

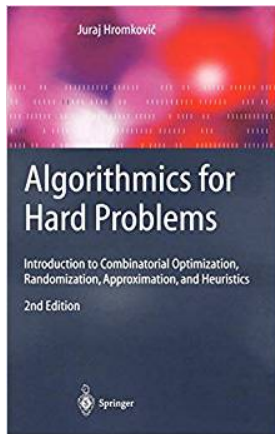
4-12 Approximation Algorithms

Hengfeng Wei

hfwei@nju.edu.cn

June 03, 2019





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

sol : $x \in sol(l)$ is a feasible solution of l

$cost$: $cost(x)$ 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

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

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

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

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

$f(n)$ -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
- ▶ $P : \text{Poly}(n) \quad O((1/\epsilon) \cdot n^2) \quad O(n^{2/\epsilon})$

FPTAS: Fully poly. time approximation scheme

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

(if $P \neq NP$)

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

(if $P \neq NP$)

$PO \subsetneq FPTAS \subsetneq PTAS$
 $\subsetneq APX \subsetneq \text{Log-APX} \subsetneq \text{Poly-APX} \subsetneq \text{Exp-APX}$
 $\subsetneq NPO$

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

Makespan Scheduling Problem (MS)

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

Makespan Scheduling Problem (MS)

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

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

Makespan Scheduling Problem (MS)

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

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

$$T \leq \Delta$$

$$T^* \geq \nabla$$

$$T^* \geq \nabla$$

$$T \leq \Delta$$

$$T^* \geq \nabla$$

$$T \leq \Delta$$

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

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

$$T^* \geq \nabla$$

$$T \leq \Delta$$

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

J_i : the last job to complete

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

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

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

$$ms_i \leq \sum_j p_j$$

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

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

$$ms_i \leq \sum_j p_j$$

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

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

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

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

LS (List-Scheduling) Algorithm (JH 4.2.1.4)

- ▶ Online scheduling
- ▶ Assign job to the least heavily loaded

$$ms_i \leq \sum_j p_j$$

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

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

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

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

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

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

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

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

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

$$T = s_i + p_i$$

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

$$p_i \leq T^*$$

Longest Processing Time (LPT) Rule (JH 4.2.1.5)

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

$$T = s_i + p_i$$

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

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

Longest Processing Time (LPT) Rule (JH 4.2.1.5)

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

$$T = s_i + p_i$$

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

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

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

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

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

$$p_1 \geq \cdots \geq p_i \geq \cdots \geq p_n$$

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

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

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

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

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

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

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

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

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

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

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

We show that $T = T^*$.

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

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

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

We show that $T = T^*$.

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

$$n = 2m - h$$

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

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

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

We show that $T = T^*$.

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

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

$$n = 2m - h$$

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

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

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

We show that $T = T^*$.

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

$$n = 2m - h$$

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

$$J_1, J_2, \dots, J_h, (J_{h+1}, J_n), (J_{h+2}, J_{n-1}), \dots$$

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

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

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

We show that $T = T^*$.

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

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

$$n = 2m - h$$

$$J_1, J_2, \dots, J_h, (J_{h+1}, J_n), (J_{h+2}, J_{n-1}), \dots$$

By Exchange Argument.

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

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

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

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

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

$$n = 2m + 1$$

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

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, \dots, S_m :

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

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, \dots, S_m :

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

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

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

3-Partition \leq_p MS

3-Partition \leq_p MS

MS : (J, m, t)

3-Partition \leq_p MS

MS : (J, m, t)

3-Partition : (A, B)

3-Partition \leq_p MS

MS : (J, m, t)

3-Partition : (A, B)

$m = m, \quad t = B$

3-Partition \leq_p MS

MS : (J, m, t)

3-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 ($MS \in PTAS \setminus FPTAS$)

No FPTAS for MS.

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

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

No FPTAS for MS.

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

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

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

No FPTAS for MS.

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

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

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

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

No FPTAS for MS.

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

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

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 ($MS \in PTAS \setminus FPTAS$)

No FPTAS for MS.

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

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

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

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

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

No FPTAS for MS.

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

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

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

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

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

No FPTAS for MS.

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

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

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

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

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

- ▶ $\text{TSP} \in \text{Exp-APX} \setminus \text{Poly-APX}$
- ▶ $\Delta\text{-TSP} \in \text{APX}$
- ▶ Euclidean TSP $\in \text{PTAS}$

Reference

- ▶ “Stability of Approximation Algorithms for Hard Optimization Problems” by Juraj Hromkovič, 1999.

Distance function (JH 4.2.3.3)

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

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

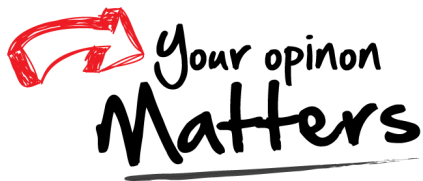
TSP $U : (G, 1)$

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

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

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





Office 302

Mailbox: H016

hfwei@nju.edu.cn