

# Complexité algorithmique

Pierre Rouchon

Centre Automatique et Systèmes  
Mines ParisTech, PSL Research University  
[pierre.rouchon@mines-paristech.fr](mailto:pierre.rouchon@mines-paristech.fr)

Octobre 2018

# Plan

- 1 Introduction
- 2 Classe **P**
- 3 Classe **NP**
- 4 Classe **RP**
- 5 Fonctions à sens unique et **P**  $\neq$  **NP**

- La donnée (on parle aussi d'instance) est un entier  $n$  et  $F_n$  une fonction booléenne de  $n$  variables booléennes ( $x_i \in \{0, 1\}$ ,  $i = 1, \dots, n$ )

$$(x_1, \dots, x_n) \mapsto F_n(x_1, \dots, x_n) \in \{0, 1\}.$$

où  $F$  est construite avec des expressions faisant intervenir les opérateurs logiques usuels (et, ou et négation). La question est :

existe-t-il un  $(x_1, \dots, x_n)$  tel que  $F_n(x_1, \dots, x_n) = 1$  ?

- Problème représentatif de la classe des problèmes dits de **complexité NP** car c'est un problème de difficulté maximale dans cette classe (problème dit **NP**-complet). Un problème est dit **NP** si l'on peut certifier ces instances positives en temps polynômial en  $n$  par un oracle ( $n$  est bien la taille des données).

- La donnée est un entier  $n$ ,  $F_n$  une fonction booléenne de  $n$  variables booléennes ( $x_i \in \{0, 1\}$ ,  $i = 1, \dots, n$ ) et la formule avec quantificateurs

$$\forall x_1, \exists x_2, \dots \quad F(x_1, \dots, x_n).$$

La question est alors : **cette formule est-elle vraie ?**

- Le second problème est représentatif de la classe des problèmes dits de **complexité PSPACE** car c'est un problème de difficulté maximale dans cette classe. Nous n'aborderons pas cette classe de problèmes qui admettent un algorithme nécessitant un espace mémoire polynomial en  $n$ .

- La donnée est un entier  $n$  et un polynôme à coefficients entiers de  $n$  variables  $P(x_1, \dots, x_n)$ . La question est alors (dixième problème de Hilbert) : l'équation diophantienne  $P(x_1, \dots, x_n) = 0$  admet-elle une solution entière  $(x_1, \dots, x_n) \in \mathbb{Z}^n$  ?
- Il n'existe pas d'algorithme pour décider si la réponse est oui ou non.
- Prototype de **problème indécidable** au sens de Gödel :

$$\forall (a_1, \dots, a_k) \in \mathbb{Z}^k, \quad \exists (x_1, \dots, x_n) \in \mathbb{Z}^n, \text{ tel que } P(a, x) = 0$$

où  $P$  est un polynôme à coefficients entiers en  $a$  et  $x$ .

**Indécidable** : qui n'est pas une conséquence finitiste (conséquence "courte") des axiomes de base de l'arithmétique.

- **Définition** : le problème  $\mathcal{P}$  sera dit de classe **P** s'il existe un **algorithme polynômial en temps** (i.e., en nombre d'opérations élémentaires) qui le résout. Par polynômial, nous entendons polynômial par rapport à l'espace nécessaire pour coder les données  $x$ .
- **Exemple** : Les données sont deux entiers positifs  $m < n$ . La question est : les entiers  $m$  et  $n$  sont-ils premiers entre eux ? Nous savons par l'algorithme d'Euclide calculer le pgcd. Ce calcul nécessite au plus  $E \left( \frac{\log(n)}{\log\left(\frac{1+\sqrt{5}}{2}\right)} \right)$  divisions pour  $m \leq n$ .

## Equivalence entre

- le calcul en temps polynômial d'une famille de fonctions  $(f_n)_{n \in \mathbb{N}}$  de  $\{1, \dots, n\}$  dans lui même et le problème de décision suivant.
- Les données sont l'entier  $n$  et deux autres entiers  $x$  et  $t$  plus petits que  $n$ . La question est :

$$\Sigma : \text{A-t-on } f_n(x) \leq t \quad ?$$

- En effet si  $f_n(x)$  se calcule en temps polynômial par rapport à  $\log(n)$ , le problème  $\Sigma$  est de façon évidente dans  $\mathbf{P}$ .
- **Supposons maintenant que  $\Sigma$  est dans  $\mathbf{P}$** . Prenons  $n$  et  $x$  entiers avec  $1 \leq x \leq n$ , voyons comment calculer  $f_n(x)$  en temps polynômial. Pour cela, nous pouvons savoir en temps polynômial si  $f_n(x) \in [1, n/2]$  ou  $f_n(x) \in [n/2, n]$ . Si  $f_n(x) \in [1, n/2]$  on peut savoir en temps polynômial si  $f_n(x) \in [1, n/4]$  ou  $f_n(x) \in [n/4, n/2]$ . Si  $f_n(x) \in [n/2, n]$  on peut savoir en temps polynômial si  $f_n(x) \in [n/2, 3n/4]$  ou  $f_n(x) \in [3n/4, n]$ . On voit bien qu'avec de telles **dichotomies**, on sait en faisant **appel  $s$ -fois à  $\Sigma$  si  $f_n(x)$  est dans un intervalle de longueur au plus  $n/2^s$** . Il suffit maintenant de prendre  $s = E(\log_2(n))$  pour avoir la valeur exacte de  $f_n(x)$  puisque c'est un entier. On aura obtenu ainsi la valeur de  $f_n(x)$  en résolvant  **$E(\log_2(n))$  fois le problème  $\Sigma$** . Donc le calcul de  $f_n(x)$  est polynômial en  $\log(n)$ .



Le problème de décision  $\mathcal{P}$  est dit calculable **par un algorithme non déterministe et polynômial en temps**, si et seulement si,

- il existe un algorithme (certification) ayant comme données de départ  $x$  et **aussi  $y$**  (fini et correspondant à **l'oracle**),
- tel que pour toute instance  $x$  **vérifiant  $\mathcal{P}(x)$  vrai** alors il existe un certificat  **$y(x)$**  tel que cet algorithme ayant  $x$  et  **$y(x)$**  comme données calcule  $\mathcal{P}(x)$  vrai en temps polynômial par rapport à la taille des données de départ  $x$ .

- **Le problème de la factorisation est dans NP** . Nous le traduisons d'abord en un problème de décision : les données sont deux entiers  $n$  et  $M < n$ . La question est : **existe-t-il un diviseur de  $n$  plus petit que  $M$  et  $> 1$ .**
- Par dichotomie successive, on voit que si l'on sait résoudre ce problème en  $\Upsilon(\log(n))$  opérations élémentaires , on sait trouver un diviseur de  $n$  en temps polynômial. On part de  $M = E(n/2)$ , un premier calcul donne la position du diviseur éventuel soit dans  $[2, E(n/2)[$  ou  $[E(n/2), n[$ , i.e. dans un intervalle de longueur au plus  $n/2$ . Un second calcul va le localiser dans un intervalle de longueur au plus  $n/4$ . Après  $s$  calculs on a localisé le diviseur dans un intervalle de longueur au plus  $n/2^s$ . Ainsi avec  $s = E(\log_2(n))$  on aura localisé le diviseur dans un intervalle de longueur au plus de 1, i.e. on aura donc le diviseur au bout du temps  $\Upsilon(\log(n))\log_2(n)$ .
- Les spécialistes pensent que  $\Upsilon$  ne peut pas être un polynôme en  $\log n$ .

- Le problème de décision, **existe-t-il un diviseur de  $n$  plus petit que  $M$  et  $> 1$**  est dans **NP**.
- Il suffit pour les instances  $x = (n, M)$  positives (i.e., telles qu'il existe un diviseur de  $n$  plus petit que  $M$ ) de prendre un diviseur de  $n$  plus petit que  $M$  que nous noterons  $y(n, M)$  (**oracle**).  
L'algorithme de vérification consiste simplement à diviser  $n$  par  $y(n, M)$  et ainsi on vérifie que  $n$  a bien un diviseur non trivial plus petit que  $M$ .

- Le problème de décision, **existe-t-il un diviseur de  $n$  plus petit que  $M$  et  $> 1$  est dans **coNP**** .
- On s'intéresse à  $x = (n, M)$  tel qu'il n'existe pas de diviseur de  $n$  plus petit que  $M$ . Pour cela, **la structure de  $y$  est plus lourde**. En effet, il faut que  $y$  comporte les données suivantes : la décomposition de  $n$  en facteurs premiers  $n = \prod_{i=1}^k p_i^{\nu_i}$ . Avec ces données supplémentaires  $y = (p_i, \nu_i)_{i=1, \dots, k}$  nous pouvons proposer l'algorithme non optimal suivant :
  - 1 vérification via AKS que les  $k$  nombres  $p_i$  sont des nombres premiers ;
  - 2 vérification que les  $p_i$  sont plus grands que  $M$ .
  - 3 vérification que  $n = \prod_{i=1}^k p_i^{\nu_i}$ .

On laisse au lecteur le soin de montrer que cet algorithme totalement sous optimal est cependant en temps polynômial par rapport à  $\log(n)$ .

On peut utiliser la même démarche pour montrer que le logarithme discret est dans **NP** et aussi dans **coNP** .

- Pour  $p$  premier,  $\alpha$  primitif modulo- $p$  et  $n < p$  on définit la fonction  $\log$  ainsi

$$(p, \alpha, n) \mapsto \log(p, \alpha, n) = \begin{cases} m & \text{si } p \text{ est premier, } \alpha \text{ primitif modulo-}p \\ & \text{et } m \text{ l'unique entier dans } \{1, \dots, p-1\} \\ & \text{tel que } \alpha^m = n \pmod{p} \\ 0 & \text{sinon.} \end{cases}$$

- le problème de décision est alors le suivant : les données sont  $p$  et les nombres  $\alpha$ ,  $n$  et  $t$  plus petits que  $p$  ;  
la question est " A-t-on  $\log(p, \alpha, n) < t$  ? "

Problèmes pouvant être résolus par des **algorithmes probabilistes polynômiaux**. Un problème  $\mathcal{P}(x)$  est dans **RP** ssi :

- 1 ils existent des polynômes  $p(n)$  et  $q(n)$  ( $n$  est la taille de  $x$  ;  $n = \log(x)$  si  $x$  est un entier)
- 2 il existe un algorithme ayant comme données de départ  $x$  et  $y$  (certificat) et qui donne une réponse binaire  $R(x, y) \in \{-1, 1\}$  en un temps plus petit que  $q(n)$  si le certificat  $y$  est de taille plus petite que  $p(n)$ .
- 3  $\mathcal{P}(x)$  est faux si et seulement si  $R(x, y) = -1$  pour tout certificat  $y$  de taille plus petit que  $p(n)$ .
- 4 Si  $\mathcal{P}(x)$  est vrai, alors en choisissant au hasard un certificat  $y$  de taille plus petite que  $p(n)$ , on a  $R(x, y) = 1$  avec une probabilité supérieure ou égale à  $1/2$

**Exo:** Montrer que le seuil de  $1/2$  pour la probabilité peut être remplacé par n'importe quel réel  $\sigma$  avec  $0 < \sigma < 1$ .

**Exo:** Montrer que **RP** est dans **NP**.

Cette définition est faite sur mesure pour le **test de Miller-Rabin de primalité**.

$\mathcal{P}(x)$  : l'entier  $x$  est-il composé ?

Avec  $n = \log(x)$

- 1  $y$  correspond à un entier entre 2 et  $x$ ,  $p(n) = n$  et  $q(n)$  est la complexité du test de Miller-Rabin :  $x$  est-il fortement premier en base  $y$  ?
- 2  $R(x, y) = -1$  si  $x$  est fortement premier en base  $y$ ,  
 $R(x, y) = 1$  sinon.
- 3  $\mathcal{P}(x)$  faux équivaut à  $\forall y \leq x$  on a  $R(x, y) = -1$  ( $x$  premier équivaut à  $\forall y \leq x$ ,  $x$  fortement premier en base  $y$ ).
- 4  $\mathcal{P}(x)$  vrai implique que, pour au moins la moitié des certificats  $y$  de taille plus petite que  $p(n)$ , on a  $R(x, y) = 1$  (si  $x$  est composé alors pour au moins les  $3/4$  des  $y$  entre 2 et  $x - 1$ ,  $x$  n'est pas fortement premier en base  $y$ ).

- L'existence de fonctions à sens unique est une conjecture aussi difficile que  $\mathbf{P} \neq \mathbf{NP}$ .
- Chaque  $f_n$  est une bijection de  $A_n = \{1, \dots, n\}$  dans  $B_n = \{1, \dots, n\}$ . On considère alors la famille  $(f_n)_{n \in \mathbb{N}}$ .
- Le fait que les  $f_n$  soient faciles à calculer se formalise alors via le problème noté  $\mathcal{F}$  suivant : les données sont un entier  $n$  et deux autres entiers  $x$  et  $t$  entre 1 et  $n$ . La question est :  
$$\mathcal{F}(n, x, t) : \text{A-t-on } f_n(x) \leq t ?$$
- Si pour chaque  $n$  le calcul de  $f_n(x)$  est polynômial, le problème  $\mathcal{F}$  est dans  $\mathbf{P}$ .



Supposons donc  $\mathcal{F}$  dans  $\mathbf{P}$  .

- Calcul de l'inverse des  $f_n$  est associé au problème de décision suivant noté  $\mathcal{F}^{-1}$  : les données sont un entier  $n$  et deux autres entiers  $x$  et  $t$  entre 1 et  $n$  ; la question est :  
$$\mathcal{F}^{-1}(n, x, t) : \text{a-t-on } f_n^{-1}(x) \leq t ?$$
- $\mathcal{F}^{-1}$  est dans  $\mathbf{NP}$  . En effet, il suffit de prendre comme certificat  $y = f_n^{-1}(x)$ .
- Le fait que le problème  $\mathcal{F}^{-1}$  soit difficile, i.e., que les  $f_n$  soient à sens unique, se traduit donc par le fait que  $\mathcal{F}^{-1}$  n'est pas dans  $\mathbf{P}$  (car sinon le calcul de  $f_n^{-1}$  serait polynômial).

Comme  $\mathcal{F}^{-1}$  est nécessairement dans  $\mathbf{NP}$  si  $\mathcal{F}$  est dans  $\mathbf{P}$  , on voit que l'existence d'une fonction à sens unique implique  $\mathbf{P} \neq \mathbf{NP}$  .

- On dit qu'un problème de décision  $\mathcal{F}$  se réduit polynomialement à un problème  $\mathcal{F}^*$ , s'il est possible, à partir d'un algorithme qui résout les instances positives de  $\mathcal{F}^*$ , de construire, avec des **calculs supplémentaires au plus polynômiaux**, un algorithme qui résout les instances positives de  $\mathcal{F}$  (si  $\mathcal{F}^*$  est dans **P** (resp. **NP**) alors  $\mathcal{F}$  l'est aussi).
- Un problème  $\mathcal{F}^*$  est dit **NP** -complet, s'il est dans **NP** et si tout problème de **NP** se réduit polynomialement à  $\mathcal{F}^*$ .
- Le problème SAT est **NP** -complet :  
    **existe-t-il un  $(x_1, \dots, x_n)$  tel que  $F_n(x_1, \dots, x_n) = 1$  ?**  
    où  $F_n$  est une fonction booléenne des variables binaires  $x_j$ .

# Les diverses conjectures sur les classes de complexité

- On a les inclusions évidentes suivantes :

$$\mathbf{P} \subseteq \mathbf{RP} \subseteq \mathbf{NP} \quad \text{et} \quad \mathbf{P} \subseteq (\mathbf{NP} \cap \mathbf{coNP})$$

- On conjecture que :

$$\mathbf{P} \subsetneq \mathbf{RP} \subsetneq \mathbf{NP} \quad \text{et} \quad \mathbf{P} \subsetneq (\mathbf{NP} \cap \mathbf{coNP}) \subsetneq \mathbf{NP}.$$