Approximation Algorithms

Hengfeng Wei

hfwei@nju.edu.cn

May 14 $\sim$ May 18, 2017

Approximation Algorithms

Approximation Classes

NPO: NP optimization problems

APX: constant factor ϵ-approximation ($\epsilon > 1$)

$f(n)$-APX: Exp-APX, Poly-APX, Log-APX

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

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

PO: polynomial time solvable optimization problems

Approximation Classes

$$\begin{aligned} \text{PO} &\subsetneq \text{FPTAS} \subsetneq \text{PTAS} \subsetneq \text{APX} \\ &\subsetneq \text{Log-APX} \subsetneq \text{Poly-APX} \subsetneq \text{Exp-APX} \subsetneq \text{NPO} \\ &\qquad (\text{if P} \neq \text{NP}) \end{aligned}$$

1. Knapsack $\in$ FPTAS $\setminus$ PO
2. Makespan $\in$ PTAS $\setminus$ FPTAS (TODAY)
3. Vertex Cover $\in$ APX $\setminus$ PTAS
4. Set Cover $\in$ Log-APX $\setminus$ APX (CLRS 35.3)
5. Clique $\in$ Poly-APX $\setminus$ Log-APX
6. TSP $\in$ Exp-APX $\setminus$ Poly-APX

Reference

- "A Survey on the Structure of Approximation Classes" by Bruno Escoffier, Vangelis Th. Paschos, 2010.

Stability of approximation

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.

Stability of approximation

Distance function (JH 4.2.3.3)

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

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

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

Stability of approximation

h_{index} (JH 4.2.3.4)

h_{index}: using canonical order

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

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

Stability of approximation

h (JH 4.2.3.5)

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

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

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

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

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

Approximation Algorithms

Makespan scheduling problem

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

MS is NP-complete

Definition (Partition)

Instance:

$$\forall a \in A : s(a) \in \mathbb{Z}^+$$

Question: Is there a subset $A' \subseteq A$:

$$\sum_{a \in A'} s(a) = \sum_{a \in A \setminus A'} s(a)$$

$$A = \{5, 1, 3, 4, 8, 2, 7\} \implies A' = \{5, 3, 7\}, A \setminus A' = \{1, 2, 4, 8\}$$

MS is strongly NP-complete

Definition (3-Partition)

Instance:

$$|A| = 3m, B \in \mathbb{Z}^+$$
$$\forall a \in A : s(a) \in \mathbb{Z}^+, B/4 < s(a) < B/2$$

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

$$\forall 1 \leq i \leq m : |S_i| = 3, \sum_{a \in S_i} s(a) = B$$

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

List-Scheduling (LS) algorithm

List-Scheduling algorithm (JH 4.2.1.4)

- online
- assign job to the least heavily loaded

$$J = 2, 3, 4, 6, 2, 2 \qquad m = 3$$

LS is 2-approx.

$$T \text{ vs. } T^* : \frac{T}{T^*}$$

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

$$T^* \geq \max_j t_j$$

1. M_i: with the maximum load
2. J_i: the last job on M_i

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

$$T = c_i = s_i + p_i \leq T^* + T^* = 2T^*$$

2-approx. is (almost) tight

$$n = \underbrace{m(m-1)}_{p_i=1} + \underbrace{1}_{p_i=m}$$

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

LS is $(2 - \frac{1}{m})$-approx.

$$
\begin{aligned}
ms_i \le \sum_{j \ne i} p_j &= \frac{1}{m}(\sum_j p_j - p_i) \\
&= \frac{1}{m}\sum_j p_j - \frac{1}{m}p_i \\
&\le T^* - \frac{1}{m}p_i
\end{aligned}
$$

$$
\begin{aligned}
T = c_i &= s_i + p_i \\
&\le T^* + (1 - \frac{1}{m})p_i
\end{aligned}
$$

Sorting-Scheduling algorithm

Sorting-Scheduling algorithm (JH 4.2.1.5)

Longest Processing Time (LPT) rule:

- sorting non-increasingly
- applying LS

1. M_i: with the maximum load
2. J_i: the last job on M_i

$$|M_i| = 1 \implies T = T^*$$

$$\begin{aligned} |M_i| \geq 2 &\implies p_i \leq \frac{1}{2} T^* \\ &\implies T = s_i + p_i \leq (\frac{3}{2} - \frac{1}{2m}) T^* \end{aligned}$$

LPT rule is $(\frac{4}{3} - \frac{1}{3m})$-approx.

$$p_1 \geq p_2 \geq \cdots \geq p_n$$

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

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

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

$$p_i \equiv p_n (\text{w.l.o.g: } T \text{ unchanged; } T^* \text{ not smaller})$$
$$\implies p_1 \geq p_2 \geq \cdots \geq p_n > \frac{1}{3}T^* \implies |M_i| \leq 2$$
$$\implies n \leq 2m \implies n = 2m - h \xrightarrow[\text{argument}]{\text{exchange}} T = T^*$$

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

$$n = 2m + 1$$

$$\begin{aligned} \text{LPT: } & J_1 = x, J_{2m} = y, J_{2m+1} = z \\ \text{OPT: } & J_{2m-1} = J_{2m} = J_{2m+1} = y \\ \textit{vs. } & \frac{x+2y}{3y} = \frac{4}{3} - \frac{1}{3m} \end{aligned}$$

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

Reference

- "Bounds on Multiprocessing Timing Anomalies" by R.L.Graham, 1969
- "Approximation Algorithms for NP-Hard Problems" edited by Dorit Hochbaum, 1996 (Theorem 1.5)

PTAS for MS

1. M_i: with the maximum load
2. J_i: the last job on M_i

$$T = s_i + p_i \leq T^* + p_i$$

1. $J = J_L \triangleq \{\text{long jobs}\} \uplus J_S \triangleq \{\text{short jobs}\}$
2. S_L: the optimal schedule for J_L
3. S: apply List-Scheduling to S_L and J_S

Reference

- "The Design of Approximation Algorithms" by David P. Williamson and David Shmoys, 2011 (Section 3.2).

PTAS for MS

1. Split J:

$$J_i \in J_S \iff p_i \le \epsilon \cdot \frac{1}{m} \sum_j p_j$$

$$\implies |J_L| < \frac{1}{\epsilon} \cdot m$$

2. Time for S_L (m being a constant!):

$$m^{\frac{1}{\epsilon} \cdot m} \cdot O(n)$$

3. Approx. ratio ($p_i \in J_S$ case):

$$T = s_i + p_i \le \frac{1}{m} \sum_j p_j + \epsilon \cdot \frac{1}{m} \sum_j p_j$$

$$= (1 + \epsilon) \frac{1}{m} \sum_j p_j$$

$$\le (1 + \epsilon) T^*$$

No FPTAS for MS

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

No FPTAS for MS.

MS is strongly NP-complete $\implies$ MS with $\max_j p_j \leq q(n)$ is NP-complete.

Reference

- "The Design of Approximation Algorithms" by David P. Williamson and David Shmoys, 2011 (Section 3.2).

No FPTAS for MS

Theorem

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

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

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

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


THANK YOU