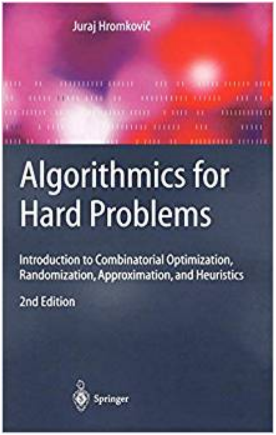
4-12 Approximation Algorithms

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 $f : \mathbb{N} \to \mathbb{R}^+$, 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)

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

Definition (NPO: NP Optimization)

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

$$L: \; l \in L \text{ is an instance}$$

$$sol: \; x \in sol(l) \text{ is a feasible solution of } l$$

$$cost: \; cost(x) \text{ is the cost of } x$$

$$goal: \; goal \in \{\min, \max\}$$

Definition (NPO: NP Optimization)

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

L : $l \in L$ is an instance
decidable in poly. time

sol : $x \in sol(l)$ is a feasible solution of l
verifiable in poly. time

$cost$: $cost(x)$ is the cost of x
computable in poly. time

$goal$: $goal \in \{\min, \max\}$

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

$$
\begin{aligned}
\text{Exp-APX:} \quad & f(n) = O(2^{n^c}) \\
\text{Poly-APX:} \quad & f(n) = O(n^c) \\
\text{Log-APX:} \quad & f(n) = O(\log n) \\
\text{APX:} \quad & f(n) = c \ (c > 1)
\end{aligned}
$$

$f(n)$-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 $\mathrm{P} \neq \mathrm{NP}$)

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

(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: $J_1, \ldots, J_n$
- processing time: $p_1, \ldots, p_n$
- $m \geq 2$ machines: $M_1, \ldots, M_m$
- goal: minimize the makespan

Makespan Scheduling Problem (MS)

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

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

Makespan Scheduling Problem (MS)

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

$$r = \frac{T}{T^*} \leq \Box$$

$$T \leq \triangle$$

$$T^* \geq \nabla$$

$T \leq \triangle$

$T^* \geq \bigtriangledown$

$$T \leq \triangle$$

$$T^* \geq \triangledown$$

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

$$T^* \geq \max_j p_j$$

$$T^* \geq \triangledown$$

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

$$T^* \geq \max_j p_j$$

$$T \leq \triangle$$

J_i : the last job to complete

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

LS (List-Scheduling) Algorithm (JH 4.2.1.4)

- Online scheduling
- Assign job to the least heavily loaded

LS (List-Scheduling) Algorithm (JH 4.2.1.4)

- Online scheduling
- Assign job to the least heavily loaded

$$m s_i \le \sum_j p_j$$

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

LS (List-Scheduling) Algorithm (JH 4.2.1.4)

- Online scheduling
- Assign job to the least heavily loaded

$$ms_i \leq \sum_j p_j$$

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

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

LS (List-Scheduling) Algorithm (JH 4.2.1.4)

- Online scheduling
- Assign job to the least heavily loaded

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

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

$$ms_i \le \sum_j p_j$$

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

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

LS (List-Scheduling) Algorithm (JH 4.2.1.4)

- Online scheduling
- Assign job to the least heavily loaded

$$ms_i \leq \sum_j p_j$$

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

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

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

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

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

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

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

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

$(2 - \frac{1}{m})$ 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
- Assign job to the least heavily loaded

Longest Processing Time (LPT) Rule (JH 4.2.1.5)

- Sorting non-increasingly
- Assign job to the least heavily loaded

$$T = s_i + p_i$$

$$s_i \le T^*$$

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

$$p_i \le T^*$$

Longest Processing Time (LPT) Rule (JH 4.2.1.5)

- Sorting non-increasingly
- Assign job to the least heavily loaded

$$T = s_i + p_i$$

$$s_i \leq T^*$$

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

$$p_i \leq T^*$$

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

Longest Processing Time (LPT) Rule (JH 4.2.1.5)

- Sorting non-increasingly
- Assign job to the least heavily loaded

$$T = s_i + p_i$$

$$s_i \le T^*$$

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

We show that $T = T^*$.

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

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

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

We show that $T = T^*$.

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

$$n = 2m - h$$

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

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

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

We show that $T = T^*$.

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

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

$$n = 2m - h$$

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

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

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

We show that $T = T^*$.

$\forall i : \left| M_i \right| \leq 2$ $\quad$ $J_1, J_2, \cdots, J_h, \quad J_{h+1}, J_{h+2}, \cdots, J_{n-1}, J_n$

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

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

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

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

We show that $T = T^*$.

$\forall i : \left| M_i \right| \le 2$ $\quad J_1, J_2, \cdots, J_h, \quad J_{h+1}, J_{h+2}, \cdots, J_{n-1}, J_n$

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

By Exchange Argument.

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

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

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

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

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

$$n = 2m + 1$$

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

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 $S_1, \ldots, S_m$:

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

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 $S_1, \ldots, S_m$:

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

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

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

3-Partition $\leq_p$ MS

3-Partition $\leq_p$ MS

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

3-Partition $\leq_p$ MS

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

$$\text{3-Partition} : (A, B)$$

3-Partition $\leq_p$ MS

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

$$\text{3-Partition} : (A, B)$$

$$m = m, \quad t = B$$

$$3\text{-Partition} \le_p \text{MS}$$

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

$$3\text{-Partition} : (A, B)$$

$$m = m, \quad t = B$$

MS is strongly NP-complete

3-Partition $\leq_p$ MS

MS : (J, m, t)

3-Partition : (A, B)

$m = m, \quad t = B$

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.

Theorem ($\mathrm{MS} \in \mathrm{PTAS} \setminus \mathrm{FPTAS}$)

No FPTAS for MS.

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

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

No FPTAS for MS.

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

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

Theorem ($\mathrm{MS} \in \mathrm{PTAS} \setminus \mathrm{FPTAS}$)

No FPTAS for MS.

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

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

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

Theorem ($\text{MS} \in \text{PTAS} \setminus \text{FPTAS}$)

No FPTAS for MS.

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

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

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

$$T \leq (1+\epsilon)T^* = T^* + \epsilon \cdot T^*$$

Theorem ($\text{MS} \in \text{PTAS} \setminus \text{FPTAS}$)

No FPTAS for MS.

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

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

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

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

Theorem ($\text{MS} \in \text{PTAS} \setminus \text{FPTAS}$)

No FPTAS for MS.

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

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

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

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

Theorem ($\text{MS} \in \text{PTAS} \setminus \text{FPTAS}$)

No FPTAS for MS.

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

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

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

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

$$T = T^*$$

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

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

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

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

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