Introduction à la complexité algorithmique

(survol des classes de complexité usuelles)

Arnaud Casteigts

29 juin et 6 juillet 2021

GT Info-Quantique du LaBRI

Complexité algorithmique ?

Étude des ressources nécessaires pour résoudre un **problème**.

Problème $\Pi : \{0,1\}^* \to \{0,1\}^*$

Met en relation une **entrée** (ou **instance**) avec une **sortie**.

Résoudre un problème Π = savoir calculer $\Pi(x)$ pour toute instance $x \in \{0,1\}^*$.

Types de problèmes (exemples où l'instance est un graphe G)

- **Décision:** est-ce que G admet une k-coloration pour un certain k ?
- **Recherche:** donner une k-coloration de G pour un certain k.
- **Dénombrement:** combien G admet-il de k-colorations pour un certain k ?
- **Optimisation:** trouver le plus petit k tel que G admet une k-coloration.

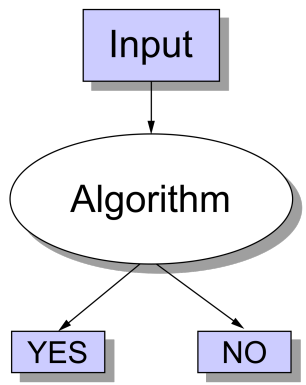
Problème de décision (focus)

Cas particulier où $\Pi : \{0,1\}^* \to \{0,1\}$ (réponse OUI ou NON)

$x \in \{0,1\}^*$ est une **instance positive** (resp. **négative**) si $\Pi(x) = 1$ (resp. 0)

Les instances positives définissent un **langage formel** $L \subseteq \{0,1\}^*$

Résoudre un problème de décision $\equiv$ reconnaître le langage associé



Complexité algorithmique ?

Étude des **ressources** nécessaires pour résoudre un problème.

Quel type de ressources ?

- Temps
- Espace (= mémoire)
- Non-déterminisme
- Aléa (p.ex. nombre de bits aléatoires)
- ...

Point de vue asymptotique

- **Évolution** de ces quantités en fonction de la **taille** de l'entrée (en général, notée n)
- Notations $O(\cdot), \Omega(\cdot), \Theta(\cdot), o(\cdot), \omega(\cdot)$
  Intuition $\leq \quad \geq \quad = \quad < \quad >$ pour $n \to \infty$, en ignorant les facteurs constants
- Quelques adjectifs:

Constant	$O(1)$	Quadratique	$O(n^2)$
Logarithmique	$O(\log n)$	Polynomial	$O(n^c) = n^{O(1)} = poly(n)$
Linéaire	$O(n)$	Exponentiel	$O(2^n)$ ou $O(2^{poly(n)})$
Quasi-linéaire	$O(n \log n)$	Factoriel	$O(n^n)$

L'espace versus le temps

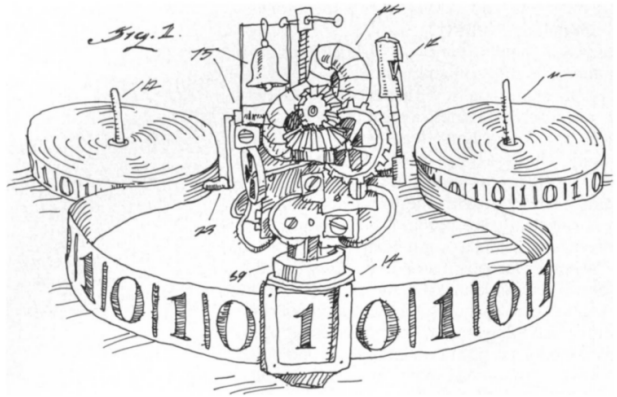
Sauf mention contraire, on ne parle désormais que de problèmes **de décision**

Classes génériques

- $\mathrm{TIME}(f(n))$: Ensemble de problèmes que l'on peut résoudre en temps $O(f(n))$.
- $\mathrm{SPACE}(f(n))$: Ensemble de problèmes que l'on peut résoudre en espace $O(f(n))$.

Modèle de machine ?

- Machine de Turing déterministe
- Intuition "algorithme standard" OK
- Universalité ?

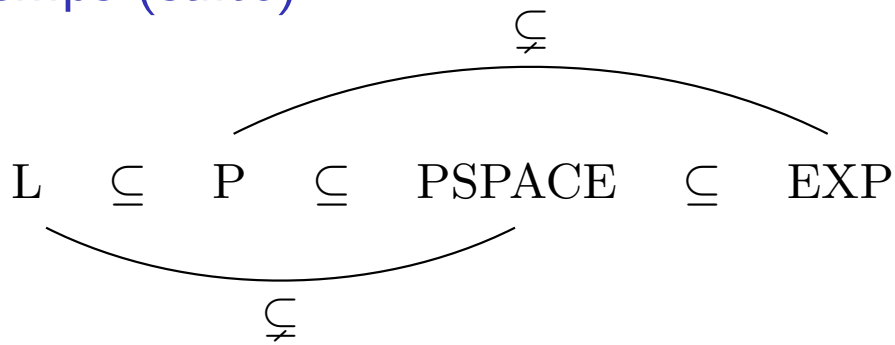


Classes incontournables

L	:=	$\mathrm{SPACE}(\log n)$	Problèmes résolubles en **espace logarithmique**
P	:=	$\mathrm{TIME}(poly(n))$	Problèmes résolubles en **temps polynomial**
PSPACE	:=	$\mathrm{SPACE}(poly(n))$	Problèmes résolubles en **espace polynomial**
EXP	:=	$\mathrm{TIME}(2^{poly(n)})$	Problèmes résolubles en **temps exponentiel**

$$\mathrm{L} \subseteq \mathrm{P} \subseteq \mathrm{PSPACE} \subseteq \mathrm{EXP}$$

L'espace versus le temps (suite)



Aucune inclusion n'est démontrée stricte, mais elles sont toutes soupçonnées l'être.

Théorèmes de hiérarchie (légèrement simplifiés)

Pour deux fonctions $f_1(n)$ et $f_2(n)$ abrégées en f_1 et f_2:

- Hiérarchie en temps :
  $f_1 = o(f_2 \log f_2) \implies \mathrm{TIME}(f_1) \subsetneq \mathrm{TIME}(f_2)$ (Stearns et Hartmanis'65, puis d'autres)
- Hiérarchie en espace :
  $f_1 = o(f_2) \implies \mathrm{SPACE}(f_1) \subsetneq \mathrm{SPACE}(f_2)$

Espace versus temps ?

- $\mathrm{TIME}(f(n)) \subseteq \mathrm{SPACE}(f(n)/\log(f(n)))$ (Hopcroft, Paul, Valiant'77)
  $\rightarrow$ L'espace est **strictement** plus fort que le temps !

Non-déterminisme (NP et coNP)

Plusieurs définitions équivalentes, la plus simple :

NP: ∃ **preuve courte** que la réponse est OUI (certificat positif)

coNP: ∃ **preuve courte** que la réponse est NON (certificat négatif)

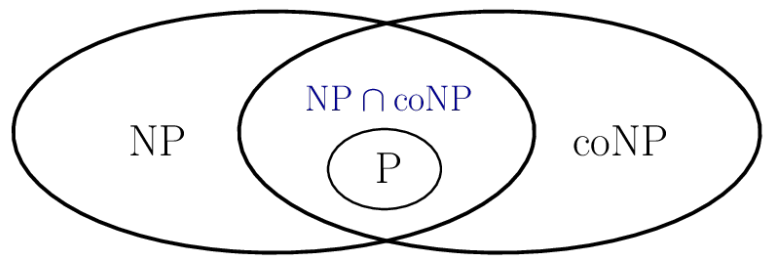
Preuve courte = vérifiable en temps polynomial

Exemple : 3-COLORATION ∈ NP (certificat = la coloration elle-même)

Intuition non-déterministe : capacité à explorer en parallèle (**deviner**) un certificat.

Exercice : classer les problèmes suivant

- Ce graphe est-il connexe ?
- Ce nombre est-il premier ? est-il composite ?
- Ce nombre admet-il un facteur $\leq k$?
- Ces deux graphes sont-ils isomorphes ?
- Ce graphe admet-il une clique de taille k ?
- Ce plateau d'échec ($n \times n$) admet-il une stratégie gagnante pour le joueur 1 ?
- ~~L'hypothèse de Riemann est-elle vraie ?~~
  Cet énoncé est-il vrai ?



Non-déterminisme (suite)

Plus généralement

- $\mathrm{NTIME}(f(n))$: $\exists$ certificats positifs vérifiables en temps $O(f(n))$.
- $\mathrm{NSPACE}(f(n))$: $\exists$ certificats positifs vérifiables en espace $O(f(n))$.
- $\mathrm{coNTIME}$ et $\mathrm{coNSPACE}$: idem pour certificats négatifs.

Non-déterminisme (suite)

Plus généralement

- $\mathrm{NTIME}(f(n))$: $\exists$ certificats positifs vérifiables en temps $O(f(n))$.
- $\mathrm{NSPACE}(f(n))$: $\exists$ certificats positifs vérifiables en espace $O(f(n))$.
- $\mathrm{coNTIME}$ et $\mathrm{coNSPACE}$: idem pour certificats négatifs.

Classes non-déterministes incontournables (les mêmes qu'avant)

$$
\begin{array}{lcl}
\mathrm{NL} & := & \mathrm{NSPACE}(\log n) \\
\mathrm{NP} & := & \mathrm{NTIME}(\mathit{poly}(n)) \\
\mathrm{NPSPACE} & := & \mathrm{NSPACE}(\mathit{poly}(n)) \\
\mathrm{NEXP} & := & \mathrm{NTIME}(2^{\mathit{poly}(n)})
\end{array}
$$

Non-déterminisme (suite)

Plus généralement

- $\mathrm{NTIME}(f(n))$: $\exists$ certificats positifs vérifiables en temps $O(f(n))$.
- $\mathrm{NSPACE}(f(n))$: $\exists$ certificats positifs vérifiables en espace $O(f(n))$.
- $\mathrm{coNTIME}$ et $\mathrm{coNSPACE}$: idem pour certificats négatifs.

Classes non-déterministes incontournables (les mêmes qu'avant)

$$
\begin{array}{lcl}
\mathrm{NL} & := & \mathrm{NSPACE}(\log n) \\
\mathrm{NP} & := & \mathrm{NTIME}(\mathit{poly}(n)) \\
\mathrm{NPSPACE} & := & \mathrm{NSPACE}(\mathit{poly}(n)) \\
\mathrm{NEXP} & := & \mathrm{NTIME}(2^{\mathit{poly}(n)})
\end{array}
$$

Relations

- $\mathrm{P} \subseteq \mathrm{NP} \cap \mathrm{coNP}$ (algo = vérifieur avec certificat vide)
- $\mathrm{NSPACE}(f(n)) \subseteq \mathrm{SPACE}(f^2(n))$ (théorème de Savitch'70)
- $\mathrm{NSPACE}(f(n)) = \mathrm{coNSPACE}(f(n))$ (théorème d'Immerman–Szelepcsényi'87)
- $\mathrm{NTIME}(f(n)) \subseteq \mathrm{SPACE}(f(n))$ (énumération des certificats)

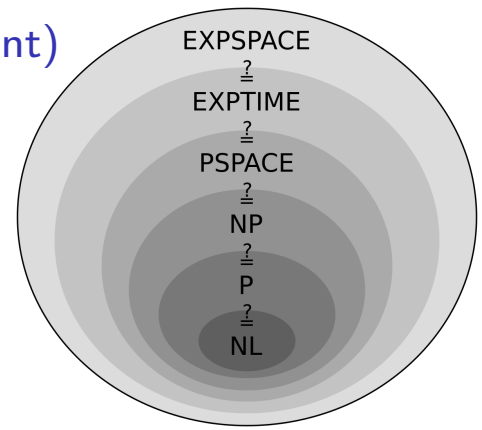
Non-déterminisme (suite)

Plus généralement

- $\mathrm{NTIME}(f(n))$: $\exists$ certificats positifs vérifiables en temps $O(f(n))$.
- $\mathrm{NSPACE}(f(n))$: $\exists$ certificats positifs vérifiables en espace $O(f(n))$.
- $\mathrm{coNTIME}$ et $\mathrm{coNSPACE}$: idem pour certificats négatifs.

Classes non-déterministes incontournables (les mêmes qu'avant)

$$
\begin{array}{lcl}
\mathrm{NL} & := & \mathrm{NSPACE}(\log n) \\
\mathrm{NP} & := & \mathrm{NTIME}(\mathit{poly}(n)) \\
\mathrm{NPSPACE} & := & \mathrm{NSPACE}(\mathit{poly}(n)) \\
\mathrm{NEXP} & := & \mathrm{NTIME}(2^{\mathit{poly}(n)})
\end{array}
$$



Relations

- $\mathrm{P} \subseteq \mathrm{NP} \cap \mathrm{coNP}$ (algo = vérifieur avec certificat vide)
- $\mathrm{NSPACE}(f(n)) \subseteq \mathrm{SPACE}(f^2(n))$ (théorème de Savitch'70)
- $\mathrm{NSPACE}(f(n)) = \mathrm{coNSPACE}(f(n))$ (thérorème d'Immerman–Szelepcsényi'87)
- $\mathrm{NTIME}(f(n)) \subseteq \mathrm{SPACE}(f(n))$ (énumération des certificats)
- $\mathrm{L} \stackrel{?}{=} \mathrm{NL} \stackrel{?}{=} \mathrm{P} \stackrel{?}{=} \mathrm{NP} \stackrel{?}{=} \mathrm{PSPACE} \stackrel{?}{=} \mathrm{EXP} \stackrel{?}{=} \mathrm{NEXP} \stackrel{?}{=} \mathrm{EXPSPACE}$ (inclusions OK)

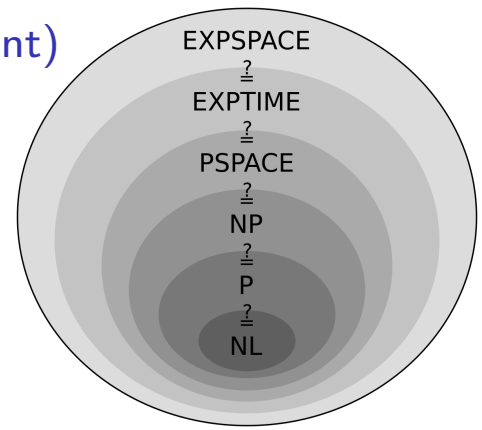
Non-déterminisme (suite)

Plus généralement

- $\mathrm{NTIME}(f(n))$: $\exists$ certificats positifs vérifiables en temps $O(f(n))$.
- $\mathrm{NSPACE}(f(n))$: $\exists$ certificats positifs vérifiables en espace $O(f(n))$.
- $\mathrm{coNTIME}$ et $\mathrm{coNSPACE}$: idem pour certificats négatifs.

Classes non-déterministes incontournables (les mêmes qu'avant)

$$
\begin{array}{lcl}
\mathrm{NL} & := & \mathrm{NSPACE}(\log n) \\
\mathrm{NP} & := & \mathrm{NTIME}(\mathit{poly}(n)) \\
\mathrm{NPSPACE} & := & \mathrm{NSPACE}(\mathit{poly}(n)) \\
\mathrm{NEXP} & := & \mathrm{NTIME}(2^{\mathit{poly}(n)})
\end{array}
$$



Relations

- $\mathrm{P} \subseteq \mathrm{NP} \cap \mathrm{coNP}$ (algo = vérifieur avec certificat vide)
- $\mathrm{NSPACE}(f(n)) \subseteq \mathrm{SPACE}(f^2(n))$ (théorème de Savitch'70)
- $\mathrm{NSPACE}(f(n)) = \mathrm{coNSPACE}(f(n))$ (thérorème d'Immerman–Szelepcsényi'87)
- $\mathrm{NTIME}(f(n)) \subseteq \mathrm{SPACE}(f(n))$ (énumération des certificats)
- $\mathrm{L} \stackrel{?}{=} \mathrm{NL} \stackrel{?}{=} \mathrm{P} \stackrel{?}{=} \mathrm{NP} \stackrel{?}{=} \mathrm{PSPACE} \stackrel{?}{=} \mathrm{EXP} \stackrel{?}{=} \mathrm{NEXP} \stackrel{?}{=} \mathrm{EXPSPACE}$ (inclusions OK)
- Arguments de remplissage (padding) : $\mathrm{P} = \mathrm{NP} \implies \mathrm{EXP} = \mathrm{NEXP}$ (et d'autres)

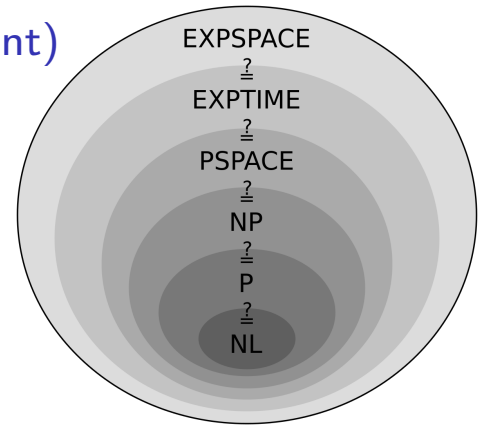
Non-déterminisme (suite)

Plus généralement

- $\mathrm{NTIME}(f(n))$: $\exists$ certificats positifs vérifiables en temps $O(f(n))$.
- $\mathrm{NSPACE}(f(n))$: $\exists$ certificats positifs vérifiables en espace $O(f(n))$.
- $\mathrm{coNTIME}$ et $\mathrm{coNSPACE}$: idem pour certificats négatifs.

Classes non-déterministes incontournables (les mêmes qu'avant)

$$
\begin{array}{lcl}
\mathrm{NL} & := & \mathrm{NSPACE}(\log n) \\
\mathrm{NP} & := & \mathrm{NTIME}(\mathit{poly}(n)) \\
\mathrm{NPSPACE} & := & \mathrm{NSPACE}(\mathit{poly}(n)) \\
\mathrm{NEXP} & := & \mathrm{NTIME}(2^{\mathit{poly}(n)})
\end{array}
$$



Relations

- $\mathrm{P} \subseteq \mathrm{NP} \cap \mathrm{coNP}$ (algo = vérifieur avec certificat vide)
- $\mathrm{NSPACE}(f(n)) \subseteq \mathrm{SPACE}(f^2(n))$ (théorème de Savitch'70)
- $\mathrm{NSPACE}(f(n)) = \mathrm{coNSPACE}(f(n))$ (thérorème d'Immerman–Szelepcsényi'87)
- $\mathrm{NTIME}(f(n)) \subseteq \mathrm{SPACE}(f(n))$ (énumération des certificats)
- $\mathrm{L} \stackrel{?}{=} \mathrm{NL} \stackrel{?}{=} \mathrm{P} \stackrel{?}{=} \mathrm{NP} \stackrel{?}{=} \mathrm{PSPACE} \stackrel{?}{=} \mathrm{EXP} \stackrel{?}{=} \mathrm{NEXP} \stackrel{?}{=} \mathrm{EXPSPACE}$ (inclusions OK)
- Arguments de remplissage (padding) : $\mathrm{P} = \mathrm{NP} \implies \mathrm{EXP} = \mathrm{NEXP}$ (et d'autres)
- $\mathbf{P} \stackrel{?}{=} \mathbf{NP} \sim$ **Ce qui est facile à vérifier est-il facile à trouver ?**

Réductions et complétude

Réduction polynomiale

Un problème Π_1 **se réduit** en *temps polynomial* à un problème Π_2, noté $\Pi_1 \leq \Pi_2$, si...

Réductions et complétude

Réduction polynomiale

Un problème Π_1 **se réduit** en *temps polynomial* à un problème Π_2, noté $\Pi_1 \leq \Pi_2$, si...

- (par transformation) ... pour toute instance x_1 de Π_1, on peut construire en temps polynomial une instance x_2 de Π_2 telle que $\Pi_1(x_1) = \Pi_2(x_2)$.

Réductions et complétude

Réduction polynomiale

Un problème Π_1 **se réduit** en *temps polynomial* à un problème Π_2, noté $\Pi_1 \leq \Pi_2$, si...

- (par transformation) ... pour toute instance x_1 de Π_1, on peut construire en temps polynomial une instance x_2 de Π_2 telle que $\Pi_1(x_1) = \Pi_2(x_2)$.
- (par oracle) ... on peut écrire un algorithme qui résoud Π_1 en temps polynomial en ayant accès à **un oracle** pour Π_2 (machine qui répond en une étape à des questions de Π_2).

On dit que "Π_2 est au moins aussi difficile que Π_1".

Réductions et complétude

Réduction polynomiale

Un problème Π_1 **se réduit** en *temps polynomial* à un problème Π_2, noté $\Pi_1 \leq \Pi_2$, si...

- (par transformation) ... pour toute instance x_1 de Π_1, on peut construire en temps polynomial une instance x_2 de Π_2 telle que $\Pi_1(x_1) = \Pi_2(x_2)$.
- (par oracle) ... on peut écrire un algorithme qui résoud Π_1 en temps polynomial en ayant accès à **un oracle** pour Π_2 (machine qui répond en une étape à des questions de Π_2).

On dit que "Π_2 est au moins aussi difficile que Π_1".

Autres réductions ? Équivalence entre transformation et oracle ?

Réductions et complétude

Réduction polynomiale

Un problème Π_1 **se réduit** en *temps polynomial* à un problème Π_2, noté $\Pi_1 \leq \Pi_2$, si...

- (par transformation) ... pour toute instance x_1 de Π_1, on peut construire en temps polynomial une instance x_2 de Π_2 telle que $\Pi_1(x_1) = \Pi_2(x_2)$.
- (par oracle) ... on peut écrire un algorithme qui résoud Π_1 en temps polynomial en ayant accès à **un oracle** pour Π_2 (machine qui répond en une étape à des questions de Π_2).

On dit que "Π_2 est au moins aussi difficile que Π_1".

Autres réductions ? Équivalence entre transformation et oracle ?

Difficulté et complétude

Pour une classe $\mathcal{C}$ de problèmes et un type de réduction:

- $\mathcal{C}$-difficile: au moins aussi difficile que tout problème dans $\mathcal{C}$
- $\mathcal{C}$-complet: à la fois dans $\mathcal{C}$ et $\mathcal{C}$-difficile

Réductions et complétude

Réduction polynomiale

Un problème Π_1 **se réduit** en *temps polynomial* à un problème Π_2, noté $\Pi_1 \leq \Pi_2$, si...

- (par transformation) ... pour toute instance x_1 de Π_1, on peut construire en temps polynomial une instance x_2 de Π_2 telle que $\Pi_1(x_1) = \Pi_2(x_2)$.
- (par oracle) ... on peut écrire un algorithme qui résoud Π_1 en temps polynomial en ayant accès à **un oracle** pour Π_2 (machine qui répond en une étape à des questions de Π_2).

On dit que "Π_2 est au moins aussi difficile que Π_1".

Autres réductions ? Équivalence entre transformation et oracle ?

Difficulté et complétude

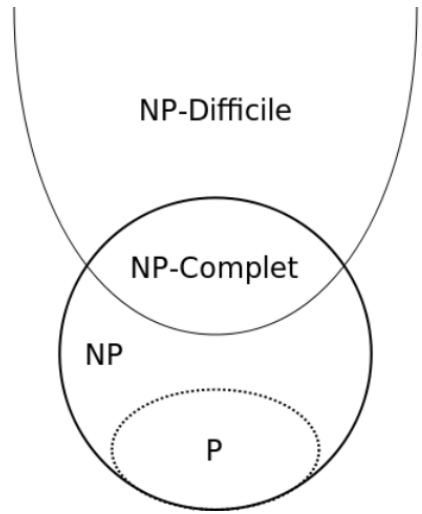
Pour une classe $\mathcal{C}$ de problèmes et un type de réduction:

- $\mathcal{C}$-difficile: au moins aussi difficile que tout problème dans $\mathcal{C}$
- $\mathcal{C}$-complet: à la fois dans $\mathcal{C}$ et $\mathcal{C}$-difficile

Ex: $\text{NP-complet} = \text{dans NP et NP-difficile}$

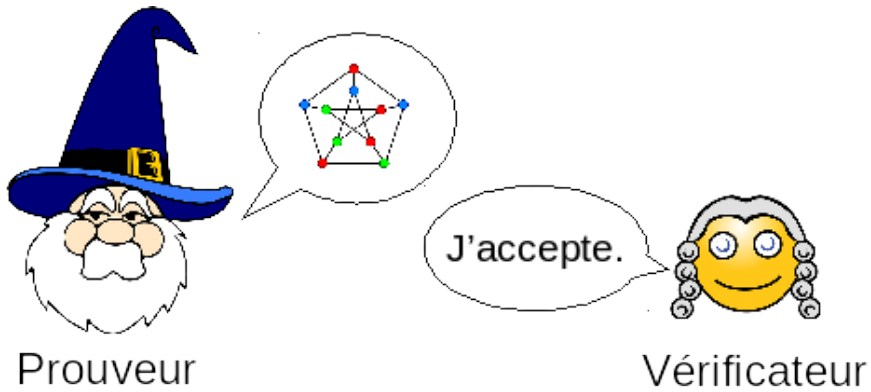
(ici, *par transformation* et *en temps polynomial*)

$\text{SAT} \in \text{NP-complet}$ (Cook, Levin'71)



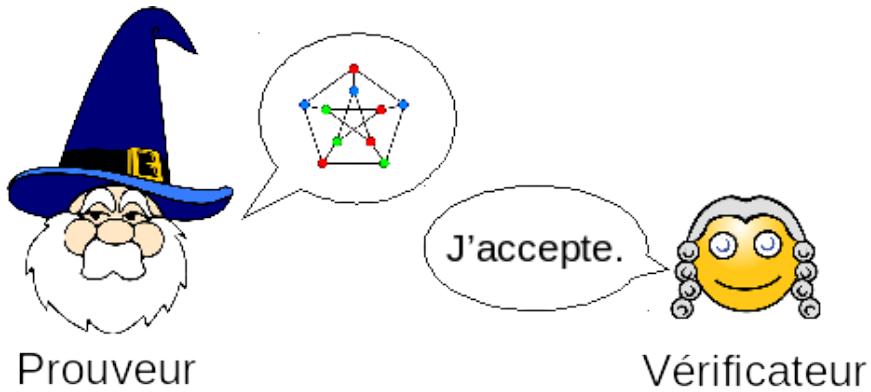
Preuves interactives

Une autre façon de voir NP



Preuves interactives

Une autre façon de voir NP



Variantes

- Le vérificateur peut utiliser de l'aléa: Arthur-Merlin (AM) et Merlin-Arthur (MA)
- Aléa + nombre d'interaction polynomial: classe IP
- $\mathrm{IP} = \mathrm{PSPACE}$ (Shamir'92)
- Plusieurs prouveurs (MIP), $\mathrm{MIP} = \mathrm{NEXPTIME}$ (Babai, Fortnow, Lund'91)
- $\mathrm{MIP}^* = \mathrm{RE}$ (Ji, Natarajan, Vidick, Wright, Yuen'20)

  (MIP^* : les prouveurs partagent des bits intriqués)
  (RE : langages récursivement énumérables)

Machines à oracles et hiérarchie polynomiale

Classes définies par oracles

- Soient deux classes X et Y, on note X^Y la classe des problèmes résolubles par un algorithme de X ayant accès à un oracle pour un problème Y-complet.
- Par exemple, $\mathrm{P}^{\mathrm{NP}} =$ problèmes résolubles en temps polynomial avec un oracle pour SAT.
- Les oracles ne sont pas forcément des pbs de décisions, par exemple, $\#\mathrm{P}$ (dénombrement).

Machines à oracles et hiérarchie polynomiale

Classes définies par oracles

- Soient deux classes X et Y, on note X^Y la classe des problèmes résolubles par un algorithme de X ayant accès à un oracle pour un problème Y-complet.
- Par exemple, P^{NP} = problèmes résolubles en temps polynomial avec un oracle pour SAT.
- Les oracles ne sont pas forcément des pbs de décisions, par exemple, $\#\mathrm{P}$ (dénombrement).

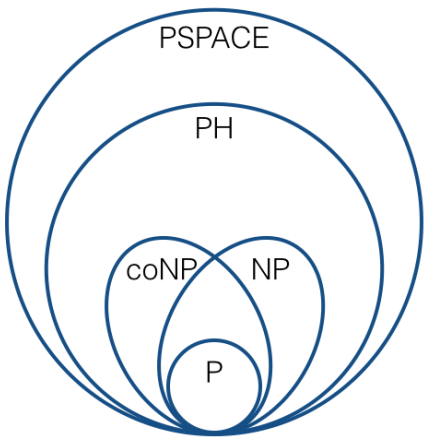
Hiérarchie polynomiale (PH)

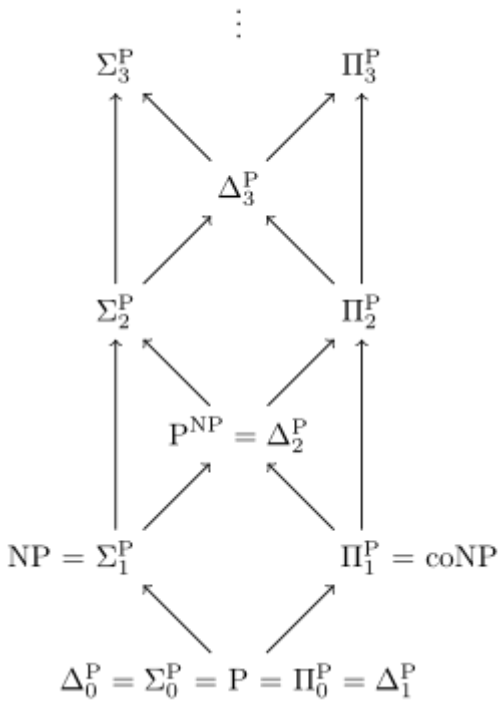
(Meyer et Stockmeyer'72)

- Principe : Itération d'oracles impliquant P, NP et coNP
- Problèmes complets à chaque étage

  Quantified SAT (QSAT):

  $\exists_{x1} \forall_{x_2} \exists_{x_3} \ldots, \phi(x_1, x_2, x_3, \ldots)$
- Machine de Turing alternantes
- $\mathrm{PH} \subseteq \mathrm{P}^{\#\mathrm{P}}$ (Toda'91)
- $\mathrm{PH} \stackrel{?}{=} \mathrm{PSPACE}$





Aléa et ressources quantiques

BPP: *Bounded-error probabilistic polynomial time*

- Problèmes résolubles en temps polynomial par un **algorithme probabiliste** (peut tirer à pile ou face) avec une probabilité d'erreur inférieure à $1/3$

Aléa et ressources quantiques

BPP: *Bounded-error probabilistic polynomial time*

- Problèmes résolubles en temps polynomial par un **algorithme probabiliste** (peut tirer à pile ou face) avec une probabilité d'erreur inférieure à $1/3$

BQP: *Bounded error quantum polynomial time*

- Problèmes résolubles en temps polynomial par un **ordinateur quantique** avec une probabilité d'erreur inférieure à $1/3$

Aléa et ressources quantiques

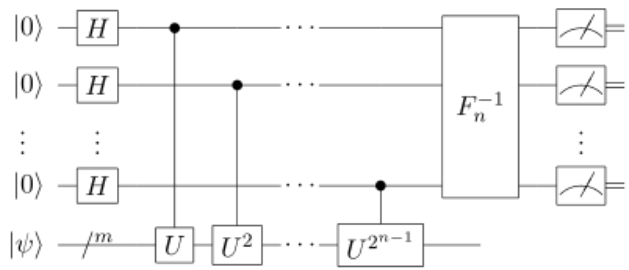
BPP: *Bounded-error probabilistic polynomial time*

- Problèmes résolubles en temps polynomial par un **algorithme probabiliste** (peut tirer à pile ou face) avec une probabilité d'erreur inférieure à 1/3

BQP: *Bounded error quantum polynomial time*

- Problèmes résolubles en temps polynomial par un **ordinateur quantique** avec une probabilité d'erreur inférieure à 1/3
- Concrètement, circuit ou machine de Turing ?

  Les deux, au choix :
  - circuits quantiques uniformes
  - machines de turing quantiques



Aléa et ressources quantiques

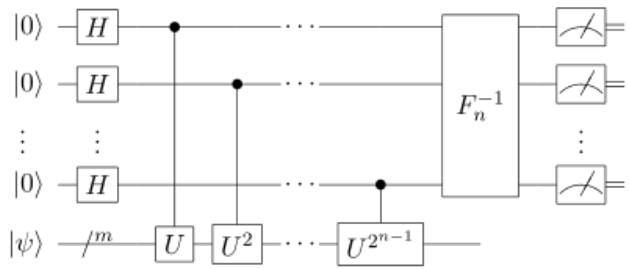
BPP: *Bounded-error probabilistic polynomial time*

- Problèmes résolubles en temps polynomial par un **algorithme probabiliste** (peut tirer à pile ou face) avec une probabilité d'erreur inférieure à $1/3$

BQP: *Bounded error quantum polynomial time*

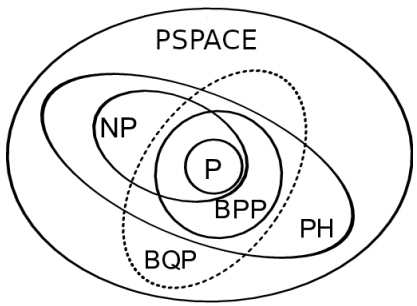
- Problèmes résolubles en temps polynomial par un **ordinateur quantique** avec une probabilité d'erreur inférieure à $1/3$
- Concrètement, circuit ou machine de Turing ?

  Les deux, au choix :
  - circuits quantiques uniformes
  - machines de turing quantiques



Que sait-on ?

- $\mathrm{P} \subseteq \mathrm{BPP}$ $(0 \leq 1/3)$
- $\mathrm{BPP} \subseteq \mathrm{BQP}$ (non-réversibilité simulable en temps poly)
- $\mathrm{BQP} \subseteq \mathrm{PSPACE}$ (Bernstein et Vazirani'97)
- $\textsc{Factoring} \in \mathrm{BQP}$ (Shor'94)
- Quid de BQP *versus* NP ? (pressentie incomparables)



(attention ↑ conjectures)

Séparation par oracle

Principe : on veut montrer que deux classes sont différentes, mais on y arrive pas.

$\rightarrow$ Essayer de montrer au moins qu'elles sont différentes **relativement** à un certain oracle.

Séparation par oracle

Principe : on veut montrer que deux classes sont différentes, mais on y arrive pas.

$\rightarrow$ Essayer de montrer au moins qu'elles sont différentes **relativement** à un certain oracle.

Exemples de séparations

- Il existe un oracle X tel que $P^X \neq NP^X$ (Baker, Gill, Solovay'75)

  (cependant : il existe aussi Y tel que $P^Y = NP^Y$) ("Relativization barrier")
- $\exists X$ tel que $\mathrm{NP}^X \subsetneq \mathrm{BQP}^X$ (Bennett, Bernstein, Brassard, Vazirani'97)
- $\exists X$ tel que $\mathrm{BQP}^X \subsetneq \mathrm{NP}^X$ (Berthiaume, Brassard'92)
- $\exists X$ tel que $\mathrm{BQP}^X \subsetneq \mathrm{PH}^X$ (Raz, Tal'19)

Séparation par oracle

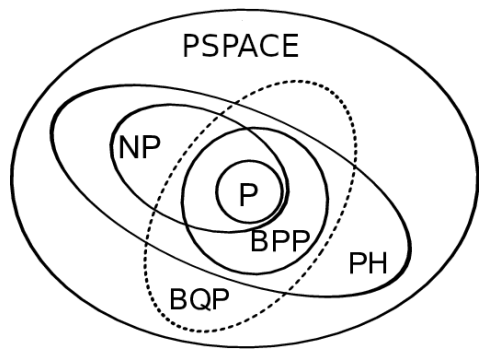
Principe : on veut montrer que deux classes sont différentes, mais on y arrive pas.

$\rightarrow$ Essayer de montrer au moins qu'elles sont différentes **relativement** à un certain oracle.

Exemples de séparations

- Il existe un oracle X tel que $P^X \neq NP^X$ (Baker, Gill, Solovay'75)

  (cependant : il existe aussi Y tel que $P^Y = NP^Y$) ("Relativization barrier")
- $\exists X$ tel que $\mathrm{NP}^X \subsetneq \mathrm{BQP}^X$ (Bennett, Bernstein, Brassard, Vazirani'97)
- $\exists X$ tel que $\mathrm{BQP}^X \subsetneq \mathrm{NP}^X$ (Berthiaume, Brassard'92)
- $\exists X$ tel que $\mathrm{BQP}^X \subsetneq \mathrm{PH}^X$ (Raz, Tal'19)



(attention $\uparrow$ conjectures)

Séparation par oracle

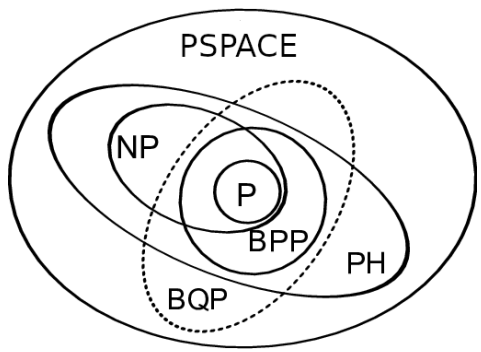
Principe : on veut montrer que deux classes sont différentes, mais on y arrive pas.

$\rightarrow$ Essayer de montrer au moins qu'elles sont différentes **relativement** à un certain oracle.

Exemples de séparations

- Il existe un oracle X tel que $P^X \neq NP^X$ (Baker, Gill, Solovay'75)

  (cependant : il existe aussi Y tel que $P^Y = NP^Y$) ("Relativization barrier")
- $\exists X$ tel que $\mathrm{NP}^X \subsetneq \mathrm{BQP}^X$ (Bennett, Bernstein, Brassard, Vazirani'97)
- $\exists X$ tel que $\mathrm{BQP}^X \subsetneq \mathrm{NP}^X$ (Berthiaume, Brassard'92)
- $\exists X$ tel que $\mathrm{BQP}^X \subsetneq \mathrm{PH}^X$ (Raz, Tal'19)



(attention $\uparrow$ conjectures)

Remarque

N'importe quelle séparation non-relative impliquerait, à minima, que $\mathrm{P} \neq \mathrm{PSPACE}$, ce qui est un des principaux problèmes ouverts du domaine.

Les paris sont ouverts

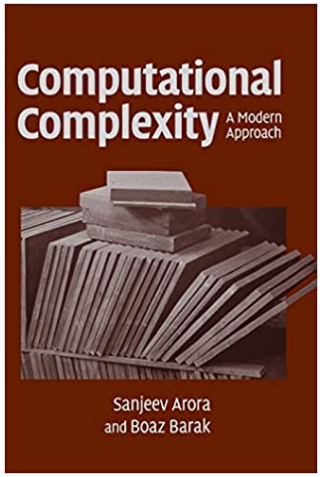
Proposition	RW's Estimated Likelihood
TRUE	100%
$\mathsf{EXP}^{\mathsf{NP}} \neq \mathsf{BPP}$	99%
$\mathsf{NEXP} \not\subset \mathsf{P/poly}$	97%
$\mathsf{L} \neq \mathsf{NP}$	95%
$\mathsf{NP} \not\subset \mathsf{SIZE}(n^k)$	93%
$\mathsf{BPP} \subseteq \mathsf{SUBEXP}$	90%
$\mathsf{P} \neq \mathsf{PSPACE}$	90%
$\mathsf{P} \neq \mathsf{NP}$	80%
ETH	70%
$\mathsf{NC1} \neq \mathsf{TC0}$	50%
$\mathsf{NEXP} \neq \mathsf{EXP}$	45%
SETH	25%
$\mathsf{NEXP} \neq \mathsf{coNEXP}$	20%
NSETH	15%
$\mathsf{L} \neq \mathsf{RL}$	5%
FALSE	0%

Ryan Williams (2019)

Accept

Pour aller plus loin

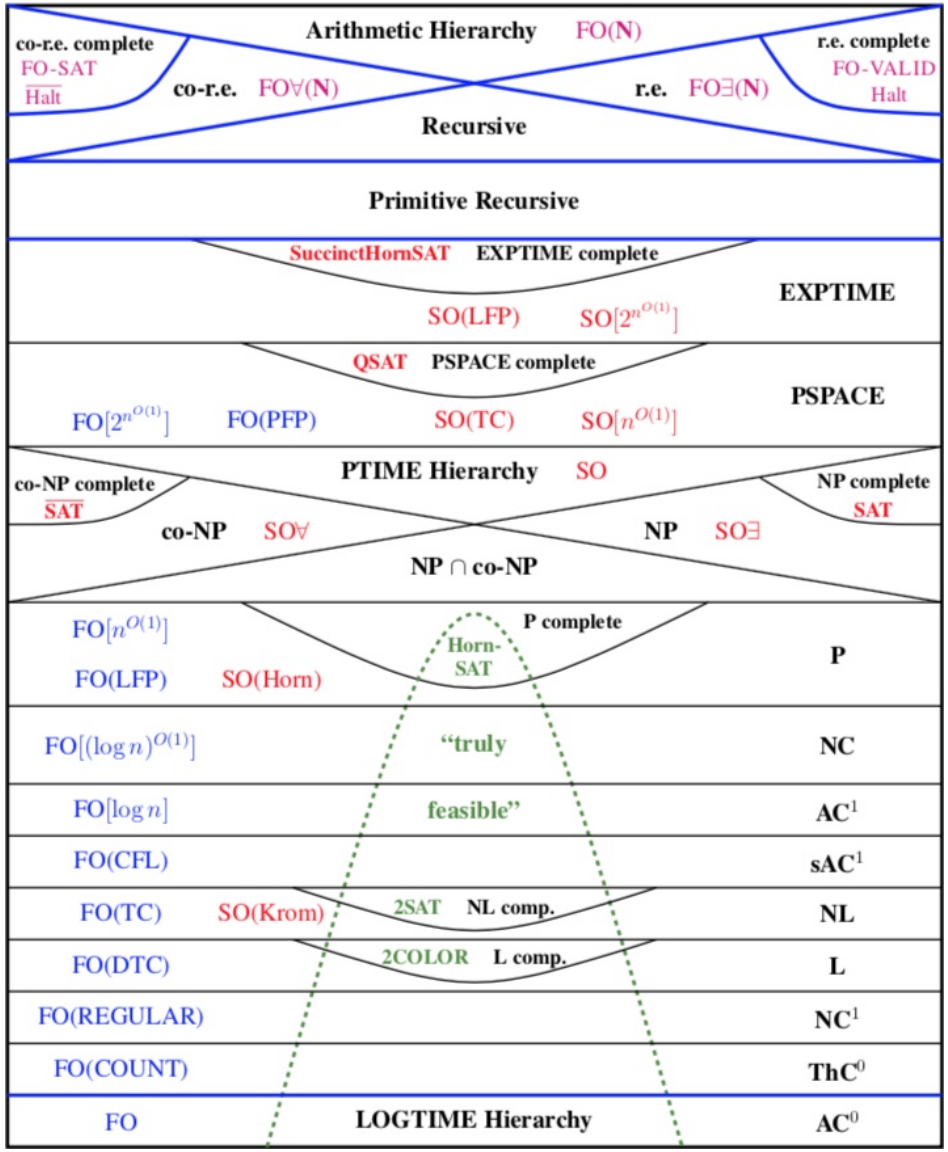
- Livre “Computational Complexity” (Arora et Barak’2007)
- Chaîne YouTube de Ryan O’Donnell (cours Carnegie Mellon)
- Wikipedia (si si !)



Choses dont je n’ai pas parlé

- Complexité de circuits (classe SIZE (f(n)), AC^0, NC, $\mathrm{P/poly}$)
- Bornes inférieures en lien avec les circuits
- Calcul du permanent ($\#\mathrm{P}$-complet) et Boson Sampling
- Complexité descriptive (liens entre logique et complexité algorithmique)
- Cryptographie : comment générer des instances difficiles
- Les cinq mondes d’Impagliazzo
- ...

Complexité descriptive



Neil Immerman (1999)