

# Calculabilité et complexité

## Cours n° 2

**Nicolas (Miki) Hermann**

LIX, École Polytechnique

hermann@lix.polytechnique.fr

# Hiérarchie des classes de complexité

Soit  $f$  une fonction de complexité. Définissons le langage

$$H_f = \{(M, x) \mid M \text{ accepte } x \text{ après au plus } f(|x|) \text{ pas}\}$$

où  $M$  parcourt toutes les machines de Turing déterministes (multi-bande ou pas)

## Théorème

$$H_f \in \text{DTIME}(f(n)^3)$$

## Idée de la preuve

Construire une machine de Turing déterministe universelle  $U_f$  qui simule  $M$ , avec “truc” supplémentaire, la **sonette d'alarme** (implantée comme un mot sur un ruban), qui fait rejeter automatiquement chaque mot  $x$  qui n'est pas accepté par  $M$  en temps  $f(|x|)$ . La machine  $U$

- 1 construit la sonette d'alarme  $\square^{f(|x|)}$  pour  $x$  sur un ruban en temps  $O(f(|x|))$ ,
- 2 simule chaque pas du travail de  $M$  en temps  $O(f(|x|)^2)$ ,
- 3 accepte ou rejette le mot  $x$  (s'il le faut) en temps  $O(f(|x|)^3)$ . □

## Théorème

$$H_f \notin \text{DTIME}(f(\lfloor n/2 \rfloor))$$

## Preuve par diagonalisation

Supposons qu'il existe une machine de Turing déterministe  $M_f$  qui décide  $H_f$  en temps  $f(\lfloor n/2 \rfloor)$ . Construisons une machine de Turing déterministe diagonalisante  $D_f$ , dont le programme sera

$$D_f(M) = \text{if } M_f(M, M) == \text{YES then NO else YES fi}$$

$D_f$  utilise le mot  $M$  autant de fois que la machine  $M_f$  sur le mot  $(M, M)$ , i.e.,  $f(\lfloor (2n + 1)/2 \rfloor) = f(n)$ .

...

## Preuve par diagonalisation (cont)

Est-ce que  $D_f$  accepte sa propre description ?

- Si  $D_f(D_f) = \text{YES}$  alors  $M_f(D_f, D_f) = \text{NO}$  impliquant  $(D_f, D_f) \notin H_f$ . Ceci veut dire que  $D_f$  n'accepte pas sa propre définition en temps  $f(n)$ . Ceci implique  $D_f(D_f) = \text{NO}$ . **Contradiction.**
- Si  $D_f(D_f) = \text{NO}$ , ceci implique  $M_f(D_f, D_f) = \text{YES}$ , puis  $(D_f, D_f) \in H_f$ , ce qui donne  $D_f(D_f) = \text{YES}$ . **Contradiction.** □

# Hierarchie des classes de complexité

## Théorème [Hiérarchie du temps]

Si  $f(x) \geq n$  est une fonction de complexité, alors

$$\text{DTIME}(f(n)) \subsetneq \text{DTIME}(f(2n + 1)^3)$$

## Preuve

Les deux théorèmes précédents. □

# Hiérarchie des classes de complexité

## Théorème [Hiérarchie du temps]

Si  $f(x) \geq n$  est une fonction de complexité, alors

$$\text{DTIME}(f(n)) \subsetneq \text{DTIME}(f(2n + 1)^3)$$

## Preuve

Les deux théorèmes précédents. □

Il existe un meilleur résultat.

## Théorème

Si  $f(n) \geq n$  est une fonction de complexité, alors

$$\text{DTIME}(f(n)) \subsetneq \text{DTIME}(f(n)(\log f(n))^2)$$

Corollaire

$P \subsetneq EXP$

# Hiérarchie des classes de complexité

## Corollaire

$$P \subsetneq EXP$$

## Preuve

Chaque polynôme  $p(n)$  devient plus petit que  $2^n$  pour un  $n$  suffisamment grand. Alors,

$$P \subseteq DTIME(2^n) \subseteq EXP$$

Mais on sait que

$$P \subseteq DTIME(2^n) \subsetneq DTIME((2^{2n+1})^3) \subseteq DTIME(2^{n^2}) \subseteq EXP \quad \square$$

# Hierarchie des classes de complexité

On connaît un théorème similaire pour l'espace.

## Théorème [Hiérarchie de l'espace]

Soit  $f(n)$  une fonction de complexité. Alors

$$\text{DSPACE}(f(n)) \subsetneq \text{DSPACE}(f(n) \log f(n))$$

# Hiérarchie des classes de complexité

On connaît un théorème similaire pour l'espace.

## Théorème [Hiérarchie de l'espace]

Soit  $f(n)$  une fonction de complexité. Alors

$$\text{DSPACE}(f(n)) \subsetneq \text{DSPACE}(f(n) \log f(n))$$

## Corollaire

$$L \subsetneq \text{PSPACE}$$

## Théorème

Soit  $f(n)$  une fonction de complexité. Alors

- 1  $\text{DTIME}(f(n)) \subseteq \text{NTIME}(f(n))$
- 2  $\text{DSPACE}(f(n)) \subseteq \text{NSPACE}(f(n))$
- 3  $\text{NTIME}(f(n)) \subseteq \text{DSPACE}(f(n))$
- 4  $\text{NSPACE}(f(n)) \subseteq \text{DTIME}(k^{\log n + f(n)})$

Les deux premières inclusions sont triviales.

## Preuve de l'inclusion 3

Soit  $L \in \text{NTIME}(f(n))$ . Il existe une machine de Turing **non-déterministe**  $M$  telle que  $L = L(M)$ . La machine fait  $f(n)$  pas sur un mot de longueur  $n$ . La machine **déterministe**  $M'$  correspondant, simulant  $M$ , essaie consécutivement toutes les alternatives pour les choix de  $M$ , avec les retours en arrière, jusqu'à ce qu'elle trouve la bonne. Pour chaque alternative il lui faut effectuer  $f(n)$  pas. Si une alternative est rejetée, la machine  $M'$  **réutilise** le même espace pour tester l'alternative suivante. Par conséquent, il suffit un espace  $f(n)$  pour simuler  $M$  par  $M'$ . Donc  $L(M) \in \text{DSPACE}(f(n))$ . □

## Preuve de l'inclusion 4

Soit  $M$  une machine **non-déterministe** avec  $k$  rubans qui décide  $L$  en espace  $f(n)$ , i.e.,  $L = L(M)$  et  $L \in \text{NSPACE}(f(n))$ . Il nous faut trouver une méthode **déterministe** pour simuler le calcul de  $M$  sur  $x$  en temps  $c^{\log n + f(n)}$  où  $n = |x|$  pour une constante  $c$  qui dépend de  $M$ . Soit

$$(q, w_1, u_1, \dots, w_k, u_k)$$

la configuration de  $M$ . Car le ruban d'entrée reste inchangé, on a toujours  $w_1 = \triangleright$  et  $u_1 = x$ . Le ruban de sortie n'est pas intéressant non plus. Les rubans  $2, \dots, k-2$  auront la longueur bornée par  $f(n)$ . Donc la configuration est représentée par le tuple

$$(q, i, w_2, u_2, \dots, w_{k-1}, u_{k-1})$$

où  $i$  est un entier avec  $1 \leq i \leq n + 1 = |x|$ , indiquant la position du curseur de lecture sur le ruban d'entrée et  $|w_j u_j| \leq f(n)$  pour chaque  $j = 2, \dots, k-1$ .

## Preuve de l'inclusion 4 (cont 1)

Combien de configuration existe-t-il ? Il existe  $|Q|$  choix pour l'état  $q$ ,  $n + 1$  choix pour  $i$  et  $|\Gamma|^{(k-1)f(n)}$  choix pour les rubans de travail. En gros, nous avons au plus

$$|Q| \cdot (n + 1) \cdot |\Gamma|^{(k-1)f(n)} = nc_1^{f(n)} = c_1^{\log n + f(n)}$$

configurations de  $M$  travaillant sur un mot de longueur  $n$ , pour une constante  $c_1$  dépendant de  $M$ .

## Preuve de l'inclusion 4 (cont 2)

Définissons le **graphe de configuration** de  $M$  travaillant sur le mot  $x$ , noté  $G(M, x)$ . Les sommets sont toutes les configurations possibles du calcul de  $M$  sur  $x$ . Il existe une arrête entre deux configurations  $C_1$  et  $C_2$  si et seulement si  $C_1 \vdash_M C_2$ . Il est claire que **décider**  $x \in L$  équivaut de dire s'il existe le **chemin** entre la configuration **initiale** (START, 1,  $\epsilon, \dots, \epsilon$ ) et une configuration **acceptante** (YES,  $i, \dots$ ).

**Problème:** REACHABILITY

*Entrée:* Un graphe  $G = (V, E)$  avec  $|V| = n$  et  $s, t \in V$ .

*Question:* Existe-t-il un chemin entre les sommets  $s$  et  $t$  dans  $G$  ?

## Théorème

Le problème REACHABILITY est décidable en temps  $O(n^2)$ .

## Preuve de l'inclusion 4 (cont 3)

Dans notre cas, on cherche le chemin dans le graphe  $G(M, x)$  possédant  $c_1^{\log n + f(n)}$  sommets. On trouve en principe ce chemin en temps  $c_2 c_1^{2(\log n + f(n))}$ . Si on prend  $k = c_2 c_1^2$ , on arrive à la borne souhaitée.

Il faut **construire** le graphe  $G(M, x)$  au fur et à mesure, sommet par sommet, arrête par arrête, à partir de la configuration initiale. □

## Corollaire

$$L \subseteq NL \subseteq P \subseteq NP \subseteq \text{PSPACE}$$

## Corollaire

$$L \subseteq NL \subseteq P \subseteq NP \subseteq \text{PSPACE}$$

## Frustration de la théorie de complexité

On sait que  $L \subsetneq \text{PSPACE}$ , mais on n'arrive pas à décider laquelle des inclusions dans le corollaire précédent est propre.

L'espace est une unité **réutilisable**. Est-ce que d'avantage d'espace ne dépassant pas la borne polynomiale n'arrive pas à absorber l'effet du non-déterminisme ?

L'espace est une unité **réutilisable**. Est-ce que d'avantage d'espace ne dépassant pas la borne polynomiale n'arrive pas à absorber l'effet du non-déterminisme ?

**Réponse : OUI**, on va le démontrer.

## Théorème [Savitch]

$\text{REACHABILITY} \in \text{DSPACE}(\log^2 n)$

## Théorème [Savitch]

$$\text{REACHABILITY} \in \text{DSPACE}(\log^2 n)$$

### Preuve :

Soit  $G$  un graphe avec  $n$  sommets, soit  $x$  et  $y$  deux sommets de  $G$ .  
Construisons le prédicat

$$\text{CHEMIN}(x, y, i) = \text{il existe un chemin } p \text{ de } x \text{ à } y \text{ avec } |p| \leq 2^i$$

Il y a  $n$  sommets dans  $G$ , alors il y a à la limite  $\text{CHEMIN}(x, y, \lceil \log n \rceil)$   
pour chaque  $x, y \in V$ .

Machine de Turing déterministe  $M$  avec 4 rubans :

- ruban d'entrée : matrice d'adjacence de  $G$
- 2 rubans de travail
  - 1<sup>er</sup> ruban : triples  $(x, y, i)$  en binaire
  - 2<sup>e</sup> ruban :  $O(\log n)$  espace pour résultats intermédiaires

## Preuve du théorème de Savitch (cont)

```

CHEMIN( $x, y, i$ ) = if  $i == 0$  then return( $(x \equiv y) \vee (x, y) \in E$ )
                   else
                     forall  $z \in V$  do
                       if CHEMIN( $x, z, i - 1$ )  $\wedge$  CHEMIN( $z, y, i - 1$ )
                         then return(YES) fi
                     od
                   return(NO)
                   fi

```

Ruban 1 = pile : espace  $\lceil \log n \rceil \cdot 3 \log n = O(\log^2 n)$

Ruban 2 = gestion des résultats intermédiaires : espace  $O(\log n)$

REACHABILITY possède une solution si et seulement si  
 CHEMIN( $s, t, \lceil \log n \rceil$ ) = *true*. □

## Corollaire 1

$$\text{NSPACE}(f(n)) \subseteq \text{DSPACE}(f^2(n))$$

pour chaque fonction de complexité  $f(n) \geq \log n$ .

## Corollaire 1

$$\text{NSPACE}(f(n)) \subseteq \text{DSPACE}(f^2(n))$$

pour chaque fonction de complexité  $f(n) \geq \log n$ .

## Preuve :

Chaque machine de Turing non-déterministe  $M$  avec  $L(M) \in \text{NSPACE}(f(n))$  et chaque mot  $x$  avec  $|x| = n$  peut être décrit par un graphe de configuration  $G(M, x)$  avec  $O(c^{f(n)})$  sommets. Utiliser le théorème de Savitch avec  $O(\log^2 c^{f(n)}) = O(f^2(n))$ . □

## Corollaire 2

$$\text{NPSPACE} = \text{PSPACE}$$

## Corollaire 2

$$\text{NPSPACE} = \text{PSPACE}$$

**Question** : Quid de **NL** et **coNL** ?