Algèbre et Arithmétique Effectives - 17/09/24
Cours 2

La dernière fois

Théorème 1 : Soient $a, b \in \mathbb{Z}$, avec $(a,b) \neq (0,0)$. Alors il existe $d \in \mathbb{Z}_{\geq 0}$ tel que $\text{pgcd}(a,b) = d$.

Théorème 2 : Soient $a, b \in \mathbb{Z}$ $b \neq 0$. Alors il existe un unique couple d'entiers (q,r) tel que

$$a = b\underbrace{q}_{a \text{ quo } b} + \underbrace{r}_{a \bmod b} \quad \text{et} \quad 0 \leq r < |b|.$$

Dém théorème 1

La démonstration est basée sur l'algorithme d'Euclide du calcul du pgcd.

Puisque $d \mid a \Leftrightarrow d \mid -a$, on a $\forall a, b \in \mathbb{Z}$, $(a,b) \neq (0,0)$, $\text{pgcd}(a,b) = \text{pgcd}(|a|, |b|)$.

Donc sans perte de généralité on peut supposer $a, b \geq 0$.

Algorithme d'Euclide

Euclide(a,b)

Entrée : $a, b \in \mathbb{Z}_{\geq 0}$ tels que $(a,b) \neq (0,0)$.

Sortie : $\text{pgcd}(a,b)$

1. Si $a < b$, $(a,b) \leftarrow (b,a)$
2. Tant que $b > 0$:

   $(a,b) \leftarrow (b, a \bmod b)$: division euclidienne
3. Renvoyer a

Quelques rappels de théorie de la complexité

Dans l'analyse d'un algorithme, un des buts est d'estimer le temps de calcul en fonction de la <u>taille</u> de l'entrée.

Pour un entier la taille est le nombre de bits nécessaires pour représenter cet entier en mémoire.

On dénote la taille d'un entier $a \in \mathbb{Z}$ par $len(a)$ (de l'anglais length). On a :

$$len(a) = \begin{cases} \lfloor \log_2 |a| \rfloor + 1, & \text{si } a \neq 0 \\ 0, & \text{si } a = 0 \end{cases}$$

Pour décrire la croissance du temps de calcul on utilise souvent une notation asymptotique.

Soient f, g deux fonctions à valeurs réelles :

- Grand - O (O) (limite supérieure - pire cas)

  $f(n) = O(g(n)) \iff \exists\, c, n_0 > 0$ telles que
  $$|f(n)| \leq c|g(n)| \quad \forall\, n \geq n_0$$

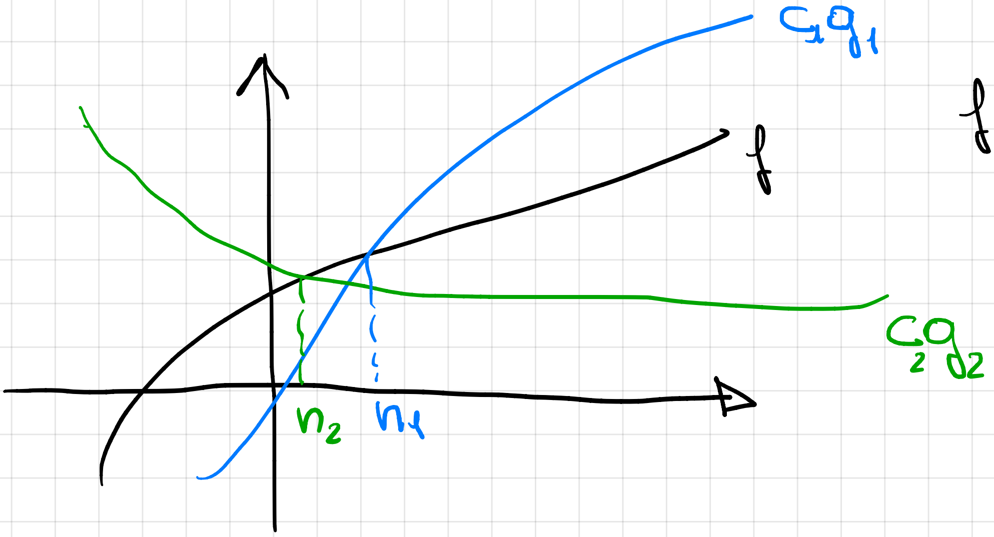
- Grand - Oméga (Ω) (limite inférieure - meilleure cas)

  $f(n) = \Omega(g(n)) \iff \exists\, c, n_0 > 0$ telles que
  $$|f(n)| \geq c\,|g(n)|, \quad \forall\, n \geq n_0$$

- Thêta (Θ) - croissance exacte - cas moyen

  $f(n) = \Theta(g(n)) \iff f(n) = O(g(n))$ et $f(n) = \Omega(g(n))$.

  $\iff \exists\, c_1, c_2, n_0 > 0$ telles que
  $$|g(n)|\, c_1 < |f(n)| < |g(n)| \cdot c_2$$



$$f = \Theta(g_1)$$
$$= \Omega(g_2)$$

$$n = \Theta(n^2) = \Theta(n^3) = \Theta(2^n)$$

<u>Exemple</u>

$\Theta(1)$: temps constant, peu importe la taille de l'entrée

$\Theta(\log(n))$: temps logarithmique

$\Theta(n)$: temps linéaire

$\Theta(2^n)$: temps exponentiel.

<u>Théorème</u> (Chapitre 3.3 de Shoup)

Soient $a, b \in \mathbb{Z}$:

1) On peut calculer $a \pm b$ en $\Theta(\operatorname{len}(a) + \operatorname{len}(b))$

temps linéaire

2) On peut calculer ab en $\Theta(\operatorname{len}(a) \cdot \operatorname{len}(b))$.

3) Si $b \neq 0$ on peut calculer a quo b et a mod b en $\Theta(\operatorname{len}(b) \operatorname{len}(q))$.

<u>Attention</u> : il existe des algorithmes plus rapides pour calculer des multiplications et des divisions.

Coût de l'algorithme d'Euclide

Nous devons estimer :

① Le nombre d'itérations (c'est-à-dire le nombre de divisions euclidiennes)

② Coût de chaque itération (c'est-à-dire le coût de chaque division)

$$\begin{aligned}
& r_0 = a \\
& r_1 = b \\
1)\quad & r_0 = r_1 q_1 + r_2 && (0 < r_2 < r_1) \\
2)\quad & r_1 = r_2 q_2 + r_3 && (0 < r_3 < r_2) \\
& \vdots \\
i)\quad & r_{i-1} = r_i q_i + r_{i+1} && (0 < r_{i+1} < r_i) \\
& \vdots \\
\lambda-1)\quad & r_{\lambda-2} = r_{\lambda-1} q_{\lambda-1} + \underbrace{r_\lambda}_{\mathrm{pgcd}(a,b)} && (0 < r_\lambda < r_{\lambda-1}) \\
\lambda)\quad & r_{\lambda-1} = r_\lambda q_\lambda && (r_{\lambda+1} = 0)
\end{aligned}$$

λ itérations

Combien ça vaut λ au plus ?

__Théorème__ : Soient $a, b \in \mathbb{Z}$ avec $a \geq b > 0$
L'algorithme d'Euclide effectue au plus

$$\frac{\log(b)}{\log(\phi)} + 1 = \Theta(\log(b)) = \Theta(\ln(b)),$$

divisions euclidiennes
où $\phi := \frac{1+\sqrt{5}}{2}$ est le nombre d'or
$(\phi^2 = \phi + 1)$

__Dém__

Puisque $b > 0$, alors $\lambda > 0$ (j'effectue au moins une division)

Si $\lambda = 1$, alors l'énoncé est vrai $\left(\text{car } \frac{\log(b)}{\log(\phi)} > 0\right)$

On montre que $\forall\, i = 0, \dots, \lambda-1$, on a :

$$\boxed{r_{\lambda-i} \geq \phi^i}$$

On procède par récurrence :

initialisation : $i=0 \longrightarrow r_\lambda \geq 1 = \phi^0$ ✓ ($r_\lambda > 0$)

$i = 1 \longrightarrow r_{\lambda-1} \geq r_\lambda + 1 \geq 2 \geq \phi^1$ ✓

$\phi^1 \simeq 1{,}62$

$r_{\lambda-1} > r_\lambda$

Pour $i = 2, \dots, \lambda-1$ on obtient par récurrence :

$$r_{\lambda-i} \geq \underbrace{r_{\lambda-(i-1)}}_{\lambda-i+1} + \underbrace{r_{\lambda-(i-2)}}_{\lambda-i+2} \geq \phi^{i-1} + \phi^{i-2} =$$

$$= \phi^{i-2}(\underbrace{\phi+1}_{\phi^2}) = \phi^i$$

$q > 0$

hypothèse de récurrence

Pour $i = \lambda - 1$ on a :

$$r_1 \geq \phi^{\lambda-1}$$
$$\Updownarrow$$
$$b \geq \phi^{\lambda-1}$$
$$\Downarrow$$
$$\log(b) \geq \log\left(\phi^{\lambda-1}\right)$$
$$\Updownarrow$$
$$\log(b) \geq (\lambda-1)\log(\phi) \iff \boxed{\lambda \leq \frac{\log(b)}{\log(\phi)} + 1}$$

$\square$

À chaque itération le temps de calcul est $\Theta(\mathrm{len}(r_i)\,\mathrm{len}(q_i))$

On peut montrer que le coût total est :

$$T = \sum_{i=1}^{\lambda} \mathrm{len}(r_i)\,\mathrm{len}(q_i) \underset{\substack{\uparrow \\ \text{Théorème 4.2} \\ \text{Shoup}}}{=} \Theta(\mathrm{len}(a)\,\mathrm{len}(b))$$

<u>Théorème</u> : Le coût de calcul de Euclide(a,b) est $\Theta(\mathrm{len}(a)\,\mathrm{len}(b))$.

Il existe une variante de cet algorithme.

Algorithme binaire du pgcd : utilise seulement des soustractions et des multiplications ou divisions par 2 (⟶ simple décalage de bits)

Il se base aussi sur les propriétés suivantes :

- si les deux entiers sont pairs :

$$PGCD(2a, 2b) = 2PGCD(a,b)$$

- si un entier est pair et l'autre impair :

$$PGCD(2a, \underset{\substack{\uparrow \\ \text{impair}}}{b}) = PGCD(a,b)$$

- si les deux entiers sont impairs :

$$PGCD(a,b) = PGCD(a-b,b).$$

Algorithme d'Euclide Étendu

Théorème (identité de Bézout)

Soient $a, b \in \mathbb{Z}$, $(a,b) \neq (0,0)$, et soit $d = pgcd(a,b)$

Alors il existe un couple d'entiers (u,v) tels que

$$\boxed{au + bv = d}$$

identité de Bézout.

Exemple :

$a = 87$, $b = 24$

$$87 = 24 \cdot 3 + 15$$
$$24 = 15 \cdot 1 + 9$$
$$15 = 9 \cdot 1 + 6$$
$$\boxed{9 = 6 \cdot 1 + 3} = pgcd(87, 24)$$
$$6 = 3 \cdot 2 + 0$$

$$3 = 9 - 6 \cdot 1 = 9 - (15 - 9) =$$
$$= 9 \cdot 2 - 15 = (24 - 15) \cdot 2 - 15 =$$
$$= 24 \cdot 2 - 15 \cdot 3 = 24 \cdot 2 - (87 - 24 \cdot 3) \cdot 3 =$$
$$= \underbrace{-3}_{u} \cdot \underbrace{87}_{a} + \underbrace{11}_{v} \cdot \underbrace{24}_{b}$$

TD 1

Exo 2 :

1) $\mathrm{pgcd}(13, 21) = 1$

$1 = 5 \cdot 21 - 8 \cdot 13$

$= -8 \cdot 21 + 13 \cdot 13$

2) $\mathrm{pgcd}(2926, 2046) = 22$

$22 = 7 \cdot 2926 - 10 \cdot 2046$.

Exo 4

Soient $a, b \in \mathbb{Z}$, $(a,b) \neq (0,0)$. Soit $d = \mathrm{pgcd}(a,b)$

1) S'il existe $s, t \in \mathbb{Z}$ tels que $as + bt = r$, alors $d \mid r$.

Puisque $d = \mathrm{pgcd}(a,b) \Rightarrow d \mid a$ et $d \mid b \Rightarrow$
$\Rightarrow \exists\, k \in \mathbb{Z}$ tel que $a = dk$ et $\exists\, k' \in \mathbb{Z}$ tel que $b = dk'$.

Donc en remplaçant on obtient :

$$dk \cdot s + dk't = r$$
$$\Updownarrow$$
$$d(\underbrace{ks + k't}_{\in \mathbb{Z}}) = r$$
$$\Downarrow$$
$$d \mid r.$$

Corollaire : S'il existe $s, t \in \mathbb{Z}$ tels que $as + bt = 1 \Rightarrow \mathrm{pgcd}(a,b) \mid 1 \Rightarrow \mathrm{pgcd}(a,b) = 1$

2) Soient $u,v \in \mathbb{Z}$ tels que $d = au + bv$, alors $\mathrm{pgcd}(u,v) = 1$.

<u>Méthode 1</u> : <u>Par l'absurde</u>, on suppose $\mathrm{pgcd}(u,v) = d' > 1$

$\Rightarrow d' \mid u$ et $d' \mid v \Rightarrow \exists\, k, k' \in \mathbb{Z}$ tels que

$u = d'k$ et $v = d'k'$

Donc en remplaçant dans $d = au + bv$, on obtient :

$$d = ad'k + bd'k' = d'(ak + bk') \Rightarrow d' \mid d$$

$\Rightarrow \exists\, z \in \mathbb{Z}$ tel que $d = d'z$ $(z < d)$

Donc $\cancel{d'}z = \cancel{d'}(ak + bk') \Rightarrow z = ak + bk'$

$\Rightarrow d = \mathrm{pgcd}(a,b) \mid z$ ↯ car $z < d$

<u>Méthode 2</u> : $d = \mathrm{pgcd}(a,b) \Rightarrow \exists\, x, x' \in \mathbb{Z}$ t. q.

$dx = a$ et $dx' = b$

Donc :

$$d = au + bv = dxu + dx'v$$

$$\Downarrow d \neq 0$$

$$1 = xu + x'v$$

$$\Downarrow \text{ Corollaire}$$

$$\mathrm{pgcd}(u,v) = 1.$$

Algorithme (Euclide Étendu)

EuclideEtendu(a,b)

<u>Entrées</u> : $a, b \in \mathbb{Z}_{\geq 0}$, $(a,b) \neq (0,0)$

<u>Sortie</u> : un triplet (d,u,v) tel que $\text{pgcd}(a,b) = d$
et $d = au + bv$.

1. si $a < b$:
2. $\quad (d,u,v) \leftarrow$ EuclideEtendu(b,a)
3. $\quad$ Renvoyer (d,v,u)
4. Si $b = 0$: Renvoyer $(a,1,0)$
5. $(q,r) \leftarrow (a \text{ quo } b, \ a \bmod b)$
6. $(d,u',v') \leftarrow$ EuclideEtendu(b,r)
7. Renvoyer $(d, v', u' - qv')$