Calculabilité & Complexité

Complexité (1/4) : reboot

Nicolas Ollinger (LIFO, Université d'Orléans)

M1 info, Université d'Orléans — S2 2025/2026

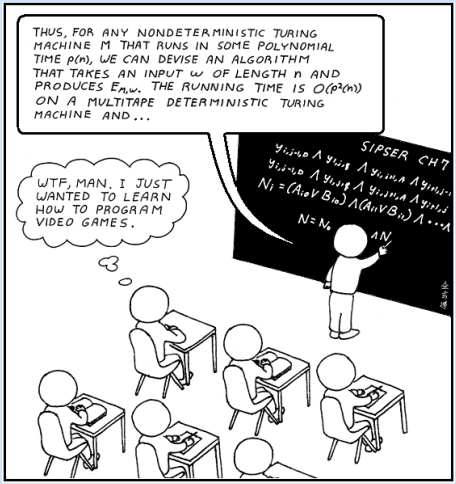
Objet du cours

Partie I. Calculabilité

« Que peut-on calculer avec un ordinateur ? »

Partie II. Complexité

*« Que peut-on calculer **efficacement** avec un ordinateur ? »*



2. Complexité

Théorie de la complexité

Le but de la **théorie de la complexité** est de prendre en compte les limitations qui rendent certaines solutions inaccessibles en pratique, bien que **calculables**.

Dans un premier temps, on s'intéressera aux ressources en **temps** et **espace** de calcul.

Algorithmique *Étant donné un problème, on cherche un algorithme le plus efficace possible pour le résoudre.*

Complexité *Étant donné un problème, on cherche une borne sur la complexité d'un meilleur algorithme pour le résoudre.*

Thèse de Church-Turing

Thèse de Church-Turing Toute fonction calculable par une méthode effective est Turing-calculable.

Thèse de Church-Turing étendue

Thèse de Church-Turing Toute fonction calculable par une méthode effective est Turing-calculable.

Thèse de Church-Turing étendue Tous les modèles de calcul **classiques** raisonnables ont une même complexité à un facteur polynomial près.

Thèse de Church-Turing étendue

Thèse de Church-Turing Toute fonction calculable par une méthode effective est Turing-calculable.

Thèse de Church-Turing étendue Tous les modèles de calcul **classiques** raisonnables ont une même complexité à un facteur polynomial près.

Attention, on ne mesure pas toujours la taille de l'entrée de la même manière en **algorithmique** et en **complexité** !

Modèle de complexité : MT à k rubans

La **machine de Turing** classique est enrichie : un **contrôle fini** couplé à $k \geq 2$ **rubans** biinfinis munis chacun d'une **tête** d'E/S mobile pointant sur une cellule. Parmi ces rubans, un **ruban d'entrée** en lecture seule et un **ruban de sortie** en écriture seule.

$$\delta(q, \underbrace{a_0, a_1, \ldots, a_{k-2}}_{\text{lecture sur } k-1 \text{ rubans}}) = (q', \underbrace{b_1, b_2, \ldots, b_{k-1}}_{\text{écriture sur } k-1 \text{ rubans}}, \underbrace{\Delta_0, \ldots, \Delta_{k-1}}_{k \text{ déplacements}})$$

Formellement

Définition Une **MT à k-rubans** est un tuple $(Q, \Gamma, \Sigma, \delta, q_0, B, q_a, q_r)$ où

- Q est l'ensemble fini des **états** ;
- Γ est l'**alphabet** fini de **travail** ;
- $\Sigma \subseteq \Gamma$ est l'**alphabet** fini **d'entrée** ;
- $\delta : (Q \setminus \{q_a, q_r\}) \times \Gamma^{k-1} \to Q \times \Gamma^{k-1} \times \{\blacktriangleleft, \blacktriangledown, \blacktriangleright\}^k$ est la **fonction de transition** ;
- $q_0 \in Q$ est l'**état initial** ;
- $B \in \Gamma \setminus \Sigma$ est le **symbole blanc** ;
- $q_a \in Q$ est l'**état d'acceptation** de la machine ;
- $q_r \in Q$ est l'**état de rejet** de la machine.

Exercice

Construire une MT à 2 rubans qui teste **efficacement** (en $O(3n)$ étapes de calcul) si un mot $u \in \{a, b\}^*$ (de longueur n) est un **palindrome**.

Remarque Avec un **unique** ruban, il n'est pas possible d'être **aussi efficace**. Sauriez-vous le démontrer ?

Mesurer le temps et l'espace

Temps de calcul

$t_{\mathcal{M}}(u)$ = nombre de pas de calcul avant arrêt pour $\mathcal{M}$ sur l'entrée u

$$t_{\mathcal{M}}(n) = \max_{u \in \Sigma^n} t_{\mathcal{M}}(u) \qquad \forall n \in \mathbb{N}$$

Espace de calcul

$s_{\mathcal{M}}(u)$ = nombre total de cases des rubans de travail visitées

$$s_{\mathcal{M}}(n) = \max_{u \in \Sigma^n} s_{\mathcal{M}}(u) \qquad \forall n \in \mathbb{N}$$

Proposition Pour toute k-MT $\mathcal{M}$ il existe un α tel que

$$\begin{cases} s_{\mathcal{M}}(u) \leqslant & (k-1)t_{\mathcal{M}}(u) \\ t_{\mathcal{M}}(u) \leqslant & 2^{\alpha(s_{\mathcal{M}}(u)+|u|)} \end{cases} \qquad \forall u \in \Sigma^*$$

Des tas de rubans

Proposition Toute MT $\mathcal{M}$ à $k \geqslant 2$ rubans est équivalente à une MT $\mathcal{M}'$ à 2 **rubans** telle que

$$t_{\mathcal{M}'}(n) \leqslant t_{\mathcal{M}}(n) \log t_{\mathcal{M}}(n) \qquad \forall n \in \mathbb{N}$$

Remarque La borne proposée demande des constructions subtiles. On pourra commencer par une version naïve avec

$$t_{\mathcal{M}'}(n) \leqslant t_{\mathcal{M}}(n)^2 \qquad \forall n \in \mathbb{N}$$

Classes de complexité en temps déterministe

Hartmanis et Stearns 1965

Définition Pour toute fonction $f : \mathbb{N} \to \mathbb{N}$, la classe $\mathrm{DTIME}(f(n))$ est l'ensemble des langages reconnus par une MT $\mathcal{M}$ en temps borné par $\alpha f(n)$ pour un certain α.

$$\forall u \in \Sigma^* \qquad t_{\mathcal{M}}(u) \leqslant \alpha f(|u|)$$

Définition Si $\mathcal{F}$ est un ensemble de fonctions on note

$$\mathrm{DTIME}(\mathcal{F}) = \bigcup_{f \in \mathcal{F}} \mathrm{DTIME}(f(n))$$

$$\mathrm{P} = \mathrm{DTIME}\left(\left\{n^k \,\middle|\, k \in \mathbb{N}\right\}\right)$$

Premières propriétés

Proposition Si $f(n) \leqslant g(n)$ pour tout $n \in \mathbb{N}$ alors
$$\mathrm{DTIME}(f(n)) \subseteq \mathrm{DTIME}(g(n)).$$

Proposition Si $f(n) \geqslant n$ pour tout $n \in \mathbb{N}$ alors $\mathrm{DTIME}(f(n))$ est clos par union finie, intersection finie et complémentaire.

Théorème[accélération] Pour toute fonction $f : \mathbb{N} \to \mathbb{N}$ et tout $\varepsilon > 0$, si L est reconnu en temps $f(n)$ alors il existe une MT $\mathcal{M}'$ qui reconnaît L en temps $(1 + \varepsilon)n + \varepsilon f(n)$.

Théorème de hiérarchie

Définition Une fonction $f : \mathbb{N} \to \mathbb{N}$ est **constructible en temps** s'il existe une constante α et une MT qui sur l'entrée 1^n (l'entier n en unaire) renvoie $1^{f(n)}$ (l'entier $f(n)$ en unaire) en temps $\leqslant \alpha f(n)$.

Théorème[H&S 65] Pour toute fonction f **constructible en temps**

$$\mathrm{DTIME}(f(n)) \subsetneq \mathrm{DTIME}(nf(n)^2)$$

Première séparation

$$\begin{aligned} \mathrm{P} &= \bigcup_{k\in\mathbb{N}} \mathrm{DTIME}(n^k) \\ \mathrm{EXP} &= \bigcup_{k\in\mathbb{N}} \mathrm{DTIME}(2^{n^k}) \end{aligned}$$

Corollaire $\mathrm{P} \subsetneq \mathrm{EXP}$

Idée $\mathrm{P} \subseteq \mathrm{DTIME}(n^{\log n}) \subsetneq \mathrm{DTIME}(n^{1+\log^2 n}) \subseteq \mathrm{DTIME}(2^n) \subseteq \mathrm{EXP}$