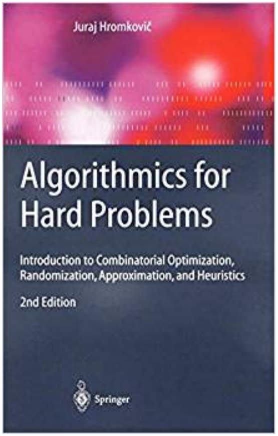
4-12 Approximation Algorithms

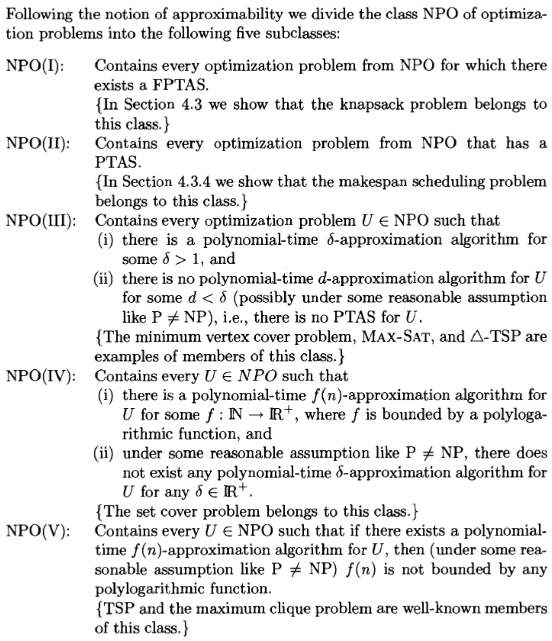
Hengfeng Wei

hfwei@nju.edu.cn

June 03, 2019


Juraj Hromkovič
Algorithmics for Hard Problems
Introduction to Combinatorial Optimization, Randomization, Approximation, and Heuristics
2nd Edition
Springer

2条亚马逊美国的评论



Following the notion of approximability we divide the class NPO of optimization problems into the following five subclasses:

NPO(I): Contains every optimization problem from NPO for which there exists a FPTAS.
{In Section 4.3 we show that the knapsack problem belongs to this class.}

NPO(II): Contains every optimization problem from NPO that has a PTAS.
{In Section 4.3.4 we show that the makespan scheduling problem belongs to this class.}

NPO(III): Contains every optimization problem $U \in$ NPO such that
- (i) there is a polynomial-time δ-approximation algorithm for some $\delta > 1$, and
- (ii) there is no polynomial-time d-approximation algorithm for U for some $d < \delta$ (possibly under some reasonable assumption like P $\neq$ NP), i.e., there is no PTAS for U.

{The minimum vertex cover problem, MAX-SAT, and $\triangle$-TSP are examples of members of this class.}

NPO(IV): Contains every $U \in NPO$ such that
- (i) there is a polynomial-time $f(n)$-approximation algorithm for U for some $f : \mathbb{N} \rightarrow \mathbb{R}^+$, where f is bounded by a polylogarithmic function, and
- (ii) under some reasonable assumption like P $\neq$ NP, there does not exist any polynomial-time δ-approximation algorithm for U for any $\delta \in \mathbb{R}^+$.

{The set cover problem belongs to this class.}

NPO(V): Contains every $U \in$ NPO such that if there exists a polynomial-time $f(n)$-approximation algorithm for U, then (under some reasonable assumption like P $\neq$ NP) $f(n)$ is not bounded by any polylogarithmic function.
{TSP and the maximum clique problem are well-known members of this class.}

Definition (NPO: NP Optimization)

$$\Pi = (L, sol, cost, goal)$$

$$\begin{aligned}
L &: l \in L \text{ is an instance} \\
&\quad \textit{decidable in poly. time} \\
sol &: x \in sol(l) \text{ is a feasible solution of } l \\
&\quad \textit{verifiable in poly. time} \\
cost &: cost(x) \text{ is the cost of } x \\
&\quad \textit{computable in poly. time} \\
goal &: goal \in \{\min, \max\}
\end{aligned}$$

$f(n)$-APX: $f(n)$-approximation

Exp-APX: $f(n) = O(2^{n^c})$

Poly-APX: $f(n) = O(n^c)$

Log-APX: $f(n) = O(\log n)$

APX: $f(n) = c \; (c > 1)$

PTAS: Poly. time approximation scheme

- $\forall \epsilon > 0 : (1 + \epsilon)$-approximation
- P : Poly(n) $\quad O((1/\epsilon) \cdot n^2) \quad O(n^{2/\epsilon})$

FPTAS: Fully poly. time approximation scheme

- $\forall \epsilon > 0 : (1 + \epsilon)$-approximation
- FP : Poly$(n, 1/\epsilon)$ $\quad O((1/\epsilon)^2 \cdot n^3)$

(if $P \neq NP$)

$$
\begin{aligned}
\text{PO} &\subsetneq \text{FPTAS} \subsetneq \text{PTAS} \\
&\subsetneq \text{APX} \subsetneq \text{Log-APX} \subsetneq \text{Poly-APX} \subsetneq \text{Exp-APX} \\
&\subsetneq \text{NPO}
\end{aligned}
$$

- Knapsack $\in$ FPTAS $\setminus$ PO
- Makespan $\in$ PTAS $\setminus$ FPTAS (TODAY)
- Vertex Cover $\in$ APX $\setminus$ PTAS
- Set Cover $\in$ Log-APX $\setminus$ APX (CLRS 35.3)
- Clique $\in$ Poly-APX $\setminus$ Log-APX
- TSP $\in$ Exp-APX $\setminus$ Poly-APX

Makespan Scheduling Problem (MS)

- n jobs: $J_1, \dots, J_n$
- processing time: $p_1, \dots, p_n$
- $m \ge 2$ machines: $M_1, \dots, M_m$
- goal: minimize the makespan

$$r = \frac{T}{T^*} \le \square$$

$$T \le \triangle$$

$$T^* \ge \triangledown$$

$$T \le \triangle$$

$$T^* \ge \bigtriangledown$$

$$T^* \ge \frac{1}{m} \sum_j p_j$$

J_i : the last job to complete

$$T^* \ge \max_j p_j$$

$$\begin{aligned} T &= s_i + p_i \\ &\le ?+? \end{aligned}$$

LS (List-Scheduling) Algorithm (JH 4.2.1.4)

- Online scheduling
- Assign job to the least heavily loaded

$$ms_i \leq \sum_j p_j$$

$$s_i \leq \frac{1}{m} \sum_j p_j \leq T^*$$

$$\begin{aligned} T &= s_i + p_i \\ &\leq T^* + T^* \\ &= 2T^* \end{aligned}$$

$$ms_i \leq \sum_{j \neq i} p_j$$

$$s_i \leq \frac{1}{m} \sum_j p_j - \frac{1}{m} p_i$$

$$\begin{aligned} T &= s_i + p_i \\ &\leq \frac{1}{m} \sum_j p_j + (1 - \frac{1}{m}) p_i \\ &\leq T^* + (1 - \frac{1}{m}) T^* \\ &= (2 - \frac{1}{m}) T^* \end{aligned}$$

$$(2 - \frac{1}{m}) \text{ is tight}$$

$$\frac{T}{T^*} = 2 - \frac{1}{m} = \frac{2m-1}{m}$$

$$n \text{ jobs } = \Big\{ \underbrace{p_i = 1}_{\#=m(m-1)}, \underbrace{p_i = m}_{\#=1} \Big\}$$

Longest Processing Time (LPT) Rule (JH 4.2.1.5)

- Sorting non-increasingly
- Applying LS

$$T = s_i + p_i$$

$$s_i \leq T^*$$

$$s_i \leq \frac{1}{m} \sum_j p_j - \frac{1}{m} p_i \qquad\qquad p_i \leq T^*$$

$$\left| M_i \right| \geq 2 \implies p_i \leq \frac{1}{2} T^* \qquad (J_i \text{ is on } M_i)$$

$$T = s_i + p_i \leq \frac{1}{m} \sum_j p_j + (1 - \frac{1}{m}) p_i \leq (\frac{3}{2} - \frac{1}{2m}) T^*$$

$$T = s_i + p_i \le \frac{1}{m}\sum_j p_j + (1 - \frac{1}{m})p_i \le \frac{4}{3} - \frac{1}{3m}$$

$$p_1 \ge \cdots \ge p_i \ge \cdots \ge p_n$$

CASE $p_i \le \frac{1}{3}T^*$:

$$T \le (\frac{4}{3} - \frac{1}{3m})T^*$$

CASE $p_i > \frac{1}{3}T^*$:

Consider $p_1 \ge \cdots \ge p_i > \frac{1}{3}T^*$

Upper bound for $\frac{T}{T^*}$ (T unchanged; T^* not smaller)

$$T = T^*$$

$$\forall i : \left|M_i\right| \le 2$$

$$J_1, J_2, \cdots, J_h, \quad J_{h+1}, J_{h+2}, \cdots, J_{n-1}, J_n$$

$$n = 2m - h$$

$(\frac{4}{3} - \frac{1}{3m})$ is tight

$$\frac{4}{3} - \frac{1}{3m} = \frac{4m-1}{3m}$$

$$n = 2m + 1$$

$$J = \{2m-1, 2m-1, \dots, m+1, m+1, m, m, m\}$$

Definition (3-Partition)

Instance:

$$A \subseteq \mathbb{Z}^+, \quad |A| = 3m$$

$$B \in \mathbb{Z}^+$$

$$\forall a \in A : B/4 < a < B/2$$

Question: Can A be partitioned into m disjoint sets $S_1, \ldots, S_m$:

$$\forall 1 \le i \le m : \sum_{a \in S_i} a = B$$

$$A = \{1, 2, 2, 3, 3, 4, 6, 7, 8\}, \quad m = 3, \quad B = 12$$

$$\Big\{1, 3, 8; \quad 2, 4, 6; \quad 2, 3, 7\Big\}$$

3-Partition $\leq_p$ MS

$$\text{MS} : (J, m, t)$$

$$t = B$$

MS is strongly NP-complete

MS with $(\max_j p_j) \leq q(n)$ is still NP-complete

Theorem (MS $\in$ PTAS $\setminus$ FPTAS)

No FPTAS for MS.

$$\exists \text{FPTAS for MS} \implies \text{MS} \in \text{P}$$

$$A_\epsilon : \epsilon = \frac{1}{\lceil 2nq(n) \rceil}$$

$$\text{Time: } \text{Poly}(\frac{1}{\epsilon}, n) = \text{Poly}(\lceil 2nq(n) \rceil, n) = \text{Poly}(n)$$

$$\begin{aligned} T &\leq (1+\epsilon)T^* = T^* + \epsilon \cdot T^* \\ &\leq T^* + \frac{1}{\lceil 2nq(n) \rceil} \cdot nq(n) \\ &\leq T^* + \frac{1}{2} \end{aligned}$$

$$T = T^*$$

TSP: worst-case complexity *vs.* inapproximability according to instances

- TSP $\in$ Exp-APX $\setminus$ Poly-APX
- Δ-TSP $\in$ APX
- Euclidean TSP $\in$ PTAS

Reference

- "Stability of Approximation Algorithms for Hard Optimization Problems" by Juraj Hromkovič, 1999.

Distance function (JH 4.2.3.3)

$$\text{dist}_k(G, c) =$$
$$\max\left\{0, \max\left\{\frac{c(\{u,v\})}{\sum_{i=1}^{m} c(\{p_i, p_{i+1}\})} - 1 \middle|\, |u = p_1 \rightsquigarrow v = p_m| \leq k\right\}\right\}$$

~~enumerate~~: $k = n^{\frac{1}{3}}$

shortest paths of length $\leq k$ (Bellman-Ford)

h_{index} (JH 4.2.3.4)

h_{index}: using canonical order

$$|\text{Ball}_{r,h_{\text{index}}}(L_I)| < \infty$$

$$\delta_{r,\epsilon} = \max\{R_A(x) : x \in \text{Ball}_{r,h_{\text{index}}}(L_I)\}$$

h (JH 4.2.3.5)

- h: infinite jumps
- δ-approx. algorithm A for U is stable according to h

$$\text{TSP } U : (G, 1)$$

$$\text{Multi-TSP } \overline{U} : (G, k)$$

$$h(G, k) = k - 1$$

$$A \text{ is } \delta\text{-approx. for } (G, 1) \implies A \text{ is } r\delta\text{-approx. for } (G, r \in \mathbb{N})$$