ℓ du plus long chemin d'une entrée du circuit à une sortie. Notez que deux circuits de profondeur ℓ peuvent être placés côte à côte pour former un nouveau circuit plus grand dont la profondeur est toujours égale à ℓ.

   Un *additionneur* est un circuit de profondeur $\Theta(\log m)$ qui accepte en entrée deux entiers de m bits et qui calcule à la sortie la somme de ces deux entiers.

   Un *compacteur* est un circuit de profondeur 2 qui accepte en entrée 4 entiers a, b, c, d de m bits chacun et qui calcule à la sortie 2 entiers x et y de m bits chacun ayant la propriété que les m bits les moins significatifs de $a+b+c+d$ sont les m bits les moins significatifs de $x+y$.

   En prenant pour acquis l'existence d'additionneurs et de compacteurs, donnez la construction d'un circuit, dont la profondeur est la plus faible possible, pour calculer la somme de n entiers de n bits. Donnez ensuite l'ordre exact de la profondeur de votre circuit en fonction de n, après avoir posé une récurrence pertinente à la portion "diviser pour régner" de votre solution. (Vous pouvez juxtaposer des bits 0 aux entrées et manipuler partout des entiers plus longs que nécessaire pour simplifier l'analyse si vous le désirez.)