Classes de complexité probabilistes

François Schwarzentruber

1[er] mars 2022

1 Définitions

Définition 1 ZPP (*Zero-Error Probabilistic Polynomial time*) est la classe des problèmes de décision décidés par un algorithme probabiliste en **temps polynomial en espérance**.

Définition 2 RP (*Randomized Polynomial time*) est la classe des problèmes de décision L pour lesquels il existe un algorithme probabiliste A dont le temps est polynomial et :
- si $x \in L$, alors $\mathbb{P}(A(x)$ répond oui$) \geq 1/2$;
- si $x \notin L$, alors $\mathbb{P}(A(x)$ répond non$) = 1$.

Définition 3 $\text{coRP} = \{L \mid \overline{L} \in \text{RP}\}$.

Proposition 4 coRP est la classe des problèmes de décision L pour lesquels il existe un algorithme probabiliste A dont le temps est polynomial et :
- si $x \notin L$, alors $\mathbb{P}(A(x)$ répond non$) \geq 1/2$;
- si $x \in L$, alors $\mathbb{P}(A(x)$ répond oui$) = 1$.

2 Lien avec P

Proposition 5 $\text{P} \subseteq \text{ZPP}$.

Proposition 6 $\text{P} \subseteq \text{RP}$.

3 Exemples de problèmes

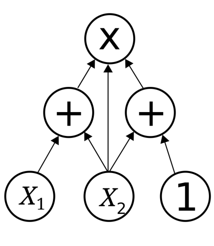
Définition 7 **PRIMES**

entrée : un nombre entier écrit en binaire
sortie : oui, s'il est premier, non sinon.

Le test de primalité de Solovay et Strassen [SS77] (ainsi que celui de Miller-Rabin) montre que **PRIMES** est dans coRP. Adleman et Huang [AH87] ont montré qu'il est aussi dans RP. Finalement, Agrawal-Kayal-Saxena [AKS04] ont montré que **PRIMES** est dans P.

Définition 8 **Polynomial identity testing (PIT)**

entrée : un circuit arithmétique représentant un polynôme multivarié
sortie : oui, si le circuit représente le polynôme nul, non sinon.



PIT est dans RP et on ne sait pas s'il est dans P.

4 Lien avec NP

Proposition 9 (reformulation de la définition de NP) Un langage L est dans NP s'il existe un algorithme déterministe V en temps poly tel que :

— si $x \in L$, il existe $y \in \{0,1\}^{poly(|x|)}$ tel que $V(x,y) = 1$.
— si $x \notin L$, pour tout $y \in \{0,1\}^{poly(|x|)}$, on a $V(x,y) = 0$.

Proposition 10 (reformulation de la définition de RP) Un langage L est dans RP s'il existe un algorithme déterministe V en temps poly tel que :

— si $x \in L$, plus de la moitié des $y \in \{0,1\}^{poly(|x|)}$ sont tels que $V(x,y) = 1$.
— si $x \notin L$, pour tout $y \in \{0,1\}^{poly(|x|)}$, on a $V(x,y) = 0$.

Proposition 11 RP $\subseteq$ NP

Démonstration. On convertit les choix aléatoires en des choix non-déterministes. ■

5 Lien entre RP et ZPP

Proposition 12 ZPP $=$ RP $\cap$ coRP.

Démonstration. $\boxed{\supseteq}$ Soit L un problème dans RP $\cap$ coRP.

Comme $L \in$ RP, il existe un algorithme A en temps polynomial avec :

— si $w \in L$, alors $\mathbb{P}(A(w)$ renvoie oui$) \geq 1/2$;
— si $w \notin L$, alors $\mathbb{P}(A(w)$ renvoie non$) = 1$.

Comme $L \in$ coRP, il existe un algorithme B en temps polynomial avec :

— si $w \in L$, alors $\mathbb{P}(B(w)$ renvoie oui$) = 1$;
— si $w \notin L$, alors $\mathbb{P}(B(w)$ renvoie non$) \geq 1/2$.

On définit l'algorithme suivant, ZPP-algo :

```
entrée : une instance x
sortie : renvoie oui ssi x ∈ L
fonction ZPP-algo(x)
    tant que vrai
        si A(x) renvoie oui alors
            renvoyer oui
        si B(x) renvoie non alors
            renvoyer non
```

Soit $T(x)$ le temps d'exécution de ZPP-algo(x). Soit P un polynôme tel que $P(|x|)$ majore le temps d'exécution de $A(x)$, et celui de $B(x)$.

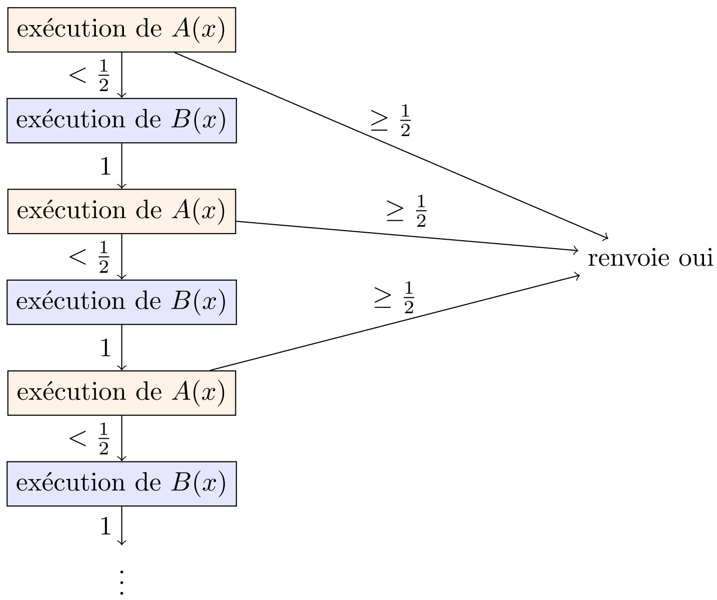
$$\begin{aligned}
\mathbb{E}(T(x)) &= \sum_{k>0} \mathbb{P}(\text{l'algo fait exactement } k \text{ itérations}) \times \text{temps d'exécution de } k \text{ itérations} \\
&< \sum_{k>0} \frac{1}{2^{k-1}} \times 2kP(|x|) \text{ par le lemme 13} \\
&= 4P(|x|) \times \sum_{k\geq 0} \frac{k}{2^k} \text{ par le lemme 14} \\
&\leq 8P(|x|)
\end{aligned}$$

Lemme 13 $\mathbb{P}(\text{l'algo fait exactement } k \text{ itérations}) < \frac{1}{2^{k-1}}$.

Démonstration.

$$\mathbb{P}(\text{l'algo fait exactement } k \text{ itérations}) \leq \mathbb{P}(\text{l'algo ne s'arrête pas durant les } k-1 \text{ premières itérations})$$

Supposons que $x \in L$ (l'analyse du cas $x \notin L$ est symétrique). Voici le graphe des exécutions possibles de ZPP-algo(x) :



On lit que $\mathbb{P}($l'algo ne s'arrête pas durant les $k-1$ premières itérations$) < \frac{1}{2^{k-1}}$, ce qui conclut la démonstration du lemme. ■

Lemme 14 $\sum_{k\geq 0} \frac{k}{2^k} = 2$.

Démonstration. Introduisons la série génératrice $G(z) = \sum_{k>=0} \frac{z^k}{2^k}$,

— D'une part, $G'(z) = \sum_{k>=1} \frac{kz^{k-1}}{2^k}$, et on reconnait $\sum_{k\geq 0} \frac{k}{2^k} = G'(1)$.

— D'autre part, $G(z) = \frac{2}{2-z}$, et $G'(z) = \frac{2}{(2-z)^2}$. D'où $G'(1) = 2$.

■

$\boxed{\subseteq}$ Réciproquement, soit L un problème dans ZPP. Montrons que L est dans RP. Il existe un algorithme probabiliste A qui décide exactement L en temps d'espérance borné par un certain polynôme $P(n)$. On construit :

```
fonction RP-algo(x)
    exécuter A(x) pendant 2P(|x|) étapes
    si A(x) s'est arrêté alors
        renvoyer la réponse de A(x)
    sinon
        renvoyer non
```

Si $x \notin L$, alors RP-algo(x) renvoie non.
Si $x \in L$, soit $T(x)$ le temps d'exécution de $A(x)$.

$$\dfrac{\dfrac{\text{Algo } A \text{ décide exactement } L}{\mathbb{P}(\text{RP-algo}(x) \text{ renvoie oui}) = \mathbb{P}(\, T(x) < 2P(|x|)\,)} \qquad \dfrac{\dfrac{\dfrac{\text{Définition de ZPP}}{\mathbb{E}(T(x)) \leq P(|x|)} \quad \dfrac{\text{Inégalité de Markov}}{\mathbb{P}(\, T(x) \geq 2P(|x|)\,) \leq \frac{\mathbb{E}(T(x))}{2P(|x|)}}}{\mathbb{P}(\, T(x) \geq 2P(|x|)\,) \leq \frac{1}{2}}}{\mathbb{P}(\, T(x) < 2P(|x|)\,) \geq \frac{1}{2}}}{\mathbb{P}(\text{RP-algo}(x) \text{ renvoie oui}) = \mathbb{P}(\, T(x) < 2P(|x|)\,) \geq \frac{1}{2}}$$

D'où $L \in$ RP. De manière similaire on montre que $L \in$ coRP.

■

Références

[AH87] Leonard M. Adleman and Ming-Deh A. Huang. Recognizing primes in random polynomial time. In Alfred V. Aho, editor, *Proceedings of the 19th Annual ACM Symposium on Theory of Computing, 1987, New York, New York, USA*, pages 462–469. ACM, 1987.

[AKS04] Manindra Agrawal, Neeraj Kayal, and Nitin Saxena. Primes is in p. *Annals of mathematics*, pages 781–793, 2004.

[SS77] Robert Solovay and Volker Strassen. A fast monte-carlo test for primality. *SIAM J. Comput.*, 6(1) :84–85, 1977.