Définitions pour le cours de complexité et approximation

Pierre-François Dutot

7 septembre 2009

Plan

Problèmes $\mathcal{NP}$-complets

Définition

Π est $\mathcal{NP}$-complet ssi :

- Π est dans $\mathcal{NP}$,
- tout problème Π' de $\mathcal{NP}$ se réduit polynomialement à Π

$$\forall \Pi' \in \mathcal{NP},\ \Pi' \propto \Pi$$

Algorithmes pseudo-polynomiaux

Définition

Un algorithme est dit pseudo-polynomial si son temps d'exécution sur une instance x est polynomialement borné par $|\mathrm{x}|$ et $\mathrm{Max}(\mathrm{x})$.

Problèmes $\mathcal{NP}$-complet au sens fort

Définition

Pour un problème de décision Π et un polynôme p, on note Π_p le sous-problème de Π restreint aux instances x telles que $\mathit{Max}(x) \leqslant p(|x|)$.

Définition

Un problème $\Pi \in \mathcal{NP}$-complet est $\mathcal{NP}$-complet au sens fort ssi il existe un polynôme p tel que Π_p est $\mathcal{NP}$-complet.

Réduction entre problèmes $\mathcal{NP}$-complets sens fort

Définition

Π est $\mathcal{NP}$-complet au sens fort si il existe Π' $\mathcal{NP}$-complet au sens fort, tel que Π' se réduit polynomialement à Π ($\Pi' \propto \Pi$), avec f la fonction de transformation d'une instance de Π' en une instance de Π vérifiant :

- f est calculable en temps polynomial en $\text{Max}(x')$ et $|x'|$.
- il existe un polynôme p tel que pour tout x', $p(|f(x')|) \geqslant |x'|$
- il existe un polynôme q tel que $\text{Max}(f(x')) \leqslant q(\text{Max}(x'), |x'|)$

Problèmes d'optimisation

Un problème d'optimisation Π sera défini pour nous par la donnée de :

- Un ensemble $\mathcal{I}$ d'instances.
- Une fonction S telle que pour tout $x \in \mathcal{I}$, $S(x)$ représente l'ensemble des solutions réalisables pour x.
- Une fonction m à valeur entière définie sur tous les couples $x \in \mathcal{I}$ et $y \in S(x)$. Cette mesure m représente la fonction objectif d'une solution y pour l'instance x.
- Un $\mathtt{objectif} \in \{\mathtt{min}, \mathtt{max}\}$ précisant si Π est un problème de minimisation ou de maximisation.

NPO

Définition

Un problème Π d'optimisation est dans la classe NPO ssi :

- Les instances $\mathcal{I}$ peuvent être reconnues en temps polynomial.
- Il existe un polynôme p tel que pour tout $x \in \mathcal{I}$ et $y \in S(x)$, $|y|$ est borné par $p(|x|)$. De plus il est décidable en temps polynomial si un mot y appartient à $S(x)$.
- La mesure m est une fonction calculable en temps polynomial en $|x|$.

$\mathcal{APX}$

Soit r une suite à valeur dans $[1; +\infty[$. Un algorithme a est une $r(n)$-approximation pour le problème Π si :

- Pour chaque instance $x \in I$, a délivre une solution $y_a \in S(x)$.
- De plus y_a vérifie $m^*(x)/r(|x|) \leqslant m(x, y_a) \leqslant m^*(x) * r(|x|)$

On dit que $r(n)$ est la (performance) garantie de l'algorithme.

$\mathcal{APX}$

La classe $\mathcal{APX}$ est constituée des problèmes de NPO admettant une $O(1)$-approximation polynomiale en $|x|$.

$\mathcal{PTAS}$

Définition

Un schéma d'approximation polynomial (ptas, *Polynomial Time Approximation Scheme*) est une famille $(A_\epsilon)_{\epsilon>0}$ d'algorithmes polynomiaux en $|x|$ telle que pour tout $\epsilon > 0$, A_ϵ est une $(1+\epsilon)$-approximation.

La classe $\mathcal{PTAS}$ est constituée des problèmes de NPO admettant un ptas.

$\mathcal{FPTAS}$

Définition

Un schéma d'approximation pleinement polynomial (fptas, *Fully Polynomial Time Approximation Scheme*) est un algorithme $A(\epsilon)$ polynomial en $|x|$ et en $1/\epsilon$ telle que pour tout $\epsilon > 0$ fixé, $A(\epsilon)$ est une $(1+\epsilon)$-approximation.
La classe $\mathcal{FPTAS}$ est constituée des problèmes de NPO admettant un ptas.

$\mathcal{NP}$-complet sens fort vs. FPTAS

Vrai si $\mathrm{Opt}(x)$ à valeur entière et pour tout $x \in I$:

- $\mathrm{Opt}(x) \leqslant p(\mathrm{Max}(x))$ [Papadimitriou]
- $\mathrm{Opt}(x) \leqslant p(|x|, \mathrm{Max}(x))$ [Garey & Johnson, J.F. Rey]
- $\mathrm{Opt}(x) \leqslant p(|x|)$ [dérivé de $\mathcal{NPO}$-PB]

Comment obtenir un algorithme d'approximation ? [Woeginger]

En simplifiant les instances :

- arrondi
- fusion
- découpe
- alignement

En structurant la sortie (par exemple en considérant les solutions en ayant fixé un certain nombre de choix).

En structurant le fonctionnement de l'algorithme.

Machine de Turing probabiliste

Une PTM est une machine de Turing dotée :

- D'un ruban supplémentaire sur lequel un générateur aléatoire ajoute un nouveau bit 0 ou 1 à chaque transition.
- La tête de lecteur du ruban aléatoire est automatiquement placée sur le nouveau bit à chaque transition.
- Les bits 0 et 1 sont générés avec une probabilité $\frac{1}{2}$ chacun.

Temps d'exécution d'une PTM

On définit le temps d'exécution d'une PTM pour l'entrée w comme :

Temps d'exécution

$$\mathtt{t}(w) = \begin{cases} +\infty & \text{si au moins une des exécutions possibles ne termine pas} \\ \max_{exec}(\mathtt{t}) & \text{le plus grand temps d'exécution} \end{cases}$$

Classe $\mathcal{PP}$ (algorithmes de Monte-Carlo)

L est un langage de $\mathcal{PP}$ si il existe une PTM T dont le temps d'exécution est majoré par un polynôme tel que :

$$w \in \mathrm{L} \Longleftrightarrow \mathsf{Proba}^{\mathsf{T}}_{\mathsf{acc}}(w) > \frac{1}{2}$$

$\mathcal{NP} \subseteq \mathcal{PP}$

Classe $\mathcal{BPP}$

L est un langage de $\mathcal{BPP}$ si il existe une PTM T dont le temps d'exécution est majoré par un polynôme tel que :

$$w \in L \Longleftrightarrow \mathsf{Proba}^{T}_{acc}(w) \geqslant 1 - c > \frac{1}{2}$$

$$w \notin L \Longleftrightarrow \mathsf{Proba}^{T}_{acc}(w) \leqslant c < \frac{1}{2}$$

Où c est une constante.
$\mathcal{BPP} = \text{co-}\mathcal{BPP}$

Classe $\mathcal{RP}$

L est un langage de $\mathcal{RP}$ si il existe une PTM T dont le temps d'exécution est majoré par un polynôme tel que :

$$w \in \mathrm{L} \Longleftrightarrow \mathsf{Proba}^{\mathsf{T}}_{\mathsf{acc}}(w) > \frac{1}{2}$$

$$w \notin \mathrm{L} \Longleftrightarrow \mathsf{Proba}^{\mathsf{T}}_{\mathsf{acc}}(w) = 0$$

Classe $\mathcal{ZPP}$ (algorithmes de Las Vegas)

La classe $\mathcal{ZPP}$ est l'intersection des classes $\mathcal{RP}$ et co-$\mathcal{RP}$. C'est la classes des langages acceptés avec une probabilité d'erreur nulle et l'espérance du temps de calcul polynomiale.

Et l'interactivité ?

Une machine avec oracle possède une bande supplémentaire, deux états spéciaux $q_{question}$ et $q_{réponse}$ et une transition telle que le mot sur la bande supplémentaire est lu par un oracle et remplacé par un autre lors de la transition. En particulier le mot de réponse peut se résumer à un seul bit : 0 (rejet) ou 1 (acceptation).

PCP

Une machine PCP est composée de trois machines :

- T_1 est une PTM qui génère pour une entrée w des mots de requêtes $q_1, \ldots, q_m$ en utilisant un mot r aléatoire
- T_2 est une TM avec oracle qui récupère un mot y d'acceptation des requêtes
- T_3 est une TM déterministe qui prend pour entrée w, r et y

On note $\mathsf{PCP}(r(n), q(n))$ la classe des langages acceptés par les machines PCP utilisant au plus $O(r(n))$ bits aléatoires et $O(q(n))$ requêtes.

Langage accepté

Un langage L est accepté par une machine PCP ssi il existe un oracle $\mathtt{pi}^0$ tel que :

- $w \in \mathrm{L} \Longrightarrow \mathtt{Proba}^{\pi^0}_{\mathtt{acc}}(w) = 1$
- $w \notin \mathrm{L} \Longrightarrow$ pour tout oracle π on a $\mathtt{Proba}^{\pi}_{\mathtt{acc}}(w) \leqslant \frac{1}{2}$

Théorème PCP

Par extension on note peut remplacer les fonctions $r(n)$ et $q(n)$ par des classes de fonction.

$\mathsf{NP} = \mathsf{PCP}(0, \mathtt{poly})$ simplement en demandant le certificat complet.

$\mathsf{NP} \subseteq \mathsf{PCP}(\mathtt{poly}, 1)$ et même $\mathsf{NP} = \mathsf{PCP}(\log, \mathsf{O}(1))$