EPFL - Information, Calcul et Communication - Partie théorique

Série 7: Solutions

1 Nombre de bits

a)

1. Il y a environ 320 étudiants inscrits au cours ICC en GC, SIE et MX, donc le nombre de bits nécessaires pour représenter ce nombre vaut $\lceil \log_2(320) \rceil = 9$.
2. Il y a environ 14'000 étudiants inscrits en tout à l'EPFL, donc $\lceil \log_2(14'000) \rceil = 14$ bits sont nécessaires.
3. La vidéo avec le plus de vues sur Youtube est "Baby Shark Dance", avec environ 17,25 millards de vues, donc 34 bits sont nécessaires. Youtube a d'ailleurs eu un problème d'overflow à ce sujet en 2014 lorsque le nombre de vues de la vidéo "Gangnam Style" de Psy a dépassé $2'147'483'647 = 2^{31} - 1$, qui était la valeur maximum représentable sur 32 bits avec des entiers signés. Depuis, Youtube est passé à une représentation à 64 bits du nombre de vues, ce qui laisse une certaine marge pour l'occurrence du prochain overflow!
4. Le nombre d'habitants sur la planète est de 8,3 millards environ (en automne 2026), donc le nombre de bits nécessaires pour représenter cette information vaut 33.

b)

1. 250 étant plus petit ou égal à $255 = 2^8 - 1$, le nombre des étudiants présents est un nombre représentable sur 8 bits.
2. Pour établir la liste des étudiants présents, on a par contre besoin de savoir pour *chaque* étudiant inscrit s'il est présent (1) ou non (0). On a donc besoin ici de 250 bits.

2 Conversion de décimal en binaire

conversion de décimal en binaire

entrée : *L liste de chiffres de 0 à 9, de taille n*
sortie : *M liste de bits, de taille m*

$N \longleftarrow 0$
Pour *i allant de* 1 *à n*
  $N \longleftarrow N + L(i) * 10^{n-i}$
$m \longleftarrow 0$
$M \longleftarrow ()$ *(liste vide)*
Tant que $N > 0$
  $m \longleftarrow m + 1$
  $M \longleftarrow (N \mod 2) \oplus M$ *(ajouter au début de la liste M un élément dont la valeur est le reste de la division entière de N par 2)*
  $N \longleftarrow N \div 2$ *(remplacer N par le quotient de la division entière de N par 2)*
Si $M = ()$ *(pour gérer le cas particulier où* $N = 0$*)*
  $m \longleftarrow 1$
  $M \longleftarrow (0)$
Sortir : M, m

Vu qu'il y a deux boucles qui se suivent, et que m est proportionnel à n, la complexité de l'algorithme est $\Theta(n)$ (et donc $\Theta(\log_2(N))$ si on veut exprimer celle-ci en fonction de la valeur du nombre N à représenter).

3 Utiliser la décomposition binaire

a) Exemple de déroulement de l'algorithme pour $x = 7$ et $y = 5$:

x	7	7	14	28	28
y	5	4	2	1	0
z	0	7	7	7	35

Cet algorithme permet donc de calculer le produit entre deux nombres entiers: **devinette**$(x, y) = x \cdot y$. Plus précisément, on utilise ici la décomposition binaire du nombre $y = \sum_{i=0}^{n-1} b_i \, 2^i$, avec $b_i \in \{0, 1\}$. Vous pouvez vérifier (même si ça n'est pas forcément facile à voir au premier coup d'oeil) que la boucle de l'algorithme effectue le calcul $x \cdot y = \sum_{i=0}^{n-1} b_i \, x \, 2^i$ (en fait, ceci correspond à une "bête" multiplication en colonnes, mais en binaire).

b) Si on suit l'approche "naïve" choisie dès le début de ce cours qui consiste à dire "une instruction = une opération", alors cet algorithme effectue dans le pire des cas $\Theta(n)$ opérations. Ici, x et y sont des nombres qu'on suppose représentés chacun sur n bits, et à chaque passage de la boucle "tant que", si y est un nombre pair, il est divisé par 2 (donc dans ce cas, sa représentation binaire nécessite un bit de moins), et s'il est impair, on lui soustrait 1 de telle façon qu'au prochain passage, il sera pair; donc dans le pire des cas, après 2 passages, la représentation binaire de y nécessite un bit de moins. Vu que y possède n bits au départ, et que la boucle "tant que" s'arrête lorsque y atteint 0, il faudra au plus $2n$ passages dans la boucle pour que l'algorithme se termine, donc $\Theta(n)$ passages.

Maintenant, à chaque passage, il y a possiblement une addition $z + x$ à effectuer. Or ici, vu que les nombres x et y (et donc aussi z) sont chacun représentés sur n bits, ce sont en fait des nombres gigantesques (de valeur approximativement 2^n), et donc l'approximation "une addition = une opération" n'est plus vraiment valable; chaque addition coûte en fait $\Theta(n)$ opérations. Donc la complexité temporelle de l'algorithme est un $\Theta(n^2)$ dans le pire des cas.

Remarque: Dans une question d'examen, tout ceci serait bien sûr explicité pour qu'il n'y ait pas de confusion possible.

4 10 rats pour 1'000 bouteilles

- La solution de faire tester une bouteille (différente) à chaque rat, n'est clairement pas la bonne, vu la contrainte de temps: en 24h, on n'aura testé que 10 bouteilles, et 990 resteront sur le tapis. Evidemment, on pourrait avoir la chance que l'une des 10 bouteilles testées soit celle qui est empoisonnée, auquel cas on saura que les 990 autres sont bonnes. Mais on ne veut pas compter sur la chance.

- Il semble donc nécessaire de faire goûter plus d'une bouteille à chaque rat. On pourrait par exemple diviser le nombre de bouteilles en 10 ensembles de 100 et faire goûter 100 bouteilles à chaque rat. De cette manière, on serait en mesure de dire que 900 bouteilles sont bonnes, mais on devrait jeter les 100 bouteilles goûtées par le rat qui aura eu des convulsions. Un réel gâchis...

- Pour tirer le maximum d'information de cette séance de dégustation, il est aussi nécessaire de faire goûter la même bouteille à plusieurs rats. Voyons comment procéder si on n'a que 4 bouteilles (numérotées de 1 à 4) et 2 rats à disposition (disons 1 et 2): on fait alors goûter un mélange des bouteilles 1 et 3 au rat 1, ainsi qu'un mélange des bouteilles 1 et 2 au rat 2:

- si les rats 1 et 2 sont tous deux empoisonnés, alors la bouteille 1 l'est
- si seul le rat 2 est empoisonné, alors la bouteille 2 l'est
- si seul le rat 1 est empoisonné, alors la bouteille 3 l'est
- si aucun n'est empoisonné, alors la bouteille 4 l'est

C'est le même principe avec 1'000 bouteilles et 10 rats (qu'on renumérote de 0 à 999 et de 0 à 9, respectivement). Tout d'abord, pour faire un compte rond, un ajoute 24 bouteilles d'eau (non-empoisonnée) à la liste: on obtient ainsi $N = 1024 = 2^{10}$ bouteilles à tester (en oubliant qu'on sait qu'aucune des 24 dernières n'est empoisonnée). On fait ensuite goûter un mélange de 512 bouteilles à chaque rat, en les répartissant comme suit:

- le rat 0 teste un mélange de toutes les bouteilles impaires
- le rat 1 teste un mélange des bouteilles 2,3, 6,7, 10,11, 14,15, 18,19, etc.
- le rat 2 teste un mélange des bouteilles 4,5,6,7, 12,13,14,15, 20,21,22,23, etc.
- et ainsi de suite jusqu'au rat 9 qui teste un mélange des 512 dernières bouteilles

Selon les rats qui sont empoisonnés, on déduit de manière unique le numéro de la bouteille empoisonnée. Notez que cette solution correspond exactement à encoder sur 10 bits (avec la correspondance 1 = rat empoisonné, 0 = rat non-empoisonné) un nombre entre 0 et 1'023 (= le numéro de la bouteille). Il faut toutefois faire attention à l'ordre choisi. Voici un exemple:

rat numéro	9	8	7	6	5	4	3	2	1	0
empoisonné (oui/non)	0	0	0	1	0	0	1	1	0	1

Dans ce cas, la bouteille empoisonnée sera la bouteille numéro $2^6 + 2^3 + 2^2 + 2^0 = 64 + 8 + 4 + 1 = 77$.

5 Pour le plaisir: nombres premiers de Mersenne, nombres parfaits*

a) La décomposition binaire de $N = 2^p - 1 = 1 + 2 + 4 + \cdots + 2^{p-1}$ est simplement une suite de p fois le symbole 1.

b) Non, il se trouve qu'en choisissant $p = 11$, on trouve $N = 2^{11} - 1 = 2'047 = 23 \cdot 49$, qui n'est pas lui-même un nombre premier.

c) La décomposition binaire de $M = 2^{p-1}(2^p - 1) = 2^{p-1} + 2^p + 2^{p+1} + \cdots + 2^{p-2}$ est composée de p fois le symbole 1 suivie de $p - 1$ fois le symbole 0.

d) Vu que M est approximativement d'ordre N^2, le nombre de chiffres qui le composent est approximativement le double de celui de N, soit 50 millions de chiffres.