Complexité

M1 Informatique Luminy 2024-25

Kévin Perrot	<kevin.perrot@univ-amu.fr>
Pablo Concha-Vega	<pablo.concha-vega@lis-lab.fr>
Antonio E. Porreca	<antonio.porreca@univ-amu.fr>
Mathieu Roget	<mathieu.roget@univ-amu.fr>

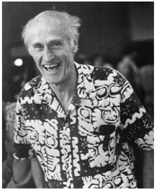
9hCM 9hTD 9hTP

NF=MAX(ET ; 0.3*CC+0.7*ET)

Résolution algorithmique de problèmes

« *An algorithm is a finite answer to an infinite number of questions* »

Stephen Kleene

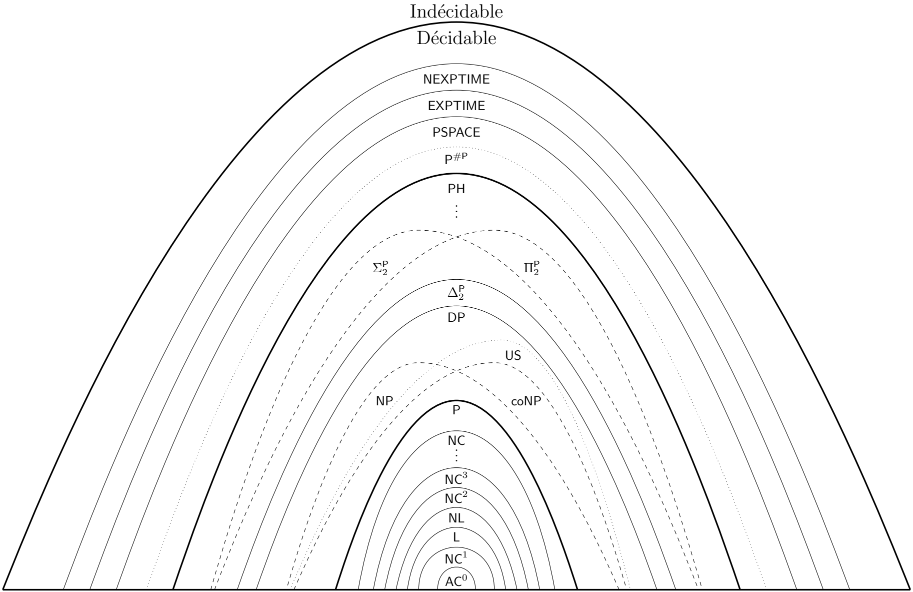


crédit photo ?

Recherce d'un mot dans un texte
Entrée : deux chaînes de caractères m et t.
Question : est-ce que m a au moins une occurrence dans t ?

Test de primalité
Entrée : un entier n.
Question : est-ce n est un nombre premier ?

Calculabilité et complexité des problèmes



Complexité algorithmique

La complexité algorithmique mesure la quantité de ressources (temps, mémoire) utilisées pour la résolution d'un problème par un programme.

- Fonction de la taille de l'entrée.
- Dans le pire cas (offre une garantie)
  ou en moyenne (attention, l'analyse est plus difficile).

Recherche d'un mot dans un texte Entrée : deux chaînes de caractères m et t. Question : est-ce que m a au moins une occurrence dans t ?

? Temps linéaire : $\mathcal{O}(|m| + |t|)$ Knuth–Morris–Pratt (1970)

Test de primalité Entrée : un entier n. Question : est-ce n est un nombre premier ?

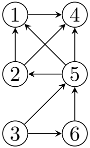
? Temps polynomial : $\tilde{\mathcal{O}}(\log(n)^6)$ Agrawal–Kayal–Saxena (2002)

Rappels : taille de l'entrée

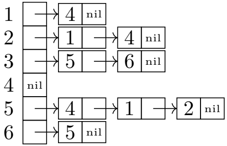
Type de donnée	**Taille en bits (environ)**
un entier naturel $n \in \mathbb{N}$	$\log_2(n) \qquad \lfloor \log_2(n) \rfloor + 1$
un entier relatif $n \in \mathbb{Z}$	$\log_2(n) \qquad \lfloor \log_2(n) \rfloor + 2$
un réel $r \in \mathbb{R}$	constante $(1, 8, 23 \text{ ou } 1, 11, 52)$
un tableau de n entiers	$c\, n$ pour des entiers de 0 à $2^c - 1$
une chaîne de caractères s	$c\, \lvert s \rvert$ pour un alphabet de $\leq 2^c$ lettres
un graphe avec	n^2 par matrice d'adjacence
n sommets et m arrêtes	$n + m$ par listes d'adjacence

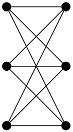
Parfums de graphes

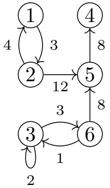
- Orienté ou non
- Pondéré ou non
- Simple ou multiple
- Acyclique ou non
- Biparti ou non
- Planaire ou non
- Connexe ou non
- Topologique ou plongé
- Creux ou dense

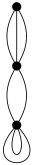


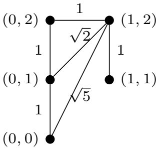
$$\begin{pmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$









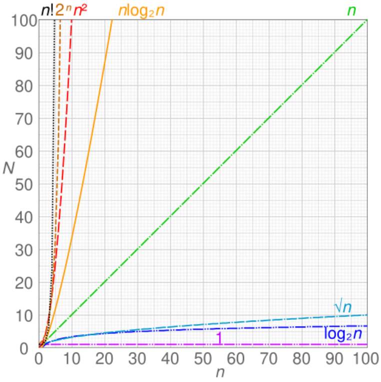


Rappels : notations de Landau

Bornes asymtotiques à un facteur multiplicatif près :

- On note $f(n) \in \mathcal{O}(g(n))$ lorsque $\exists c > 0 : \exists n_0 : \forall n \geq n_0 : f(n) \leq c\, g(n)$.
- On note $f(n) \in \Omega(g(n))$ lorsque $\exists c > 0 : \exists n_0 : \forall n \geq n_0 : f(n) \geq c\, g(n)$.
- On note $f(n) \in \Theta(g(n))$ lorsque $f(n) \in \mathcal{O}(g(n))$ et $f(n) \in \Omega(g(n))$.
- On note $f(n) \in o(g(n))$ lorsque $\forall \epsilon > 0 : \exists n_0 : \forall n \geq n_0 : f(n) \leq \epsilon\, g(n)$.

$\mathcal{O}(1)$	constant
$\mathcal{O}(\log n)$	logarithmique
$\mathcal{O}(n)$	linéaire
$\mathcal{O}(n \log n)$	quasi-linéaire
$\mathcal{O}(n^2)$	quadratique
$\mathcal{O}(n^3)$	cubique
$\mathcal{O}(n^k)$ avec k fixé	polynomial
$\mathcal{O}(k^n)$ avec $k > 1$ fixé	exponentiel
$\mathcal{O}(n!)$	factoriel



Rappels : ordres de grandeur en temps

1 million d'instructions élémentaires par seconde :

	n	$n \log n$	n^2	n^3	1.5^n	2^n	$n!$
$n = 10$	0	0	0	0	0	0	4 sec
$n = 20$	0	0	0	0	0	1 sec	77000 ans
$n = 50$	0	0	0	0	11 min	36 ans	∞
$n = 100$	0	0	0	1 sec	12891 ans	10^{17} ans	∞
$n = 1000$	0	0	1 sec	17 min	∞	∞	∞
$n = 10^4$	0	0	2 min	12 jours	∞	∞	∞
$n = 10^5$	0	2 sec	3 heures	32 ans	∞	∞	∞
$n = 10^6$	1 sec	20 sec	12 jours	32000 ans	∞	∞	∞

$0 =$ moins de 1 seconde
$\infty =$ plus de 10^{20} ans

Rappels : ordres de grandeur en temps

1 milliard d'instructions élémentaires par seconde :

	n	$n \log n$	n^2	n^3	1.5^n	2^n	$n!$
$n = 10$	0	0	0	0	0	0	0
$n = 20$	0	0	0	0	0	0	77 ans
$n = 50$	0	0	0	0	1 sec	13 jours	∞
$n = 100$	0	0	0	0	13 ans	10^{14} ans	∞
$n = 1000$	0	0	0	1 sec	∞	∞	∞
$n = 10^4$	0	0	0	17 min	∞	∞	∞
$n = 10^5$	0	0	10 sec	12 jours	∞	∞	∞
$n = 10^6$	0	0	16 min	32 ans	∞	∞	∞

$0 =$ moins de 1 seconde
$\infty =$ plus de 10^{20} ans

Complexité algorithmique : définitions

Le temps de calcul d'un algorithme A sur une entrée $w \in \Sigma^*$ est le nombre d' $\underbrace{\text{instructions élémentaires}}_{\text{Définition ? Modèle de calcul ! MT, RAM, RASP, ...}}$ exécutées, noté $t_A(w)$.

Ce nombre donne le temps de calcul en secondes à un facteur multiplicatif près correspondant à la vitesse du processeur.

2 GHz $=$ facteur $\frac{1}{2 \cdot 10^9}$.

La complexité (en temps) dans le pire cas d'un algorithme A est une fonction $\mathbb{N} \to \mathbb{N}$ de la taille de l'entrée :

$$T_A(n) = \max_{w \in \Sigma^n} t_A(w)$$

La complexité (en temps) dans le pire cas d'un problème P est la complexité du meilleur algorithme qui résoud ce problème :

$$T_P(n) = \inf_{\substack{\text{algo } A \\ \text{qui résoud } P}} T_A(n)$$

Divulgâcheur : rares sont les problèmes dont la complexité est connue (démontrée sup-inf), mais depuis les années 1970 on a une belle théorie de la complexité : NP-complétude, etc...

Quiz

Bilbiographie

Diapos des CM, pas de correction des TD et TP.
Vous devez venir en cours.

- Cormen : *Introduction to Algorithms* (1990)
- Arora-Barak : *Computational Complexity* (2009)
- Garey-Johnson : *Computers and Intractability* (1979)
- Papadimitriou : *Computational Complexity* (1994)
- Perifel : *Complexité Algorithmique* (2014)