Chapitre 2
Outils mathématiques pour l'analyse de complexité

HLIN401 : Algorithmique et Complexité

Université de Montpellier
2018 – 2019

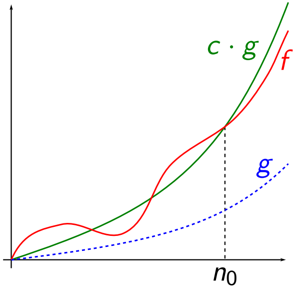
Notations de Landau

« Grand O »

Soit $f, g : \mathbb{N} \to \mathbb{R}_+$. Alors $\boldsymbol{f = O(g)}$ si

$$\exists c > 0, n_0 \geq 0, \forall n \geq n_0, f(n) \leq c \cdot g(n).$$

« f est un grand O de g s'il existe une constante c et un entier n_0 tels que pour toute valeur n plus grande que n_0, $f(n)$ est inférieur ou égal à $c \cdot g(n)$ »



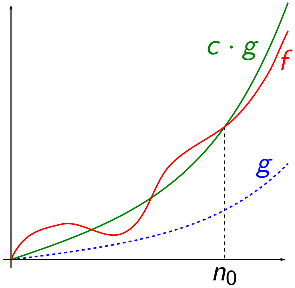
Notations de Landau

« Grand O »

Soit $f, g : \mathbb{N} \to \mathbb{R}_+$. Alors $\boldsymbol{f = O(g)}$ si

$$\exists c > 0, n_0 \geq 0, \forall n \geq n_0, f(n) \leq c \cdot g(n).$$

« f est un grand O de g s'il existe une constante c et un entier n_0 tels que pour toute valeur n plus grande que n_0, $f(n)$ est inférieur ou égal à $c \cdot g(n)$ »



$f = O(g)$ si pour n suffisamment grand, f est plus petite g, à une constante multiplicative près.

Utilisation en complexité

« Mon algo. a une complexité $O(n^2)$ (où $n =$ taille de l'entrée) »

$\rightsquigarrow$ si n est assez grand, le nb. d'opérations est $\leq$ constante$\times n^2$

Utilisation en complexité

« Mon algo. a une complexité $O(n^2)$ (où $n =$ taille de l'entrée) »

$\rightsquigarrow$ si n est assez grand, le nb. d'opérations est $\leq$ constante$\times n^2$

- Avantages pour la théorie :
  - Négliger les cas de bases
  - Pas besoin de compter chaque opération en détail
  - Flexibilité sur les opérations élémentaires

Utilisation en complexité

≪ Mon algo. a une complexité $O(n^2)$ (où $n =$ taille de l'entrée) ≫

$\rightsquigarrow$ si n est assez grand, le nb. d'opérations est $\leq$ constante$\times n^2$

- Avantages pour la théorie :
  - Négliger les cas de bases
  - Pas besoin de compter chaque opération en détail
  - Flexibilité sur les opérations élémentaires
- Avantages pour la pratique :
  - Indépendant des détails de programmation (nb. de variables intermédiaires, ...)
  - Indépendant de l'environnement d'exécution : système d'exploitation, vitesse de la machine, compilateur, ...

Utilisation en complexité

≪ Mon algo. a une complexité $O(n^2)$ (où $n =$ taille de l'entrée) ≫

$\rightsquigarrow$ si n est assez grand, le nb. d'opérations est $\leq$ constante$\times n^2$

- Avantages pour la théorie :
  - Négliger les cas de bases
  - Pas besoin de compter chaque opération en détail
  - Flexibilité sur les opérations élémentaires
- Avantages pour la pratique :
  - Indépendant des détails de programmation (nb. de variables intermédiaires, ...)
  - Indépendant de l'environnement d'exécution : système d'exploitation, vitesse de la machine, compilateur, ...

Un temps de calcul dépend du moment et de l'endroit.
Un résultat de complexité reste vrai **pour toujours !**

Exemples

$5n + 15 = O(n^2)$

- Car pour $n \geq 8$, $5n + 15 \leq n^2$ $\rightsquigarrow c = 1$ et $n_0 = 8$
- Ou alors pour $n \geq 3$, $5n + 15 \leq 5n^2$ $\rightsquigarrow c = 5$ et $n_0 = 3$

Exemples

$5n + 15 = O(n^2)$

- Car pour $n \geq 8$, $5n + 15 \leq n^2$ $\rightsquigarrow c = 1$ et $n_0 = 8$
- Ou alors pour $n \geq 3$, $5n + 15 \leq 5n^2$ $\rightsquigarrow c = 5$ et $n_0 = 3$

```
<inst. 1>;
pour i = 1 à n faire
    <inst. 2>;
pour i = 1 à n faire
    pour j = 1 à n faire
        <inst. 3>;
retourner var
```

- <inst. N> : opérations élémentaires

Exemples

$5n + 15 = O(n^2)$

- Car pour $n \geq 8$, $5n + 15 \leq n^2$ $\quad \rightsquigarrow c = 1$ et $n_0 = 8$
- Ou alors pour $n \geq 3$, $5n + 15 \leq 5n^2$ $\quad \rightsquigarrow c = 5$ et $n_0 = 3$

```
1 <inst. 1>;
2 pour i = 1 à n faire
3   <inst. 2>;
4 pour i = 1 à n faire
5   pour j = 1 à n faire
6     <inst. 3>;
7 retourner var
```

- <inst. N> : opérations élémentaires
- L1 et L7 : $O(1)$
- L2 exécute n fois L3 : $O(n)$
- L5 exécute n fois L6 : $O(n)$
- L4 exécute n fois L5 : $O(n^2)$

Total

$2 \times O(1) + O(n) + O(n^2) = O(n^2)$

Calcul avec les « grand O »

Lemme

- $O(f) + O(g) = O(f+g)$
- $O(f) \times O(g) = O(f \times g)$
- Si $f = O(g)$, alors $O(f+g) = O(g)$
- Si $\lambda \in \mathbb{R}_+$, $O(\lambda f) = O(f)$

Calcul avec les « grand O »

Lemme

- $O(f) + O(g) = O(f + g)$
- $O(f) \times O(g) = O(f \times g)$
- Si $f = O(g)$, alors $O(f + g) = O(g)$
- Si $\lambda \in \mathbb{R}_+$, $O(\lambda f) = O(f)$

Preuve du premier

- Soit $h_1 = O(f)$: $\exists c_1, n_1, \forall n \geq n_1, h_1(n) \leq c_1 f(n)$
- Soit $h_2 = O(g)$: $\exists c_2, n_2, \forall n \geq n_2, h_2(n) \leq c_2 g(n)$

Calcul avec les « grand O »

Lemme

- $O(f) + O(g) = O(f + g)$
- $O(f) \times O(g) = O(f \times g)$
- Si $f = O(g)$, alors $O(f + g) = O(g)$
- Si $\lambda \in \mathbb{R}_+$, $O(\lambda f) = O(f)$

Preuve du premier

- Soit $h_1 = O(f)$: $\exists c_1, n_1, \forall n \geq n_1, h_1(n) \leq c_1 f(n)$
- Soit $h_2 = O(g)$: $\exists c_2, n_2, \forall n \geq n_2, h_2(n) \leq c_2 g(n)$
- Donc $\forall n \geq \max(n_1, n_2)$,

$$\begin{aligned} h_1(n) + h_2(n) &\leq c_1 f(n) + c_2 g(n) \\ &\leq \max(c_1, c_2) f(n) + \max(c_1, c_2) g(n) \\ &\leq \max(c_1, c_2)(f(n) + g(n)) \end{aligned}$$

Calcul avec les « grand O »

Lemme

- $O(f) + O(g) = O(f + g)$
- $O(f) \times O(g) = O(f \times g)$
- Si $f = O(g)$, alors $O(f + g) = O(g)$
- Si $\lambda \in \mathbb{R}_+$, $O(\lambda f) = O(f)$

Preuve du premier

- Soit $h_1 = O(f) : \exists c_1, n_1, \forall n \geq n_1, h_1(n) \leq c_1 f(n)$
- Soit $h_2 = O(g) : \exists c_2, n_2, \forall n \geq n_2, h_2(n) \leq c_2 g(n)$
- Donc $\forall n \geq \max(n_1, n_2)$,

$$\begin{aligned} h_1(n) + h_2(n) &\leq c_1 f(n) + c_2 g(n) \\ &\leq \max(c_1, c_2) f(n) + \max(c_1, c_2) g(n) \\ &\leq \max(c_1, c_2)(f(n) + g(n)) \end{aligned}$$

$\rightsquigarrow h_1 + h_2 = O(f + g)$

≪ Grand O ≫ et limites

Lemme

Soit $g : \mathbb{N} \to \mathbb{R}_+^*$ *(*$g(n) > 0$*). S'il existe une constante* $c \geq 0$ *telle que* $\lim_{n\to+\infty} \left(\frac{f(n)}{g(n)}\right) = c$*, alors* $f = O(g)$*.*

« Grand O » et limites

Lemme

Soit $g : \mathbb{N} \to \mathbb{R}_+^*$ *(*$g(n) > 0$*). S'il existe une constante* $c \geq 0$ *telle que* $\lim_{n\to+\infty}\left(\frac{f(n)}{g(n)}\right) = c$*, alors* $f = O(g)$*.*

Preuve

Si $f(n)/g(n) \longrightarrow_{\infty} c$, alors $f(n)/g(n) \leq c+1$ à partir d'un certain rang. Donc $f(n) \leq (c+1)g(n)$, et $f = O(g)$.

« Grand O » et limites

Lemme

Soit $g : \mathbb{N} \to \mathbb{R}_+^*$ *(*$g(n) > 0$*). S'il existe une constante* $c \geq 0$ *telle que* $\lim_{n\to+\infty}\left(\frac{f(n)}{g(n)}\right) = c$*, alors* $f = O(g)$*.*

Preuve

Si $f(n)/g(n) \longrightarrow_\infty c$, alors $f(n)/g(n) \leq c+1$ à partir d'un certain rang. Donc $f(n) \leq (c+1)g(n)$, et $f = O(g)$.

Lemme

Soit $g : \mathbb{N} \to \mathbb{R}_+^*$ *(*$g(n) > 0$*). Si* $\lim_{n\to+\infty}\left(\frac{f(n)}{g(n)}\right) = +\infty$*, alors* $f \neq O(g)$*.*

≪ Grand O ≫ et limites

Lemme

Soit $g : \mathbb{N} \to \mathbb{R}_+^*$ ($g(n) > 0$). S'il existe une constante $c \geq 0$ telle que $\lim_{n\to+\infty} \left(\frac{f(n)}{g(n)}\right) = c$, alors $f = O(g)$.

Preuve

Si $f(n)/g(n) \longrightarrow_\infty c$, alors $f(n)/g(n) \leq c + 1$ à partir d'un certain rang. Donc $f(n) \leq (c+1)g(n)$, et $f = O(g)$.

Lemme

Soit $g : \mathbb{N} \to \mathbb{R}_+^*$ ($g(n) > 0$). Si $\lim_{n\to+\infty} \left(\frac{f(n)}{g(n)}\right) = +\infty$, alors $f \neq O(g)$.

Preuve

Si $f(n)/g(n) \to_{+\infty} +\infty$, alors pour tout c, $f(n)/g(n) \geq c$ à partir d'un certain rang. Donc $f(n) \geq cg(n)$. Donc aucune constante c ne fonctionne, et $f \neq O(g)$.

« Omega »

Définition

$f = \Omega(g)$ si (au choix !)

- $g = O(f)$

« Omega »

Définition

$f = \Omega(g)$ si (au choix !)

- $g = O(f)$
- $\exists c > 0, n_0 \geq 0, \forall n \geq n_0, f(n) \geq cg(n)$
- « pour n suffisamment grand, f est **supérieure** à g, à une constante multiplicative près »

« Omega »

Définition

$f = \Omega(g)$ si (au choix !)

- $g = O(f)$
- $\exists c > 0, n_0 \geq 0, \forall n \geq n_0, f(n) \geq cg(n)$
- « pour n suffisamment grand, f est **supérieure** à g, à une constante multiplicative près »

Remarque. Utilisé une seule fois dans le cours !

Les fonctions du cours

$$\log n, \quad \sqrt{n} = n^{1/2}, \quad n, \quad n^2, \quad n^k, \quad 2^n, \quad n!$$

Les fonctions du cours

$$\log n, \quad \sqrt{n} = n^{1/2}, \quad n, \quad n^2, \quad n^k, \quad 2^n, \quad n!$$

Lemme de croissance comparée

Pour tout $\alpha, \beta > 0$,

$$\lim_{+\infty} \frac{(\log n)^\alpha}{n^\beta} = 0 \qquad \lim_{+\infty} \frac{n^\alpha}{(2^n)^\beta} = 0 \qquad \lim_{+\infty} \frac{(2^n)^\beta}{n!} = 0$$

Si $\alpha < \beta$:

$$\lim_{+\infty} \frac{(\log n)^\alpha}{(\log n)^\beta} = 0 \qquad \lim_{+\infty} \frac{n^\alpha}{n^\beta} = 0 \qquad \lim_{+\infty} \frac{(2^n)^\alpha}{(2^n)^\beta} = 0$$

Les fonctions du cours

$$\log n, \quad \sqrt{n} = n^{1/2}, \quad n, \quad n^2, \quad n^k, \quad 2^n, \quad n!$$

Lemme de croissance comparée

Pour tout $\alpha, \beta > 0$,

$$\lim_{+\infty} \frac{(\log n)^\alpha}{n^\beta} = 0 \qquad \lim_{+\infty} \frac{n^\alpha}{(2^n)^\beta} = 0 \qquad \lim_{+\infty} \frac{(2^n)^\beta}{n!} = 0$$

Si $\alpha < \beta$:

$$\lim_{+\infty} \frac{(\log n)^\alpha}{(\log n)^\beta} = 0 \qquad \lim_{+\infty} \frac{n^\alpha}{n^\beta} = 0 \qquad \lim_{+\infty} \frac{(2^n)^\alpha}{(2^n)^\beta} = 0$$

Exemples

- $\log^2 n = O(\sqrt{n})$, $n^2 = O(n^3)$, ...
- Mais $\sqrt{n} \neq O(\log^2 n)$, $n^3 \neq O(n^2)$, ...
- $n^2(\log n)^4 = O(n^3)$
- $(\log n)^5 + n(\log n)^2 + n^3 \log n = O(n^3 \log n)$

Calcul avec exponentielles et logarithmes

Sauf mention contraire : log est le logarithme **en base 2**
$\rightsquigarrow$ $\log n$ est environ le nombre de bits de n (exact : $\lfloor \log n \rfloor + 1$)

Calcul avec exponentielles et logarithmes

Sauf mention contraire : log est le logarithme **en base 2**
$\rightsquigarrow$ $\log n$ est environ le nombre de bits de n (exact : $\lfloor \log n \rfloor + 1$)

Règles du log

- $\log 0$ non défini
- $\log 1 = 0$; $\log 2 = 1$
- $\log(a \times b) = \log a + \log b$
- $\log(a/b) = \log a - \log b$
- $\log(a^k) = k \log a$

Calcul avec exponentielles et logarithmes

Sauf mention contraire : log est le logarithme **en base 2**
$\rightsquigarrow$ $\log n$ est environ le nombre de bits de n (exact : $\lfloor \log n \rfloor + 1$)

Règles du log

- $\log 0$ non défini
- $\log 1 = 0$; $\log 2 = 1$
- $\log(a \times b) = \log a + \log b$
- $\log(a/b) = \log a - \log b$
- $\log(a^k) = k \log a$

Règles de l'exponentielle

- $2^0 = 1$; $2^1 = 2$
- $2^{a+b} = 2^a \times 2^b$
- $2^{a-b} = 2^a / 2^b$
- $2^{a \times b} = (2^a)^b = \left(2^b\right)^a$
- $2^{\log a} = \log(2^a) = a$

Calcul avec exponentielles et logarithmes

Sauf mention contraire : log est le logarithme **en base 2**
$\rightsquigarrow$ $\log n$ est environ le nombre de bits de n (exact : $\lfloor \log n \rfloor + 1$)

Règles du log

- $\log 0$ non défini
- $\log 1 = 0$; $\log 2 = 1$
- $\log(a \times b) = \log a + \log b$
- $\log(a/b) = \log a - \log b$
- $\log(a^k) = k \log a$

Règles de l'exponentielle

- $2^0 = 1$; $2^1 = 2$
- $2^{a+b} = 2^a \times 2^b$
- $2^{a-b} = 2^a / 2^b$
- $2^{a \times b} = (2^a)^b = \left(2^b\right)^a$
- $2^{\log a} = \log(2^a) = a$

Exemples

$2^{3n} = (2^3)^n = 8^n$; $n^n = (2^{\log n})^n = 2^{n \log n}$

Autres outils mathématiques

Factorielle et formule de Stirling

$$n! = n \cdot (n-1) \cdot (n-2) \cdots 1 \sim_{n \to +\infty} \sqrt{2\pi n} \left(\frac{n}{e}\right)^n$$

$\rightsquigarrow$ $n! = O(\sqrt{n}(n/e)^n)$ par exemple

Autres outils mathématiques

Factorielle et formule de Stirling

$$n! = n \cdot (n-1) \cdot (n-2) \cdots 1 \sim_{n \to +\infty} \sqrt{2\pi n} \left(\frac{n}{e}\right)^n$$

$\leadsto$ $n! = O(\sqrt{n}(n/e)^n)$ par exemple

Parties entières

- $\lfloor x \rfloor$ est le plus grand entier k tel que $k \leq x$
  (et l'unique entier k tel que $k \leq x < k+1$)

Autres outils mathématiques

Factorielle et formule de Stirling

$$n! = n \cdot (n-1) \cdot (n-2) \cdots 1 \sim_{n \to +\infty} \sqrt{2\pi n} \left(\frac{n}{e}\right)^n$$

$\rightsquigarrow$ $n! = O(\sqrt{n}(n/e)^n)$ par exemple

Parties entières

- $\lfloor x \rfloor$ est le plus grand entier k tel que $k \leq x$
  (et l'unique entier k tel que $k \leq x < k+1$)
- $\lceil x \rceil$ est le plus petit entier k tel que $x \leq k$
  (et l'unique entier k tel que $k - 1 < x \leq k$)

Autres outils mathématiques

Factorielle et formule de Stirling

$$n! = n \cdot (n-1) \cdot (n-2) \cdots 1 \sim_{n \to +\infty} \sqrt{2\pi n} \left(\frac{n}{e}\right)^n$$

$\rightsquigarrow$ $n! = O(\sqrt{n}(n/e)^n)$ par exemple

Parties entières

- $\lfloor x \rfloor$ est le plus grand entier k tel que $k \leq x$
  (et l'unique entier k tel que $k \leq x < k+1$)
- $\lceil x \rceil$ est le plus petit entier k tel que $x \leq k$
  (et l'unique entier k tel que $k-1 < x \leq k$)
- $x - 1 < \lfloor x \rfloor \leq x \leq \lceil x \rceil < x + 1$
- Pour $n \in \mathbb{N}$, $\lfloor \frac{n}{2} \rfloor + \lceil \frac{n}{2} \rceil = n$ (exercice !)

Sommes arithmétique et géométrique

$$\sum_{i=a}^{b} i = (b-a+1) \cdot \frac{b+a}{2} \quad = \text{nb de termes} \times \text{moyenne(min, max)}$$

$$\sum_{i=a}^{b} x^i = x^a \cdot \frac{x^{b-a+1}-1}{x-1} \quad = \frac{x^{b+1}-x^a}{x-1}$$

Sommes arithmétique et géométrique

$$\sum_{i=a}^{b} i = (b-a+1)\cdot\frac{b+a}{2} \qquad = \text{nb de termes} \times \text{moyenne(min, max)}$$

$$\sum_{i=a}^{b} x^i = x^a \cdot \frac{x^{b-a+1}-1}{x-1} \qquad = \frac{x^{b+1}-x^a}{x-1}$$

Exemple

$S \leftarrow 0$;
pour $i = 1$ *à* n **faire**
    $y \leftarrow 1$;
    pour $j = 1$ *à* $\boldsymbol{i}$ **faire**
        $y \leftarrow x \times y$;
    $S \leftarrow S + y$;
retourner S

Sommes arithmétique et géométrique

$$\sum_{i=a}^{b} i = (b-a+1)\cdot\frac{b+a}{2} \qquad = \text{nb de termes} \times \text{moyenne(min, max)}$$

$$\sum_{i=a}^{b} x^i = x^a \cdot \frac{x^{b-a+1}-1}{x-1} \qquad = \frac{x^{b+1}-x^a}{x-1}$$

Exemple

$S \leftarrow 0$;
pour $i = 1$ *à* n **faire**
  $y \leftarrow 1$;
  pour $j = 1$ *à* $\boldsymbol{i}$ **faire**
    $y \leftarrow x \times y$;
  $S \leftarrow S + y$;
retourner S

Complexité :
$\sum_{i=1}^{n} i = \frac{n(n+1)}{2} = O(n^2)$

Sommes arithmétique et géométrique

$$\sum_{i=a}^{b} i = (b-a+1)\cdot\frac{b+a}{2} \qquad = \text{nb de termes} \times \text{moyenne(min, max)}$$

$$\sum_{i=a}^{b} x^i = x^a \cdot \frac{x^{b-a+1}-1}{x-1} \qquad = \frac{x^{b+1}-x^a}{x-1}$$

Exemple

$S \leftarrow 0$;
pour $i = 1$ *à* n **faire**
    $y \leftarrow 1$;
    pour $j = 1$ *à* $\boldsymbol{i}$ **faire**
        $y \leftarrow x \times y$;
    $S \leftarrow S + y$;
retourner S

Complexité :
$\sum_{i=1}^{n} i = \frac{n(n+1)}{2} = O(n^2)$

Valeur : $y = x^i \rightsquigarrow$
$S = \sum_{i=1}^{n} x^i = (x^{n+1}-x)/(x-1)$