IFT436 - Série d'exercices #2 : la notation asymptotique

Manuel Lafond

Exercice 1: Donnez la notation O des fonctions suivante de façon aussi précise que possible. Justifiez vos réponses.

- $f(n) = 30n^2/n + 15n^{0.7} + 100\sqrt{n}$
- $f(n) = \sin(n)$
- $f(n) = 2^{n+10}$
- $f(n) = 2^n/1000 + n^{100} + n\log n$
- $f(n) = 1000/n$
- $(2^{\log n} + \sqrt{n}) \cdot \frac{1}{8}(\log(n^3) + \log(\log n))$

Exercice 2: Montrez que la base du log n'a pas d'importante en O. C'est-à-dire, montrez que $\log_b(n) \in O(\log n)$ pour tout $b > 1$.

Exercice 3: On va démontrer qu'ignorer les constantes est souvent bien, mais pas toujours.

a. Supposez que vous avez deux algorithmes de tri. Le premier exécute n^2 instructions et le deuxième en exécute $100n$. À partir de quel n est-ce que le deuxième algorithme est avantageux?

b. Supposez que vous avez deux algorithmes de tri. Le premier exécute $n\log n$ instructions et le deuxième en exécute $100n$. À partir de quel n est-ce que le deuxième algorithme est avantageux?

Exercice 4: (Défi) Montrez que $\log(n!) \in \Theta(n\log n)$.

Exercice 5: (Défi) Considérez le problème de la sous-somme donné en devoir, mais avec la variante dans laquelle on cherche un sous-ensemble de exactement k éléments qui somment à un entier donné t. Un algorithme naïf va tester chaque combinaison de k éléments, ce qui prendra un temps au moins $\binom{n}{k}$. Quel temps cela prendra-t-il, en terme asymptotique?

Pour le savoir, montrez que $\binom{n}{k} \in \Theta(n^k)$ pour toute constante $k \in \mathbb{N}$.

Exercice 6: Il est facile de voir que $2 \cdot 2^n \in O(2^n)$. Par contre, on ne peut pas multiplier la base de l'exposant et rester dans le même O. Pour le voir, montrez que $4^n \notin O(2^n)$.
Si vous voulez aller un peu plus loin (un peu), montrez que pour toute constante réelle $c > 1$ et tout réel $\epsilon > 0$, $(c + \epsilon)^n \notin O(c^n)$.

Exercice 7: (Défi) Montrez que $\log n \in O(\sqrt{n})$. En fait, montrez donc que $\log n \in O(n^\epsilon)$ pour toute constante réelle $\epsilon > 0$. Essayez sans utiliser la règle de l'Hôpital.
Indice: Utilisez les faits que $\log n < n$ pour tout $n \geq 2$ et $\log n = \log((\sqrt{n})^2)$. Ceci se généralise bien avec les ϵ.

Exercice 8: Considérez l'algorithme suivant qui décide si un élément r se trouve dans un tableau 2D.

```
fonction findMin(T, n, m, r) //n est la largeur, m est la hauteur
    r = ∞;
    pour i = 1..n faire
        pour j = 1..m faire
            si T[i][j] == r alors
                return true;
            fin
        fin
    fin
    return false;
```

Donnez la complexité de cet algorithme avec la notation Θ. Justifiez votre réponse.

Exercice 9: Considérez l'algorithme suivant qui prend en entrée deux mots

P et T (i.e. des strings), et trouve une occurrence de P dans T, s'il y en a une.

fonction *trouverMot(T, P) //T est le texte, P est le mot à trouver*
  //$|T|$ est le nombre de caractères de T;
  pour $i = 1..|T| - |P| + 1$ **faire**
    $k = 1$;
    tant que $k \leq |P| \wedge T[i + k - 1] == P[k]$ **faire**
      $k++$;
    fin
    si $k == |P| + 1$ **alors**
      return k;
    fin
  fin
  return nil;

Vous devriez d'abord vous convaincre que cet algorithme fonctionne. Une fois que c'est fait, donnez la complexité de l'algorithme en terme de $|T|$ et de $|P|$ avec la notation O.
Quel est le pire cas, en terme de notation Ω? Ce n'est pas évident: la deuxième boucle sort dès qu'elle ne "match" pas un caractère de P. Pouvez-vous donner un exemple du pire cas?

Exercice 10: Considérez l'algorithme suivant, qui détermine s'il y a une stratégie gagnante pour les blancs aux échecs à partir de la configuration actuelle. Ici, T un tableau 2D 8×8 dont les valeurs indiquent quelle pièce se trouve sur la case, $b = true$ si c'est aux blancs de jouer et $false$ sinon. Aussi, V est la liste des configurations rencontrées jusqu'à maintenant.

```
fonction analyserEchecs(T, b, V)
    si si T est présent au moins trois fois dans V alors
        return false; //selon les règles des échecs, c'est égalité;
    si les blancs sont échec et mat sur T alors
        return false;
    si les noirs sont échec et mat sur T alors
        return true;
    pour chaque mouvement possible m sur T faire
        T' = la configuration obtenue après avoir effectué M;
        ret = analyserEchecs(T', ¬b, V ∪ {T'});
        si ret == true alors
            return ret;
    fin
    return false;
```

Croyez-vous que cet algorithme est correct? Justifiez informellement. Puis, donnez la complexité de cet algorithme avec la notation Θ.
Indice: il n'y a rien à calculer ici.

Exercice 11: (Défi) Montrez que $\sum_{i=1}^{n} i^k \in \Theta(n^{k+1})$ pour toute constante $k \in \mathbb{N}$.

Exercice 12: Montrez que la notation O est transitive. C'est-à-dire, si $f(n) \in O(g(n))$ et $g(n) \in O(h(n))$, alors $f(n) \in O(h(n))$ pour toute fonctions $f, g, h \in \mathcal{F}$. ($\mathcal{F}$ est l'ensemble des fonctions des naturels vers les réels, voir notes de cours)

Exercice 13: Un *arbre binaire plein* est une structure de données constituée de noeuds dans laquelle chaque noeud a soit 2 enfants, soit 0 enfants. Un noeud avec 0 enfants est appelé une *feuille*. Chaque noeud a aussi un parent, excepté la racine qui n'en n'a pas. Vous devriez avoir vu les arbres en structures de données.
La hauteur d'un arbre est le nombre de noeuds entre la racine et son noeud le plus éloigné. Montrez que si un arbre binaire plein a n feuilles, alors sa hauteur est au minimum $\log n$.
Indice: par induction sur la hauteur. Chaque enfant d'un noeud est la racine de son propre sous-arbre. Considérez le sous-arbre le plus feuillu.

Exercice 14: Il y a une erreur dans la preuve qui suit. Quelle est-elle?
Nous allons démontrer que $n^2 \in O(n)$ en prouvant que pour tout $n \geq 1$, il existe une constante c telle que $n^2 \leq cn$.
Case de base: $n = 1$. Dans ce cas, $n^2 = 1 \leq cn$ avec $c = 1$.
Induction: on suppose que pour tout $m < n$, il existe une constante c telle que $m^2 \leq cm$. On a

$$n^2 = (n-1)^2 + 2n - 1 \leq c(n-1) + 2n - 1 = (c+2)n - c - 1 \leq (c+2)n$$

où nous avons utilisé l'hypothèse d'induction pour la première inégalité. Donc il existe une constante $\hat{c} = c + 2$ telle que $n^2 \leq \hat{c}n$, démontrant que $n^2 \in O(n)$.

Exercice 15: Ordonnez les fonctions suivantes avec la notation O, de celle qui grandit le moins rapidement à celle qui grandit le plus rapidement. Il est possible que certaines aient le même taux de croissance, si elle sont Θ l'une de l'autre.

$$n \log n \qquad n^{10} \qquad n^{1.1} \qquad 1.1^n \qquad n^2/\log n \qquad (n^2 + n - 2)^5$$

Le but de cet exercice n'est pas de tout vous faire calculer. Vous pouvez utiliser e.g. wolframalpha pour appliquer les règles des limites.

Exercice 16: Ordonnez les fonctions suivantes avec la notation O, de celle qui grandit le moins rapidement à celle qui grandit le plus rapidement. Il est possible que certaines aient le même taux de croissance, si elle sont Θ l'une de l'autre.

$$n! \qquad (n+1)! \qquad 2^n \qquad 2^{2n} \qquad n^n \qquad n^{\sqrt{n}} \qquad n^{\log n}$$

Le but de cet exercice n'est pas de tout vous faire calculer. Vous pouvez utiliser e.g. wolframalpha pour appliquer les règles des limites.