

Complexité en temps PC 4.

Correction

1 La question $P = NP$?

On peut voir NP comme classe des langages tel qu'en tester l'appartenance revient à déterminer s'il existe un certificat court (polynomial). On peut relier cela à l'existence d'une preuve en mathématiques. En effet, dans son principe même, la déduction mathématique consiste à produire des théorèmes à partir d'axiomes. On s'attend à ce que la validité d'une preuve soit facile à vérifier : il suffit de vérifier que chaque ligne de la preuve soit bien la conséquence des lignes précédentes, dans le système de preuve. En fait, dans la plupart des systèmes de preuves axiomatiques (par exemple en arithmétique de Peano, en théorie des ensembles de *Zermelo-Fraenkel*), cette vérification se fait en un temps qui est polynomial en la longueur de la preuve.

Autrement dit, le problème de décision suivant est NP pour chacun des systèmes de preuve axiomatiques usuels \mathcal{A} .

THEOREMS = $\{ \langle \phi, \mathbf{1}^n \rangle \mid \phi \text{ possède une preuve de longueur } \leq n$
dans le système $\mathcal{A} \}$.

1. La théorie des ensembles de *Zermelo-Fraenkel* est un des systèmes axiomatiques permettant d'axiomatiser les mathématiques avec une description finie. (Même sans connaître tous les détails de la théorie des ensembles de *Zermelo-Fraenkel*) argumenter à un haut niveau que le problème THEOREMS est NP-complet pour la théorie des ensembles de *Zermelo-Fraenkel*.

Correction: *Indice : la satisfiabilité d'un circuit booléen est un énoncé.*

2. Qu'en déduit-on si $P = NP$?

Correction: *Autrement dit, en vertu du théorème cours qui montre que si $P = NP$ on peut produire en temps polynomial un certificat, la question $P = NP$ est celle (posée par Kurt Gödel) de savoir s'il existe un algorithme qui soit capable de produire la preuve mathématique de tout énoncé ϕ en un temps polynomial en la longueur de la preuve.*

3. A quoi se relie la question $NP = coNP$? (on rappelle qu'un problème est dans $coNP$ si son complémentaire est dans NP).

Correction: *La question $NP = coNP$, est reliée à l'existence de preuve courte (de certicats) pour des énoncés qui ne semblent pas en avoir : par exemple, il est facile de prouver qu'un circuit est satisfiable (on produit une valuation de ses entrées, que l'on peut coder dans une preuve qui dit qu'en propageant les entrées vers les sorties, le circuit répond 1). Par contre, dans le cas général, il n'est pas facile d'écrire une preuve courte qu'un circuit donné est non satisfiable. Si $NP = coNP$, il doit toujours en exister une : la question est donc reliée à l'existence d'un autre moyen de prouver la non-satisfiabilité d'un circuit, que les méthodes usuelles.*

On peut reformuler l'équivalent pour chacun des problèmes NP -complets évoqués.

2 NP-complétude sur \mathbb{R}

Montrer que dans $(\mathbb{R}, +, -, *, =, <)$, le problème de l'existence d'une racine pour un polynôme en n variables à coefficients réels et de degré total 4 est $NP_{(\mathbb{R}, +, -, *, =, <)}$ -complet.

Correction: *Prouvons tout d'abord que l'existence d'une solution à un système d'équations polynomiales est $NP_{(\mathbb{R}, +, -, *, =, <)}$ -complet.*

*Cela découle du fait que la satisfiabilité d'une formule rudimentaire est $NP_{(\mathbb{R}, +, -, *, =, <)}$ -complet : les formules rudimentaires sont des conjonctions finies de formules élémentaires du type $x = a$, $x = y + z$, $x = y - z$, $x = y * z$, $x > y \wedge z = 0$, $x \leq y \wedge y = 1$, $x = y \wedge z = 0$, $x \neq y \wedge y = 1$, $x = 0 \vee x = 1$, $x = \epsilon \vee y = \epsilon'$, où $\epsilon, \epsilon' \in \{0, 1\}$ ou encore $x = \epsilon \vee y = \epsilon' \vee z = \epsilon''$, où $\epsilon, \epsilon', \epsilon'' \in \{0, 1\}$.*

*On a $x = a$ ssi $x - a = 0$, $x = y + z$ ssi $y + z - x = 0$, $x = y - z$ ssi $y - z - x = 0$, $x = y * z$ ssi $y * z - x = 0$: donc les formules élémentaires du type $x = a$, $x = y + z$, $x = y - z$, $x = y * z$ se ramène à des équations du type $P = 0$ pour un polynôme P .*

Il suffit de se convaincre que chacune des formules des autres types peut se ramener à une équation polynomiale du même type.

Une première astuce est de voir que $x \geq 0$ si et seulement si $x = z^2$ pour un certain z .

Donc $x \leq y$ si et seulement si $y - x - z^2 = 0$ pour un certain z .

Une autre astuce est d'observer que $PQ = 0$ ssi $P = 0$ ou $Q = 0$.

Donc $x \leq y \wedge y = 1$ ssi $(y - x - z^2)(y - 1) = 0$ pour un certain z .

Une autre astuce est d'observer que $x \neq 0$ si et seulement si $xy = 1$ pour un certain y . Donc $x \neq y$ si et seulement si $(x - y)z - 1 = 0$ pour un certain z .

*On obtient par exemple $x > y \wedge z = 0$ ssi $z(y - x - t^2) = 0$ et $z[(z - y)*t' - 1] = 0$ pour un certain t et un certain t' (la première équation signifie $z = 0$ ou $x \geq y$, la seconde $z = 0$ ou $x \neq y$)*

On fait de même pour chaque autre formule élémentaire.

En appliquant ces astuces (et en observant que tout se fait bien en temps polynomial) on se ramène à une conjonction d'équations polynomiales du type $P_1 = 0, P_2 = 0, \dots, P_m = 0$ (ce que l'on peut voir comme un système), ce qui prouve que l'existence d'une solution à un système d'équations polynomiales est $\text{NP}_{(\mathbb{R}, +, -, *, =, <)}$ -complet.

Maintenant, pour prouver le résultat de l'exercice, on observe que le système $P_1 = 0, P_2 = 0, \dots, P_m = 0$ possède une solution si et seulement si l'unique polynôme $P = P_1^2 + P_2^2 + \dots + P_m^2 = 0$ possède une solution.

On remarque alors que le polynôme P obtenu de degré total 4.

3 Gonflement

1. Considérons la fonction $pad : M^* \times \mathbb{N} \rightarrow M^* \#^*$ qui est définie comme suit : $pad(s, l) = s \#^j$ où $j = \max(0, l - n)$, et n est la longueur de s . En d'autres termes, $pad(s, l)$ ajoute des $\#$ à la fin de s de telle sorte que la longueur du résultat soit au moins l .

Pour un langage A et une fonction $f : \mathbb{N} \rightarrow \mathbb{N}$ on définit

$$pad(A, f(n)) = \{pad(s, f(n)) \mid s \in A \text{ et } n \text{ est la longueur de } s\}$$

Montrer que si $A \in \text{TIME}(n^2)$ alors $pad(A, n^2) \in \text{TIME}(n)$.

Correction: Supposons $A \in \text{TIME}(n^2)$. Pour déterminer si un mot w est dans $A' = pad(A, n)$, on vérifie qu'il est de la forme $pad(s, n)$, et si cela est le cas, on simule l'algorithme pour A sur s .

Déterminer si $w \in A'$ se fait en temps $\mathcal{O}(|w|) + \mathcal{O}(|s|^2) = \mathcal{O}(|w|)$. A' est donc dans $\text{TIME}(n)$.

2. La classe EXPTIME (respectivement NEXPTIME) est la classe des problèmes solvables par un algorithme (resp. non-déterministe) en temps $\mathcal{O}(n^{p(n)})$, où $p(n)$ est un polynôme en n .

Montrer que $\text{P} \subset \text{NP} \subset \text{EXPTIME} \subset \text{NEXPTIME}$.

3. Utiliser la fonction pad pour montrer que si $\text{NEXPTIME} \neq \text{EXPTIME}$ alors $\text{P} \neq \text{NP}$.

Correction: Supposons $\text{P} = \text{NP}$. Soit $A \in \text{NTIME}(2^{n^c})$. Considérons $A' = pad(A, 2^{n^c})$. Par un raisonnement similaire à celui plus haut, A' est dans NP . Il est donc dans P , puisqu'on a supposé $\text{P} = \text{NP}$. Maintenant, pour déterminer si un mot w est dans A , il suffit de déterminer si $pad(w, 2^{n^c}) \in A'$, ce qui se fait en temps $\mathcal{O}(2^{n^c})$. On a donc, $A \in \text{EXPTIME}$, et donc $\text{EXPTIME} = \text{NEXPTIME}$.

4 Problèmes P-complets

On a défini en cours une relation de réduction \leq basée sur les fonctions calculables en temps polynomial.

1. Montrer que cette notion de réduction n'est pas pertinente pour parler de problèmes de P.
2. Proposer une notion de réduction basée sur les fonctions calculables en espace mémoire logarithmique. On admettra que ces fonctions sont closes par composition.
3. Montrer que le problème de la satisfiabilité des circuits reste NP-complet pour cette réduction.
4. Montrer que le problème CIRCUITVALUE, où l'on se donne un circuit booléen C et une entrée \bar{x} , et l'on veut déterminer si $C(\bar{x}) = 1$, est P-complet.
5. Montrer que le problème MONOTONECIRCUITVALUE, où l'on se donne un circuit booléen C sans porte \neg et une entrée \bar{x} , et l'on veut déterminer si $C(\bar{x}) = 1$, est P-complet.

Correction: MONOTONECIRCUITVALUE est clairement dans P (en particulier car MONOTONECIRCUITVALUE \leq_L CIRCUITVALUE via la fonction identité.

CIRCUITVALUE \leq MONOTONECIRCUITVALUE : on élimine les négations par les lois de Morgan : $\neg(x \vee y) = (\neg x) \wedge (\neg y)$ et $\neg(x \wedge y) = (\neg x) \vee (\neg y)$. On double le nombre de portes : on construit une fonction g et $\neg g$ pour chaque porte. Le circuit obtenu se calcule facilement à partir du premier, c'est-à-dire en espace $O(\log n)$.