

Complexité - Nicolas Ollinger

Alexandre Masson

15 Janvier 2013

Table des matières

1	non-déterminisme	3
2	19 Mars 2013	5
2.1	NP-Completude	5
3	26 Mars 2013	7
3.1	NP-complétude suite	7

1 non-déterminisme

$P = \cup_{k>0} DTIME(n^k)$ la classe des problèmes qu'on peut résoudre en temps polynomial.

$NP = \cup_{k>0} NTIME(n^k)$ la classe des problèmes vérifiables en temps polynomial

Rétrospective DFA : automate déterminisme , NFA non déterminisme.

ex : $(a + b)^*baba$ mot finissant par baba. Tout NFA non déterminisme, se code en DFA, mais au prix d'une complexité plus importante en nombre d'état.

Définition une MTND (MT non déterministe), est une MT avec comme table de transition :

$$T \subseteq (Q \times \Gamma^k)^{input} \times (Q \times \Gamma^k \times \{< -, v, - >\})^{output}$$

on redéfinie la relation de transition :

$c \vdash c'$ si T contient une transition applicable sur c qui produit c'

Une exécution d'une MTND partant d'une entrée u est une suite de transition valides de la configuration initiale associée à u à une configuration d'arrêt.

$$c_0 \vdash c_1 \vdash \dots \vdash c_n$$

Une MTND M accepte un mot $u \in \Sigma^*$ s'il existe une exécution acceptée pour u.

	MT	MTND
Rejet	l'exécution doit s'arrêter sur q_{non}	toutes les exécutions doivent s'arrêter sur q_{non}
Acceptation	l'exécution s'arrête sur q_{oui}	un seul état à oui renvoie oui

remarque A un ralentissement linéaire près, on peut supposer que les choix non déterministes sont au plus binaires : pour toute configuration c :

- ou bien c n'a pas d'image (arrêt)
- ou bien $c \vdash c'$ unique.
- ou bien $c \vdash c' \& c \vdash c''$

exemple le 3-color, entrée : un graphe $G=(V,E)$, sortie : G est-il 3-colorable
3Color est reconnu par une MTND dont les exécutions s'arrêtent en temps polynomial en la taille de l'instance.

la machine fonctionne en deux temps.

- Choix non déterministe d'une couleur par sommet ($O(|V|)$)
- Vérification déterministe que le coloriage choisi est valide : si oui accepte, sinon rejette.

Définition $NTIME(f(n))$ est la classe des langages reconnus par une MTND dont toutes les exécutions sont de longueur $\leq f(n)$ sur les entrées de taille n.

proposition Si $f(n) \leq g(n)$ alors $NTIME(f(n)) \subseteq NTIME(g(n))$
 $DTIME(f(n)) \subseteq NTIME(f(n))$
 $NTIME(f(n)) \subseteq DTIME(2^{O(f(n))})$

Idée : Il suffit de simuler les choix de la MTND par du backtracking

def $NP = \cup_{k>0} NTIME(n^k)$

$NEXP = \cup_{k>0} NTIME(2^{n^k})$

Question : Est ce que $P \subseteq NP$, **rq** : $P \subseteq NP$

prop NP est clos par union et intersection.

def $co\mathfrak{S} = \{\neg L \mid L \in \mathfrak{S}\}$ ou \mathfrak{S} est une classe de langages.

Un problème B est un **certificateur** pour un problème A si $\forall x \in \Sigma^*, x \in A \iff \exists y \in \Sigma^* \langle x, y \rangle \in B$ on dit que y est un certificat pour x.

thm NP la classe des langages qui possèdent un certificateur dans P avec certificats de longueur polynomiale en la longueur de l'instance.

ex 3Color $\in NP$, cf exemple antérieur. certificateur pour 3COLOR.

sur l'entrée $\langle G, C \rangle$ avec colorisation des sommets vérifier que les arêtes ont des couleurs distinctes à chaque extrémité. Ce certificat est dans P (car il ne faut que parcourir les arêtes pour vérifier une propriété). la taille du certificat est polynomiale.

Démonstration

i (dans ce sens, on connaît A de P, et B le certificateur, on essaie de construire un chemin qui accepte x dans la machine M, ce chemin s'appelle y, si l'on suit le chemin y pour arriver en quoi avec x en entrée, alors y est certificat et le certif existe, donc B est certificateur.)

Soit A un langage qui possède un certificateur B $\in P$ avec certificat de longueur $\leq p(n)$ sur les instances de longueur n ou p est polynomiale en n.

montrons que A est dans NP en construisant une MTND M qui reconnaît A comme suit :

- choisir grâce au non déterminisme un mot y de longueur $\leq p(|x|)$
- exécuter une MT qui reconnaît B sur l'entrée $\langle x, y \rangle$
- accepter si (2) accepte

M s'exécute bien en temps polynomial en $|x|$ et M accepte x ssi $\exists y : |y| \leq p(n)$ et $\langle x, y \rangle \in B$, si $x \in A$.

ii (dans ce sens, on sait que le Y sera un certificat, puisqu'on choisit un chemin acceptant de x dans l'exécution de M, donc si B répond ok, vu que le Y est ok, cela veut dire que B est bon)

Soit $A \in NP$ On va construire un certificat $B \in P$ pour A avec certificats polynomiaux.

Soit M un MTND qui reconnaît A.

On utilise les exécutions acceptantes de M comme certificats et B se contente de

vérifier l'exécution. Sur l'entrée $\langle x, y \rangle$ B vérifie que Y est une exécution acceptable pour M sur l'entrée x. B est bien un certificateur dans P avec certificat polynomiaux pour A.

Np est la classe des problèmes vérifiables en temps polynomial.

3Color : E : graphe G, S : est il 3Colorable, certif : coloration, Verif : pas d'arete monochrome.

Hamilton : E/S idem : certif : une permutation des sommets, i.e le cycle, verif : les aretes existent bien.

CLIQUE(sous graphe complet du graphe) Entrée : un graphe et un entier k : certif : la clique, verif : les aretes existe =elles

SAT (clause satisfiables) entrée : formule Φ en CNF, sortie : Φ est elle satisfaisable ?

variable : x,y,z, littéraux : var ou neg.var, clause : disjonction de littéraux, formules : conjonction de clauses, **certificat** : une affectation qui satisfait Φ , verif : évaluer Φ **3SAT** SAT où Φ ont des clause d'ua moins 3 littéraux.

COMPOSITE entrée : entier n en binaire, question : existe il p et q > 1 tels que $n=pq$?, certif : la paire $\langle p, q \rangle$ en binaire. verif : multiplier p par q et comparer à n.

2 19 Mars 2013

2.1 NP-Completude

Au menu : théorème de Cook-Levin $SAT \in P \iff P=NP$

remarque sens \leq trivial car $SAT \in NP$.

Définition $A \leq_p B$ si $\exists f : \Sigma^* \rightarrow \Sigma^*$ totale calculable en temps polynomiale telle que $\forall x \in \Sigma^*, X \in A \iff f(x) \in B$

prop si $A \leq_p B$ et $B \in P$ alors $A \in P$. Que B soit dans P implique qu'il existe un algo poly qui résout B,

def on dit que A est NP-difficile si quelquesoit le problème que je prend dans NP, ce problème est plus simple que A. ($\forall B \in NP, B \leq_p A$)

A est NPComplet si

A est NP-difficile **ET** $A \in NP$. (**NE PAS L'OUBLIER**)

Remarque : si A est NP-complet, et $A \in P$, alors $P=NP$. En effet soit $B \in NP$, comme A est NP-difficile, d'après la def, $B \leq_p A$ et d'après la proposition

ci dessus, $B \in P$. Donc $NP \subseteq P$, i.e $P=NP$.

thm Cook-Levin SAT est NP-complet. On sais que $SAT \in NP$, on va montrer que $\forall A \in NP, A \leq_p SAT$
soit $A \in NP$. soit M une MTND totale qui reconnaît A en temps polynomial $p(n)$.

On peut supposer sans perte de généralité que M à un seul ruban, à choix binaire, à ruban semi infini, avec arrêt sur la case 0, exactement au temps $p(n)$. A l'instant 0, le ruban contient le mot de taille N , si le temps d'arrêt est $p(n)$, alors le ruban ne peut être de taille plus grand que $p(n)$, car on parcourt une case par étape, donc pour un mot donnée, on a obtenu des q_{oui} ou des q_{non} , $x \in A$ ssi \exists un digne-espace temps de M sur l'entrée X , avec q_{oui} en $(0, p(n))$.

Idée : Pour montrer que $A \leq_p SAT$ on construit f , fonction de réduction, qui a un mot $x \in \Sigma^*$ associe une formule CNF $\Phi_x^{M,p}$ qui code les diagrammes espace-temps de M sur l'entrée x de sorte que $\Phi_x^{M,p} \in SAT$ ssi le diagramme possède des choix non-déterministes acceptant, c'est à dire avec q_{oui} en $(0, p(x))$.

On va découper $\Phi_x^{M,p}$ en plusieurs parties :

$$\Phi_{sim}^{M,p(|x|)}(calcul\ de\ la\ MT) \wedge \Phi_{input}^{M,P,x}(entrée) \wedge \Phi_{accept}^{M,P(|x|)}(sortie)$$

Pour chaque case du diagramme, elle sont codées par les variables booléennes, $x_1^{ij}, \dots, x_k^{ij}$, en binaire.

les codes valides sont vérifiés par une formule CNF $\Phi_{val}(x_1^{ij}, \dots, x_k^{ij})$

$$\Phi_{sim}^{M,p(|x|)} = \bigwedge_{i=0}^N \bigwedge_{j=0}^N \Phi_{val}(x_1^{ij}, \dots, x_k^{ij}) \wedge \Phi_{trans}^{M,N}$$

la variable C_t code le choix non-det à l'instant t . (si le contenu est loin de la tête, il n'a aucune chance de changer entre t et $t+1$).

$$\Phi_{trans}^{M,N} = \bigwedge_{i=0}^N \bigwedge_{j=0}^N \Phi_{delta}^{M,N}(x_1^{i-1j}, \dots, x_k^{i-1j}, x_1^{ij}, \dots, x_k^{ij}, x_1^{i+1j}, \dots, x_k^{i+1j}, x_1^{i-1j+1}, \dots, x_k^{i-1j+1}, x_1^{ij+1}, \dots, x_k^{ij+1}, x_1^{i+1j+1}, \dots, x_k^{i+1j+1})$$

où Φ_{delta}^M vérifie que que le rectangle 3x2 cellules communiqués est une transition valide pour M

On obtient une formule CNF $\Phi_{sim}^{M,N}$ à kN^2 variables et de taille $O(N^2)$. On est tous convaincu de savoir produire $\Phi_{sim}^{M,N}$ en temps poly.

$\Phi_{input}^{M,P,x}$ vérifie que la ligne 0 code via la config initial

$$\Phi_{input}^{M,P,x} = \Phi_{case}^M(q_0, x_0), x_1^{00}, \dots, x_1^{00} \wedge \bigwedge_{i=1}^{|x|-1} \Phi_{case}^M(x_i, x_1^{i0}, \dots, x_k^{i0}) \wedge \bigwedge_{i=|x|}^{p(|x|)} (B, x_1^{i0}, \dots, x_k^{i0})$$

formule de ligne $O(n)^i = |x|$ est calculable en temps polynomial

$\Phi_{accept}^{M,N}$ vérifie que la case $(0, n]$ contient bien q_{oui}

la formule $\Phi_x^{M,N}$ est donc calculable en temps polynomial en $|x|$ et est satisfiable ssi il existe des choix non déterministes pour lesquels M accepte x , i.e $x \in A$.

Si A est NP-difficile et $A \leq_p B$ alors B est NP difficile, car B est plus compliqué que A., pour montrer que BLOP est np complet, on montre que blop in NP et blop \leq_p BLOP , comme blop plus compliqué que blip et blip np , alors blop np

prop 3SAT et variante est NP-complet,
une formule CNF avec jusqu'à 3 littéraux par clause
idem mais avec exactement
idem mais avec 3 littéraux distincts

démonstration

3SAT in NP : cf cours précédent

3SAT est NP-difficile, montrons $SAT \leq_p 3SAT$

Exhibons F calculable en temps polynomial telle que

$\forall \Phi, \Phi \in SAT \iff f(\Phi) \in 3SAT$

no doit trouver un moyen de transformer les littéraux de taille n en littéraux de taille 3 avec des conjonctions entre eux, pour respecter CNF.

F remplace chaque clause $(l_1 l_2 v \dots l_k)$ par la formule $(l_1 l_2 v \alpha_2) \wedge (\bar{\alpha}_2 l_3 n \alpha_3) \wedge \dots \wedge (\alpha_{k-2} v l_{k-1} l_k)$ où $\alpha_2, \dots, \alpha_{k-2}$ sont k-3 nouvelles variables, et cette formule est satisfiable ssi la clause l'est.

3 26 Mars 2013

3.1 NP-complétude suite

Lemme $3_{\neq} SAT \leq_p 3_{ex} SAT \leq_p 3_{\leq} SAT$

Démo Ici à chaque fois $A \leq_m B$ à prouver

avec $A \subseteq B$ et on sait séparer A de B en temps polynomial.

Proposition $3_{\leq} SAT \leq_p 3_{\neq} SAT$

Proposition $SAT \leq_p 3_{\leq} SAT$

Corollaire 3SAT est NP-Complet

Démonstration Astuce : la clause (PvQ) où P et Q sont des disjonctions de littéraux est satisfiable si et seulement si $(Pvx) \wedge (\bar{x}vQ)$ est satisfiable ou x n'apparaît ni dans P ni dans Q.

\implies Supposons (PvQ) satisfiable . Fixons une instantiation σ qui satisfait (PvQ) . Alors on construit une instantiation σ' qui satisfait $(Pvx) \wedge (\bar{x}vQ)$ ainsi

- si $\sigma(P)$ est vraie on pose $\sigma'(x) = \text{faux}$
- si $\sigma(P)$ est faux alors $\sigma(Q)$ est vrai on pose $\sigma'(x) = \text{vrai}$
- $\forall y, y \neq x, \sigma'(y) = \sigma(y)$

\leq Supposons $(Pvx) \wedge (\bar{x}vQ)$ satisfait par une instantiation σ alors σ satisfait aussi $(P \wedge Q)$. En effet si $\sigma(X)$ est vraie alors $\sigma(Q)$ est vraie, et si $\sigma(X)$ est faux alors $\sigma(P)$ est vraie.

Pour montrer $3\leq SAT \leq_p 3\neq SAT$, il faut associer à toute formule ϕ avec jusqu'à 3 lit/clauses, une formule $f(\phi)$ avec exactement 3 lit \neq /clause de sorte de ϕ satisfiable $\Leftrightarrow f(\phi)$ sat.

Il faut faire grossir les clauses à ou 2 lit \neq ON applique l'astuce une ou deux fois. Cette transformation se calcule en temps polynomial.

Pour montrer qu $eSAT \leq_p 3\leq SAT$

ϕ quelconque

$f(\phi)$ jusqu'à 3 lit/clause

on applique l'astuce pour faire maigrir les clauses :

$(l_1 v l_2 v \dots v l_k)$ on l'applique k-1 fois $\rightarrow (l_1 v \alpha_1) \wedge (\bar{\alpha}_1 v l_2 v \alpha_2) \wedge \dots \wedge (\alpha_{k-1} v l_k)$

cette transformation est calculable en temps polynomial donc $SAT \leq_p 3SAT$

CIRCUIT-SAT

entrée : un circuit boolean C avec des portes NON,ET,OU, et des entrées et une sortie

question : existe-t-il une affectation des entrées qui force la sortie de C à vrai

CIRCUIT-SAT \in NP pour décoder CIRCUIT-SAT, il suffit de deviner une instantiation des entrées et d'évaluer le circuit, en temps polynomial.

Proposition CIRCUIT-SAT est NP-Complet.

Démonstration

1 CIRCUIT-SAT \in NP (axiome)

2 $3SAT \leq_p CIRCUIT - SAT$

On code une formule ϕ de 3SAT par un circuit avec :

- une entrée /variable,
- une porte \neq /littéral négatif
- deux portes OU par clauses
- k-1 portes ET entre les k clauses

Cette transformation est une réduction many-one PTIME de 3SAT en CIRCUIT-SAT.

VERTEX-COVER

Entrée : $G=(V,E)$ un graphe et k appartient à N

question : existe-t-il un sous-ensemble $U \subseteq V$, U de taille k tel que toute arête

Proposition VERTEX-COVER est NP-Complet

Prove-it !

i VERTEX-COVER \in NP : un certificat est un sous ensemble de sommets qui couvre les arêtes de G

ii $M_q 3SAT \leq_p VERTEX-COVER$

Soit ϕ une instance de 3SAT on construit $f(\phi)$ à l'aide des gadgets suivants :

- chaque variable x est représentés par : $(x) - (\bar{x})$
- chaque clause $(l_1 v l_2 v l_3)$ est représentée par : un triangle
- on relie chaque sommet des gadgets de clauses aux sommets des gadgets de variables de même label

– $k = 2(n+m)$ avec $m = \# \text{clause}$ et $n = \# \text{variables}$.
 \Rightarrow Si ϕ est satisfiable, on obtient une couverture de $f(\phi)$ de taille $k=2m+n$
ainsi :
on fixe une instance σ qui satisfait ϕ .
– si $\sigma(x)$ est vrai, on prend le sommet x de l'haltere x sinon le sommet \bar{x}
– pour chaque clauses $(l_1 v l_2 v \dots v l_n)$ avec $\sigma(l_1) = \text{vrai}$, on prend les sommets l_2 et l_3
les aretes internes de chaque gadget sont couverts. une arete d'un littéral vrai est couverte par le littéral lui-meme et une arete d'un littéral faux est couverte par la clause

\Leftarrow Soit U une k -couverture de $f(\phi)$, elle couvre les haltères donc, comme elle couvre chaque haltère, U contient au moins un sommet par haltere (soit la var et la neg de la var).
elle couvre les triangles, comme elle couvre tous les triangles, alors U contient au moins deux sommets /triangle.
comme $\#U = k = 2m+n$ la couverture est de la forme précédente et l'instanciation associée satisfait ϕ

SUBSET-SUM

SUBSET-SUM

entrée : des entiers x_1, x_2, \dots, x_n de \mathbb{N} et un objectif S de \mathbb{N}
question: existe-t-il un sous-ensemble des x_i qui somment à S ?
peut on prendre un sous ensemble dont la somme des éléments sera S ?

proposition SUBSET-SUM est NP-Complet

let's Go!!

- SUBSET-SUM : un certificat qui est le choix I des x_i qui somment à S
- $3\text{SAT} \leq_p \text{SUBSET-SUM}$

soit ϕ une instance de 3SAT, on construit polynomialement $f(\phi)$ une instance de SUBSET-SUM qui est satisfiable ssi ϕ l'est.

ON code ϕ à n variables et m clauses par des nombre N_i décimaux à $m+n$ chiffres de la manière suivantes :

- on les decomposent en deux zones
- celle de gauche code les variables,
- celle de droite code les clauses
- à chaque variable x_i on associe un entier N_{2i} et N_{2i+1} qui est contruit de la manière suivante : que des zéro sauf, un 1 pour le chiffre qui correspond à x_i et un 1 si x_i apparaît dans la clause (ou 1 si x_i n'apparait pas pour N_{2i+1} qui code \bar{x})