TD d'Algo n° 1

EPITA ING1 2013; A. DURET-LUTZ

17 novembre 2010

1 Les notations Θ, O et Ω

On rappelle les définitions :

$$\Theta(g(n)) = \{f(n) \mid \exists c_1 \in \mathbb{R}^{+\star}, \exists c_2 \in \mathbb{R}^{+\star}, \exists n_0 \in \mathbb{N}, \forall n \geq n_0, c_1 g(n) \leq f(n) \leq c_2 g(n)\}$$
$$O(g(n)) = \{f(n) \mid \exists c_2 \in \mathbb{R}^{+\star}, \exists n_0 \in \mathbb{N}, \forall n \geq n_0, f(n) \leq c_2 g(n)\}$$
$$\Omega(g(n)) = \{f(n) \mid \exists c_1 \in \mathbb{R}^{+\star}, \exists n_0 \in \mathbb{N}, \forall n \geq n_0, c_1 g(n) \leq f(n)\}$$

Par abus de notation, on note souvent $f(n) = \Theta(g(n))$ à la place de $f(n) \in \Theta(g(n))$. De même $n^3 + \Theta(n^2)$ doit s'interpréter comme une formule de la forme « $n^3 + f(n)$ » avec $f(n) \in \Theta(n^2)$.

1.1 Propriétés

1. Étant données trois fonctions positives f, g et h, prouvez rigoureusement les propriétés suivantes :
   (a) $f(n) = \Theta(f(n))$
   (b) $f(n) = \Theta(g(n)) \iff g(n) = \Theta(f(n))$
   (c) $f(n) = \Theta(g(n))$ et $g(n) = \Theta(h(n))$ implique $f(n) = \Theta(h(n))$
   (d) $\forall \lambda > 0$, $\lambda.\Theta(f(n)) = \Theta(f(n))$
   (e) $\Theta(f(n)) + \Theta(g(n)) = \Theta(f(n) + g(n))$
   (f) $\Theta(f(n)) + \Theta(g(n)) = \Theta(\max(f(n), g(n)))$
   (g) $\Theta(f(n)).\Theta(g(n)) = \Theta(f(n).g(n))$
2. Lesquelles des propriétés ci-dessus utilisent le fait que les fonctions sont positives ?
3. Ces propriétés sont-elles valables en remplaçant Θ par O ?
   Et si l'on remplace Θ par Ω ?
4. Prouvez que les propritétés suivantes sont fausses :
   (a) $f(n) = \Theta(f(n/2))$
   (b) $f(n) = \Theta(g(n)) \iff \log(f(n)) = \Theta(\log(g(n)))$

1.2 Utilisation

1. Prouvez que $4n^2 + 2n \log n \in \Theta(n^2)$
2. Prouvez que $O(n) + O(n^2) = O(n^2)$

2 Un peu de dénombrement

Indiquez le nombre de fois où chaque message ci-dessous est affiché. Exprimez votre réponse en fonction de N.

```
#include <stdio.h>
#define N 10

int main(void)
{
  for (int i = 0; i < N; ++i)
    for (int j = 1; j <= N; ++j)
      puts("Boucle 1");

  for (int i = 0; i < N; ++i)
    for (int j = N/2; j > 0; --j)
      puts("Boucle 2");

  for (int i = 0; i < N; ++i)
    for (int j = 1; j < N; j *= 2)
      puts("Boucle 3");

  for (int i = 0; i < N; ++i)
    for (int j = i; j >= 0; j--)
      puts("Boucle 4");

  for (int i = 0; i < N; ++i)
    for (int j = i/2; j > 0; j--)
      puts("Boucle 5");

  for (int i = 0; i < N; ++i)
    for (int j = i; j >= 0; j -= 2)
      puts("Boucle 6");

  for (int i = 0; i < N; ++i)
    for (int j = i; j > 0; j /= 2)
      puts("Boucle 7");

  return 0;
}
```

La dernière boucle est difficile : vous pouvez ignorer les problèmes d'arrondi en première approximation.
Vous pouvez vérifier toutes vos réponses en appliquant vos formules pour $N = 10$:

```
% gcc -Wall -std=c99 foo.c
% ./a.out | uniq -c
    100 Boucle 1
     50 Boucle 2
     40 Boucle 3
     55 Boucle 4
     20 Boucle 5
     30 Boucle 6
     25 Boucle 7
```