Calculabilité et Complexité: CM8

Florian Bridoux

Université de Caen

2021-2022

Table des matières

Table des matières

Introduction

- Jusqu'à présent, nous avons abordé la complexité algorithmique seulement à travers le prisme du temps de calcul: pour qu'un calcul soit utilisable, il est en effet nécessaire qu'il s'exécute rapidement.
- Mais il existe une autre ressource critique : la mémoire. Par exemple, un algorithme effectuant n^4 opérations peut éventuellement encore être considéré comme raisonnablement efficace en termes de temps d'exécution puisque le temps nécessaire à une machine actuelle effectuant 10^{11} opérations par seconde pour exécuter l'algorithme sur une entrée de taille 10000 est d'environ un jour.
- Mais si à chaque étape il utilise une nouvelle case mémoire alors sa mémoire sera saturée bien avant qu'il ait pu terminer son calcul.

DSPACE

Définition:

- Si M est une machine de Turing déterministe et x une entrée, l'espace utilisé par M sur x est le nombre de cases différentes visitées par M sur ses rubans de travail au cours de son calcul.
- Si $s : \mathbb{N} \to \mathbb{N}$ est une fonction, on dit que M fonctionne en espace $O(s(n))$ si elle s'arrête sur toute entrée et s'il existe une constante $\alpha > 0$ telle que pour toute entrée x, $M(x)$ utilise un espace $\leq \alpha s(|x|)$.
- Si $s : \mathbb{N} \to \mathbb{N}$ est une fonction, la classe **DSPACE**$(s(n))$ est l'ensemble des langages reconnus par une machine déterministe fonctionnant en espace $O(s(n))$.

Une machine pourrait boucler infiniment et ne jamais s'arrêter tout en utilisant un espace $\leq s(n)$, c'est pourquoi nous devons imposer l'arrêt sur toute entrée des machines que nous considérons.

DSPACE et NSPACE

Définition:

- Si $s : \mathbb{N} \to \mathbb{N}$ est une fonction, on dit qu'une machine non déterministe N fonctionne en espace $O(s(n))$ si elle s'arrête sur toute entrée le long de toute exécution et s'il existe une constante $\alpha > 0$ telle que pour toute entrée x, $N(x)$ utilise un espace $\leq \alpha s(n)$ sur tout chemin de calcul.
- Si $s : \mathbb{N} \to \mathbb{N}$ est une fonction, la classe **NSPACE**$(s(n))$ est l'ensemble des langages reconnus par une machine non déterministe fonctionnant en espace $O(s(n))$.

DSPACE et NSPACE

Théorème:

- Pour toute fonction $t : \mathbb{N} \to \mathbb{N}$,
  $\mathbf{DTIME}(t) \subseteq \mathbf{NTIME}(t) \subseteq \mathbf{DSPACE}(t)$;
- Pour toute fonction $s : \mathbb{N} \to \mathbb{N}$,
  $\mathbf{DSPACE}(s) \subseteq \mathbf{NSPACE}(s) \subseteq \mathbf{DTIME}(2^{O(s)})$.

- En espace $t(n)$ on peut simplement énumérer tous les chemins possibles et simuler la machine fonctionnant en temps $t(n)$.
- Le nombre de configurations différentes d'une machine M fonctionnant en espace $s(n)$ sur un alphabet Γ et avec $|Q|$ états, ne dépasse pas $|Q||\Gamma|^{(s(n))}$. Si M prenait un temps supérieur à ça, elle passerait alors deux fois par la même configuration et bouclerait infiniment.

Table des matières

PSPACE

Définition: PSPACE

La classe PSPACE est l'ensemble des langages reconnus par une machine de Turing fonctionnant en espace polynomial, c'est-à-dire $\text{PSPACE} = \bigcup_{k>0} \mathbf{DSPACE}(n^k)$.

- On a $P \subseteq NP \subseteq PSPACE \subseteq EXP$.
- On ne sait pas si $NP \neq PSPACE$ ou si $NP \subseteq EXP$, mais on sait que $P \neq EXP$.

Définition: NSPACE

On définit NSPACE similairement à DSPACE:
$NSPACE = \bigcup_{k>0} \mathbf{NSPACE}(n^k)$.

Théorème

NSPACE=PSPACE

PSPACE

(QBF) Quantified Boolean Formulas :
Entrée: Une formule μ et n quantificateur $Q_i \in \{\forall, \exists\}$.
Question: $Q_1\lambda_1 Q_2\lambda_2 \dots \lambda_n Q_n\mu$ est-elle vraie?

Preuve (au tableau):

- QBF est dans PSPACE (et donc dans NSPACE).
- *QBF* est NSPACE-complet.
- Donc, PSPACE = NSPACE.

Remarque

On ne l'a pas prouvé, mais on peut considérer que la formule μ en entrée est une CNF ou DNF. Et le problème 3-QBF est également PSPACE-complet.

Table des matières

Classe LOGSPACE, EXPSPACE

- LOGSPACE $= \bigcup_{k>0} \mathbf{DSPACE}(\log^k(n))$. Cette classe de complexité nous impose d'adapter la définition de la machine de turing étant donné que simplement l'entrée du problème prend déjà un espace polynomial (linéaire).
- EXSPACE $= \bigcup_{k>0} \mathbf{DSPACE}(2^{n^k})$.

On sait que LOGSPACE $\subset$ PSPACE $\subset$ EXPSPACE.

Classe DP

La classe des problèmes qui "combinent" un problème dans NP et un problème dans co-NP.
Problème DP-complet canonique:

SAT-UNSAT
Entrée: Une formule CNF μ, une formule DNF μ'.
Question: μ est satisfiable et μ' une tautologie?

Classe probabiliste PP

La classe PP est l'ensemble de problèmes de décision décidés par une machine de Turing probabiliste en temps polynomial avec une probabilité d'erreur inférieure à un demi.
Problème PP-complet canonique:

> *MAJSAT*
> *Entrée: Une formule CNF μ sur n variables.*
> *Question: Est-ce qu'au moins 2^{n-1} valuations satisfont μ?*

Attention, par un résultat de Shyan Akmal, Ryan Williams de 2021 (!), on sait que 3-MAJSAT est dans P et donc normalement pas PP-complet.
On a $P \subseteq NP \subseteq PP \subseteq PSPACE$.

Classe probabiliste PP

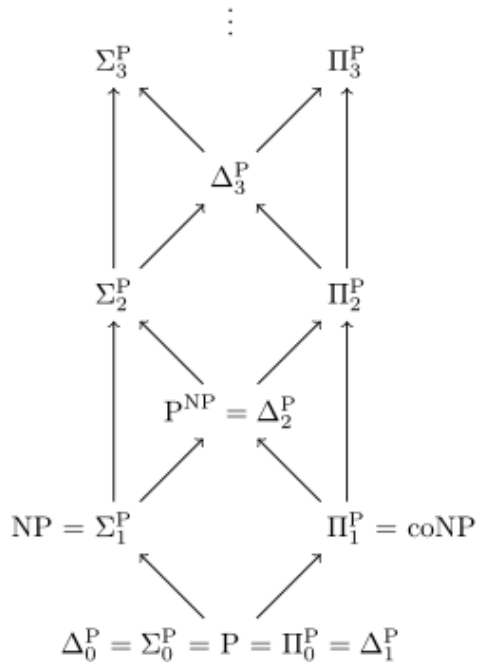
La classe PP est l'ensemble de problèmes de décision décidés par une machine de Turing probabiliste en temps polynomial avec une probabilité d'erreur inférieure à un demi.
Problème PP-complet canonique:

> *MAJSAT*
> *Entrée: Une formule CNF μ sur n variables.*
> *Question: Est-ce qu'au moins 2^{n-1} valuations satisfont μ?*

On a $P \subseteq NP \subseteq PP \subseteq PSPACE$.
Attention, par un résultat de Shyan Akmal, Ryan Williams de 2021 (!), on sait que 3-MAJSAT est dans P et donc normalement pas PP-complet.

La hierarchie polynomiale

La classe $\Sigma_2^P = NP^{NP}$ représente la classe des problèmes résolvables en temps polynomial par une machine déterministe avec en oracle une autre machines.
On définit similairement $\Sigma_3^P = NP^{NP^{NP}}$, $\Sigma_4^P = NP^{NP^{NP^{NP}}}$,...
On peut aussi définir $\Pi_2^P = coNP^{NP}$, $\Pi_3^P = coNp^{NP^{NP}}$, ...
Et également $\Delta_2^P = P^{NP}$, ...

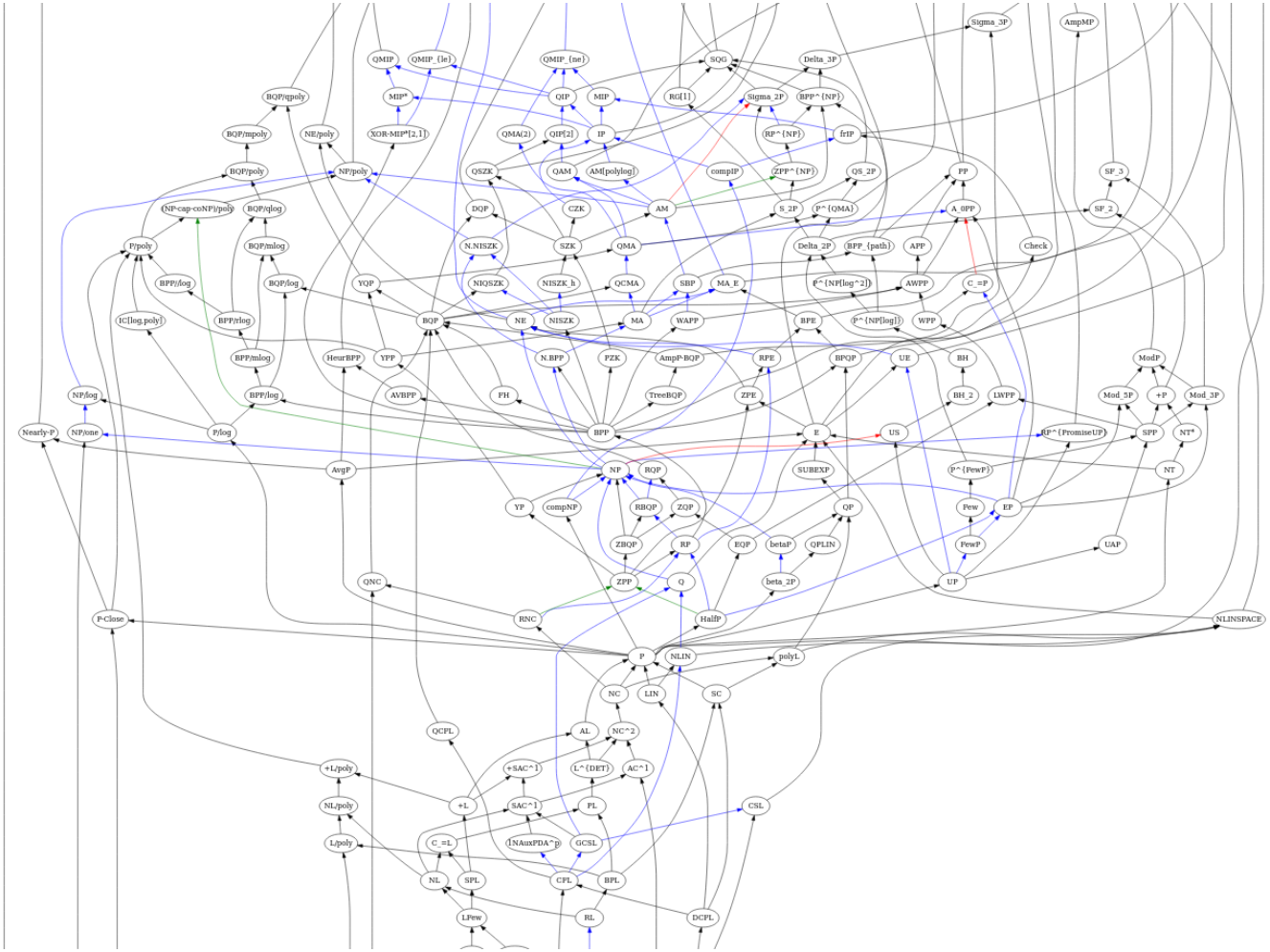


Représentation graphique de la hiérarchie polynomiale. Les flèches indiquent l'inclusion.

La hiérarchie polynomiale

- On définie $PH = \bigcup_{k \in \mathbb{N}} \{\Sigma_k^P, \Pi_k^P, \Delta_k^P\}$.
- On a
  $P = \Sigma_0^P \subseteq NP = \Sigma_1^P \subseteq \Sigma_2^P \subseteq \Sigma_3^P \subseteq \cdots \subseteq PH \subseteq PSPACE$.
- On pense que ces classes sont toutes séparées, mais il est prouvé que si $\Sigma_i^P = \Sigma_{i+1}^P$ alors $\Sigma^P i = PH$. C'est ce qu'on appelle l'hypothèse de l'effondrement polynomiale.
- On pense qu'il n'existe aucun problème PH-complet. Si c'était le cas, alors on peut prouver que l'hypothèse de l'effondrement serait vraie.

Des tonnes d'autres classes de complexités



des centaines d'autres classes de complexités.