Le problème du Sac à Dos

Johanne Cohen

Johanne.Cohen@loria.fr

Laboratoire LORIA

Énoncé du problème

Instance:

$S = \{a_1, \ldots, a_n\}$ un ensemble d'objets

ayant une taille $s(a_i) \in \mathbb{Z}$ et un bénéfice $profit(a_i) \in \mathbb{Z}$

un sac à dos de taille B

Objectif:

Trouver un sous ensemble de S tel que sa taille est bornée par B et tel que son bénéfice soit maximum.

Theoreme 1: Le problème du Sac à Dos est NP-complet.

Déf. d'un Algo. pseudo-polynomial.

- $|\mathcal{I}|$: taille de l'instance $\mathcal{I}$
- $|\mathcal{I}_u|$: taille de l'instance $\mathcal{I}$ où les entiers sont codés en unaire.

Définition Un algorithme pour un problème Π est pseudo-polynomial si son temps d'exécution pour une instance $\mathcal{I}$ est borné par un polynôme en $|\mathcal{I}_u|$.

Déf. d'un Algo. PTAS

- Soit Π un problème optimisation NP-difficile avec une fonction de coût f_Π

Définition Un algorithme $\mathcal{A}$ est un schéma d'approximation pour Π si étant donnée une entrée $(\mathcal{I}, \epsilon)$ avec $\mathcal{I}$ instance de Π et $\epsilon > 0$ une erreur fixée à l'avance , il retourne une solution s telle que

- $f_\Pi(I, s) \leq (1 + \epsilon)OPT$ si Π un problème de minimisation.
- $f_\Pi(I, s) \geq (1 - \epsilon)OPT$ si Π un prb. de maximisation.

Un algorithme $\mathcal{A}$ est un schéma d'approximation à tps polynomial si pour tout $\epsilon > 0$ fixé le temps d'exécution est borné par un polynôme en la taille de l'instance.

Déf. d'un Algo. FPTAS

Remarque: tps d'éxécution ne devra pas augmenter trop rapidement pendant que ϵ diminue.
Idéalement

Si $\epsilon \searrow$ d'1 facteur constant alors tps d'éxécution $\nearrow$ d'1 facteur constant.

tps d'éxécution dépendant de la taille de l'instance + $1/\epsilon$

- Soit Π un problème optimisation NP-difficile avec une fonction de coût f_Π

Un algorithme $\mathcal{A}$ est un schéma d'approximation entièrement à tps polynomial si pour chaque $\epsilon > 0$ fixé le temps d'exécution est borné par un polynôme en la taille de l'instance et $1/\epsilon$.

Algorithme pseudo-polynomial $(\mathcal{B})$

Borne supérieure pour le bénéfice

Chaque solution a un bénéfice $\leq nP$ avec $P = max_{a \in S} profit(a)$,

Relation entre les bénéfices et les tailles

Pour $\forall i \in \{1, \ldots, n\}$, $\forall p \in \{1, \ldots, nP\}$

- $S_{i,p}$: sous ensemble de $\{a_1, \ldots, a_i\}$ tel que son bénéfice $= p$ et tel que sa taille soit minimum.
- $A_{i,p}$: taille de $S_{i,p}$. ($A(i,p) = \infty$ si un tel ensemble n'existe pas.

Algorithme pseudo-polynomial $(\mathcal{B})$

- $S_{i,p}$: sous-ens. de $\{a_1, \ldots, a_i\}$ tel que bénéfice $= p \oplus$ taille minimum.
- $A(i,p)$: taille de $S_{i,p}$.

formule de récurrence

$$A(i+1,p) = \begin{cases} min\{A(i,p), size(a_{i+1}) + A(i, p - profit(a_{i+1}))\} & si\ profit(a_{i+1}) \\ A(i,p) & sinon \end{cases}$$

Remarque:

- algorithme en $O(n^2 P)$ opérations
- le bénéfice max $= \{p | A(n,p) \leq B\}$

$S' \leftarrow \mathcal{B}(S, s, profit, B)$

Instance: $S = \{a_1, a_2, a_3, a_4, a_5\}$, $s = \{3, 5, 3, 2, 2\}$, $profit = \{2, 10, 2, 3, 3\}$, $B = 20$

1. $A(1, 0) = 0$ avec $S_{1,0} = \emptyset$, $A(1, 2) = 3$ avec $S_{1,2} = \{a_1\}$
2. $A(2, 0) = 0$ avec $S_{2,0} = \emptyset$, $A(2, 2) = 3$ avec $S_{2,2} = \{a_1\}$, $A(2, 10) = 5$ $S_{2,10} = \{a_2\}$, $A(2, 12) = 8$ $S_{2,12} = \{a_1, a_2\}$,
3. $\ldots$

Algorithme FPTAS noté $\mathcal{A}$.

Entrée: un ensemble d'objets $S = \{a_1, \ldots, a_n\}$

2 fcts taille $s : S \to \mathbb{Z}$, bénéfice $profit : S \to \mathbb{Z}$

un sac à dos de taille B

Sortie: un sous-ensemble d'objets de S

Algorithme:

1. $K = \frac{\epsilon P}{n}$
2. Pour chaque objet a_i, définir $profit'(a_i) = \lfloor \frac{profit(a_i)}{K} \rfloor$
3. S'= solution obtenue par l'algorithme pseudo-polynomial avec $(S, s, profit', B)$
4. retourner S'

Illustration

Instance: $S = \{a_1, a_2, a_3, a_4, a_5\}$, $s = \{3, 5, 3, 2, 2\}$,
$profit = \{10, 100, 19, 20, 29\}$, $B = 20$

Paramètre: $\epsilon = 0.5$

Algorithme

1. $K = \frac{0.5 \times 100}{5} = 10$
2. $profit' = \{1, 10, 1, 2, 2\}$
3. $S' \leftarrow \mathcal{B}(S, s, profit', B)$
4. $S' = \{a_1, a_2, a_4\}$ avec $profit'(S') = 13$ et $profit(S') = 132$
5. $M^* = \{a_2, a_3, a_5\}$ avec $profit'(M^*) = 15$ et
   $profit(M^*) = 148$
6. $(1 - \epsilon)profit(M^*) \approx 75 < 132 < 149$

Remarque

Proposition 2: Soit S' la solution de l'algorithme.

$$profit(S') \geq (1-\epsilon) \times OPT$$

Preuve: Soit M^* l'ensemble optimal.

1. $\forall a \in S, K \times profit'(a_i) \leq profit(a_i) < K \times profit'(a_i) + K$
2. $profit(M^*) - K \times profit'(M^*) \leq nK$

Concentrons nous sur l'étape utilisant l'algo. $(\mathcal{B})$

- il retourne 1 solution telle que $profit'(S') \geq profit'(M^*)$.
- $profit(S') \geq K \times profit'(M^*) \geq profit(M^*) - nK (= OPT - \epsilon P)$.
- $profit(S') \geq (1-\epsilon) \times OPT$ car $OPT \geq P$

Résultat

Theorème 2: L'algorithme $\mathcal{A}$ est un algorithme entièrement polynomial d'approximation pour le problème du sac à dos.

Preuve: Soit M^* l'ensemble optimal.

1. Par la proposition **??**, on a $profit(S') \geq (1-\epsilon) \times OPT$
2. le temps d'exécution de l' algorithme est:
   (a) pas 1: $0(1)$ opération (calcul de K)
   (b) pas 2: $0(n)$ opérations (définition de $profit'$
   (c) pas 3: $0(n^2 P' = n^2 \lfloor \frac{P}{K} \rfloor)$ opérations (exécution de l'algo $\mathcal{B}$).

Résultat

Theorème 2: L'algorithme $\mathcal{A}$ est un algorithme entièrement polynomial d'approximation pour le problème du sac à dos.

Preuve: Soit M^* l'ensemble optimal.

1. Par la proposition **??**, on a $profit(S') \geq (1-\epsilon) \times OPT$
2. le temps d'exécution de l' algorithme est:
   - $O(n^2 \lfloor \frac{P}{K} \rfloor) = O(n^2 \lfloor \frac{n}{\epsilon} \rfloor)$
   - polynomial en n et en $1/\epsilon$.

□

Fortement NP-complet $/\exists$ d'un FPTAS

Theorème 2: Soit p un polynôme et Π un problème NP-difficile de minimisation telle que la fonction de coût peut être évalué pour chaque instance $\mathcal{I}$, $OPT(\mathcal{I}) < p|\mathcal{I}_u|$.

Si Π admet un FPTAS alors il admet aussi un algorithme pseudo-polynomial.

Preuve

Supposons qu'il existe un FPTAS pour Π tel que son tps d'exécution pour une instance $\mathcal{I}$, une erreur ϵ est $q(|\mathcal{I}|, 1/\epsilon)$ avec q un polynôme.

- soit $\mathcal{I}$ une instance, soit $\epsilon = 1/p|\mathcal{I}_u|$
- Soit S la solution produite par FPTAS ayant en entrée $(\mathcal{I}, \epsilon)$

  1. $(1+\epsilon)OPT(I) < OPT(I) + \epsilon p|\mathcal{I}_u| (= OPT(I) + 1)$.
- Donc FPTAS force à obtenir la solution optimale:

  temps d'exécution $= q(|\mathcal{I}|, 1/\epsilon)$ = polynomial en $|\mathcal{I}_u|$

Par conséquence:

Théorème 2: Soit p un polynôme et Π un problème NP-difficile de minimisation telle que la fonction de coût peut être évalué pour chaque instance $\mathcal{I}$, $OPT(\mathcal{I}) < p|\mathcal{I}_u|$. Si Π est fortement NP-complet alors il n'admet pas un FPTAS en supposant que $P \neq NP$.

Preuve:

Si Π admet un FPTAS, alors il admet un algorithme pseudo-polynomial.

$\Longrightarrow$ il n'est pas fortement NP-complet,

$\Longrightarrow$ ceci mène à 1 contradiction en supposant que $P \neq NP$.

□