

- 教材讨论
  - JH第4章第1节、第2节第1、2小节

# 问题1：近似算法的基本概念

- 什么样的算法可以称作近似算法？

We start with the fundamental definition of approximation algorithms. Informally and roughly, an approximation algorithm for an optimization problem is an algorithm that provides a feasible solution whose quality does not differ too much from the quality of an optimal solution.

- 你理解does not differ too much了吗？

# 问题1：近似算法的基本概念 (续)

- 你理解这些概念了吗？

- relative error

$$\varepsilon_A(x) = \frac{|cost(A(x)) - Opt_U(x)|}{Opt_U(x)}.$$

$$\varepsilon_A(n) = \max \{ \varepsilon_A(x) \mid x \in L_I \cap (\Sigma_I)^n \}.$$

- approximation ratio

$$R_A(x) = \max \left\{ \frac{cost(A(x))}{Opt_U(x)}, \frac{Opt_U(x)}{cost(A(x))} \right\}.$$

$$R_A(n) = \max \{ R_A(x) \mid x \in L_I \cap (\Sigma_I)^n \}.$$

- $\delta$ -approximation algorithm

$$R_A(x) \leq \delta \text{ for every } x \in L_I.$$

- $f(n)$ -approximation algorithm

$$R_A(n) \leq f(n) \text{ for every } n \in \mathbb{N}.$$

# 问题1: 近似算法的基本概念 (续)

- 这个算法的基本过程是什么?

**Algorithm 4.2.1.3** (GMS (GREEDY MAKESPAN SCHEDULE)).

Input:  $I = (p_1, \dots, p_n, m)$ ,  $n, m, p_1, \dots, p_n$  positive integers and  $m \geq 2$ .

Step 1: Sort  $p_1, \dots, p_n$ .

To simplify the notation we assume  $p_1 \geq p_2 \geq \dots \geq p_n$  in the rest of the algorithm.

Step 2: for  $i = 1$  to  $m$  do

**begin**  $T_i := \{i\}$ ;

$Time(T_i) := p_i$

**end**

{In the initialization step the  $m$  largest jobs are distributed to the  $m$  machines. At the end,  $T_i$  should contain the indices of all jobs assigned to the  $i$ th machine for  $i = 1, \dots, m$ .}

Step 3: for  $i = m + 1$  to  $n$  do

**begin** compute an  $l$  such that

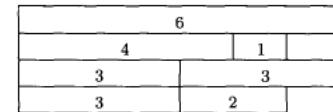
$Time(T_l) := \min\{Time(T_j) \mid 1 \leq j \leq m\}$ ;

$T_l := T_l \cup \{i\}$ ;

$Time(T_l) := Time(T_l) + p_i$

**end**

Output:  $(T_1, T_2, \dots, T_m)$ .



# 问题1: 近似算法的基本概念 (续)

- 你能逐步推导出它的approximation ratio吗?

$$Opt_{MS}(I) \geq p_1 \geq p_2 \geq \dots \geq p_n. \quad (4.1)$$

$$Opt_{MS}(I) \geq \frac{\sum_{i=1}^n p_i}{m} \quad (4.2)$$

$$p_k \leq \frac{\sum_{i=1}^k p_i}{k} \quad (4.3)$$

(1) Let  $n \leq m$ .

Since  $Opt_{MS}(I) \geq p_1$  (4.1) and  $cost(\{1\}, \{2\}, \dots, \{n\}, \emptyset, \dots, \emptyset) = p_1$ , GMS has found an optimal solution and so the approximation ratio is 1.

(2) Let  $n > m$ .

Let  $T_l$  be such that  $cost(T_l) = \sum_{r \in T_l} p_r = cost(GMS(I))$ , and let  $k$  be the largest index in  $T_l$ . If  $k \leq m$ , then  $|T_l| = 1$  and so  $Opt_{MS}(I) = p_1 = p_k$  and GMS( $I$ ) is an optimal solution.

Now, assume  $m < k$ . Following Figure 4.2 we see that

$$Opt_{MS}(I) \geq cost(GMS(I)) - p_k \quad (4.4)$$

because of  $\sum_{i=1}^{k-1} p_i \geq m \cdot [cost(GMS(I)) - p_k]$  and (4.2).

$$cost(GMS(I)) - Opt_{MS}(I) \leq p_k \leq \frac{\left(\sum_{i=1}^k p_i\right)}{k} \quad (4.5)$$

$$\frac{cost(GMS(I)) - Opt_{MS}(I)}{Opt_{MS}(I)} \leq \frac{\left(\sum_{i=1}^k p_i\right)/k}{\left(\sum_{i=1}^n p_i\right)/m} \leq \frac{m}{k} < 1.$$

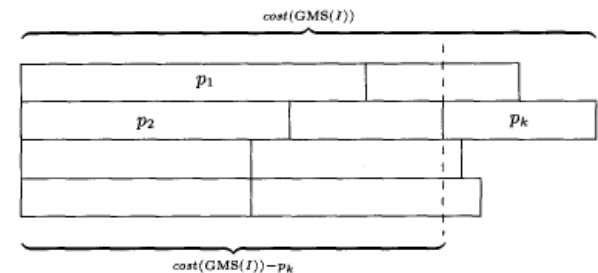


Fig. 4.2.

# 问题1：近似算法的基本概念 (续)

- 你理解PTAS和FPTAS了吗？它们的区别是什么？

**Definition 4.2.1.6.** Let  $U = (\Sigma_I, \Sigma_O, L, L_I, M, cost, goal)$  be an optimization problem. An algorithm  $A$  is called a **polynomial-time approximation scheme (PTAS)** for  $U$ , if, for every input pair  $(x, \varepsilon) \in L_I \times \mathbb{R}^+$ ,  $A$  computes a feasible solution  $A(x)$  with a relative error at most  $\varepsilon$ , and  $Time_A(x, \varepsilon^{-1})$  can be bounded by a function<sup>3</sup> that is polynomial in  $|x|$ . If  $Time_A(x, \varepsilon^{-1})$  can be bounded by a function that is polynomial in both  $|x|$  and  $\varepsilon^{-1}$ , then we say that  $A$  is a **fully polynomial-time approximation scheme (FPTAS)** for  $U$ .

- 你理解这两句话了吗？
  - The advantage of PTASs is that the user has the choice of  $\varepsilon$  in this tradeoff of the quality of the output and the amount of computer work.
  - Probably a FPTAS is the best that one can have for a NP-hard optimization problem.

# 问题2: stability

- 你觉得讨论stability的意义是什么？
- 你理解这些概念了吗？

**Definition 4.2.3.1.** Let  $U = (\Sigma_I, \Sigma_O, L, L_I, \mathcal{M}, \text{cost}, \text{goal})$  and  $\bar{U} = (\Sigma_I, \Sigma_O, L, L, \mathcal{M}, \text{cost}, \text{goal})$  be two optimization problems with  $L_I \subset L$ . A **distance function for  $\bar{U}$  according to  $L_I$**  is any function  $h_L : L \rightarrow \mathbb{R}^{\geq 0}$  satisfying the properties

- (i)  $h_L(x) = 0$  for every  $x \in L_I$ , and
- (ii)  $h$  is polynomial-time computable.

Let  $h$  be a distance function for  $\bar{U}$  according to  $L_I$ . We define, for any  $r \in \mathbb{R}^+$ ,

$$\mathbf{Ball}_{r,h}(L_I) = \{w \in L \mid h(w) \leq r\}.$$
<sup>6</sup>

Let  $A$  be a consistent algorithm for  $\bar{U}$ , and let  $A$  be an  $\epsilon$ -approximation algorithm for  $U$  for some  $\epsilon \in \mathbb{R}^{>1}$ . Let  $p$  be a positive real. We say that  $A$  is **p-stable according to  $h$**  if, for every real  $0 < r \leq p$ , there exists a  $\delta_{r,\epsilon} \in \mathbb{R}^{>1}$  such that  $A$  is a  $\delta_{r,\epsilon}$ -approximation algorithm for  $U_r = (\Sigma_I, \Sigma_O, L, \mathbf{Ball}_{r,h}(L_I), \mathcal{M}, \text{cost}, \text{goal})$ .

$A$  is **stable according to  $h$**  if  $A$  is  $p$ -stable according to  $h$  for every  $p \in \mathbb{R}^+$ . We say that  $A$  is **unstable according to  $h$**  if  $A$  is not  $p$ -stable for any  $p \in \mathbb{R}^+$ .

For every positive integer  $r$ , and every function  $f_r : \mathbb{N} \rightarrow \mathbb{R}^{>1}$  we say that  $A$  is **( $r, f_r(n)$ )-quasistable according to  $h$**  if  $A$  is an  $f_r(n)$ -approximation algorithm for  $U_r = (\Sigma_I, \Sigma_O, L, \mathbf{Ball}_{r,h}(L_I), \mathcal{M}, \text{cost}, \text{goal})$ .

# 问题2: stability (续)

- 你理解TSP中的这些distance了吗?

$$\text{dist}(G, c) = \max \left\{ 0, \max \left\{ \frac{c(\{u, v\})}{c(\{u, p\}) + c(\{p, v\})} - 1 \mid u, v, p \in V(G), \right. \right. \\ \left. \left. u \neq v, u \neq p, v \neq p \right\} \right\},$$

$$\text{dist}_k(G, c) = \max \left\{ 0, \max \left\{ \frac{c(\{u, v\})}{\sum_{i=1}^m c(\{p_i, p_{i+1}\})} - 1 \mid u, v \in V(G) \text{ and} \right. \right. \\ \left. \left. u = p_1, p_2, \dots, p_m = v \text{ is a simple path between } u \text{ and } v \right. \right. \\ \left. \left. \text{of length at most } k \text{ (i.e., } m + 1 \leq k) \right\} \right\}$$

$$\text{distance}(G, c) = \max\{\text{dist}_k(G, c) \mid 2 \leq k \leq |V(G)| - 1\}.$$

- 例如,  $\text{Ball}_{r, \text{dist}}(L_\Delta)$  中都是些什么?

$$c(\{u, v\}) \leq (1 + r)(c(\{u, p\}) + c(\{p, v\}))$$

- 你能基于其它难问题, 举出一些distance的例子吗?



# 问题2: stability (续)

- 你理解PTAS的stability了吗?

Note that applying the concept of stability to PTASs one can get two different outcomes. Let us consider a PTAS  $A$  as a collection of polynomial-time  $(1 + \varepsilon)$ -approximation algorithms  $A_\varepsilon$  for every  $\varepsilon \in \mathbb{R}^+$ . If  $A_\varepsilon$  is stable according to a distance measure  $h$  for every  $\varepsilon > 0$ , then we can obtain either

- (i) a PTAS for  $U_r = (\Sigma_I, \Sigma_O, L, Ball_{r,h}(L_I), \mathcal{M}, cost, goal)$  for every  $r \in \mathbb{R}^+$  (this happens, for instance, if  $\delta_{r,\varepsilon} = 1 + \varepsilon \cdot f(r)$ , where  $f$  is an arbitrary function), or
- (ii) a  $\delta_{r,\varepsilon}$ -approximation algorithm for  $U_r$  for every  $r \in \mathbb{R}^+$ , but no PTAS for  $U_r$  for any  $r \in \mathbb{R}^+$  (this happens, for instance, if  $\delta_{r,\varepsilon} = 1 + r + \varepsilon$ ).