- 教材讨论
  - JH第4章第3节第6小节

本节的来源

- Dorit S. Hochbaum, David B. Shmoys. Using Dual Approximation Algorithms for Scheduling Problems: Theoretical and Practical Results. *Journal of the ACM*, 34(1):144—162, 1987.

问题1：dual approximation algorithms

- 这节最终目标是求解MS，你能解释整体思路吗？

(i) to design a dual polynomial-time approximation scheme[34] (dual PTAS) for the bin-packing problem (BIN-P), and

Step 1: Use the method of dynamic programming to design a polynomial-time algorithm DPB-P for input instances of BIN-P that contain a constant number of different values of r_is (i.e., the input involves a lot of multiple occurrences of some values r_i).

Step 2: Apply DPB-P (in a similar way as for the knapsack problem in Section 4.3.4) to obtain an h-dual PTAS for the input instances of BIN-P that do not contain "very small" r_is.

Step 3: Use the above h-dual PTAS to design an h-dual PTAS for the general BIN-P.

(ii) to use the dual PTAS for the BIN-P to design a PTAS for the makespan scheduling problem (MS).

问题2：dual PTAS for BIN-P (1)

Step 1: Use the method of dynamic programming to design a polynomial-time algorithm DPB-P for input instances of BIN-P that contain a constant number of different values of r_is (i.e., the input involves a lot of multiple occurrences of some values r_i).

- 你能解释动态规划的递归式吗？

$$\text{Bin-P}(m_1, \ldots, m_s) =$$
$$1 + \min_{x_1,\ldots,x_s} \left\{ \text{Bin-P}(m_1 - x_1, \ldots, m_s - x_s) \,\middle|\, \sum_{i=1}^{s} x_i q_i \le 1 \right\}.$$

- 你能解释算法4.3.6.1及其时间复杂度吗？

Algorithm 4.3.6.1 (DPB-P$_s$).

Input: $q_1, \ldots, q_s$, $n_1, \ldots, n_s$, where $q_i \in (0,1]$ for $i = 1, \ldots, s$, and $n_1, \ldots, n_s$ are positive integers.

Step 1: BIN-P$(0, \ldots, 0) := 0$;
Bin-P$(h_1, \ldots, h_s) := 1$ for all $(h_1, \ldots, h_s) \in \{0, \ldots, n_1\} \times \cdots \times \{0, \ldots, n_s\}$ such that $\sum_{i=1}^{s} h_i q_i \le 1$ and $\sum_{i=1}^{s} h_i \ge 1$.

Step 2: Compute Bin-P$(m_1, \ldots, m_s)$ with the corresponding optimal solution $T(m_1, \ldots, m_s)$ by the recurrence (4.61) for all $(m_1, \ldots, m_s) \in \{0, \ldots, n_1\} \times \cdots \times \{0, \ldots, n_s\}$.

Output: BIN-P$(n_1, \ldots, n_s)$, $T(m_1, \ldots, m_s)$.

$$\boxed{n_1 \cdot n_2 \cdot \cdots \cdot n_s \le \left(\frac{\sum_{i=1}^{s} n_i}{s} \right)^s = \left(\frac{n}{s} \right)^s} \Rightarrow O((\tfrac{n}{s})^{2s})$$

问题2：dual PTAS for BIN-P (2)

Step 2: Apply DPB-P (in a similar way as for the knapsack problem in Section 4.3.4) to obtain an h-dual PTAS for the input instances of BIN-P that do not contain "very small" r_is.

- 你能解释算法4.3.6.2吗？
  它对输入做了怎样的处理？
- 它为什么是Bin-P$_\varepsilon$的h-dual ε-近似算法？

Algorithm 4.3.6.2 (BP-PTA$_\varepsilon$).

Input: $(q_1, q_2, \ldots, q_n)$, where $\varepsilon < q_1 \leq \cdots \leq q_n \leq 1$.

Step 1: Set $s := \lceil \log_2(1/\varepsilon)/\varepsilon \rceil$;
$l_1 := \varepsilon$, and
$l_j := l_{j-1} \cdot (1+\varepsilon)$ for $j = 2, 3, \ldots, s$;
$l_{s+1} = 1$.
{This corresponds to the partitioning of the interval $(\varepsilon, 1]$ into s subintervals $(l_1, l_2]$, $(l_2, l_3]$, ..., $(l_s, l_{s+1}]$.}

Step 2: **for** $i = 1$ **to** s **do**
  do begin $L_i := \{q_1, \ldots, q_n\} \cap (l_i, l_{i+1}]$;
    $n_i := |L_i|$
  end
{We consider that every value of L_i is rounded to the value l_i in what follows.}

Step 3: Apply DPB-P$_s$ on the input $(l_1, l_2, \ldots, l_s, n_1, n_2, \ldots, n_s)$.

To prove that BP-PTA$_\varepsilon$ is an h-dual ε-approximation algorithm for Bin-P$_\varepsilon$, we have to prove that, for every input $I = (q_1, q_2, \ldots, q_n)$, $\varepsilon < q_1 \leq \cdots \leq q_n \leq 1$, the following two facts hold:

(i) $r = cost(T_1, \ldots, T_r) = \text{Bin-P}(n_1, \ldots, n_s) \leq Opt_{\text{Bin-P}}(I)$, where $(T_1, \ldots, T_r)$ is the optimal solution for the input $Round(I) = (l_1, \ldots, l_s, n_1, \ldots, n_s)$ computed by BP-PTA$_\varepsilon$ [T_i is the set of the multiplicatives of the indices of the values packed in the ith bin], and

(ii) for every $j = 1, \ldots, r$, $\sum_{a \in T_j} q_a \leq 1 + \varepsilon$.

The fact (i) is obvious because $Round(I)$ can be considered as $(p_1, \ldots, p_n)$, where $p_i \leq q_i$ for every $i \in \{1, \ldots, n\}$.

Since DPB-P$_s(Round(I)) \leq Opt_{\text{Bin-P}}(I)$, we obtain

$$\text{Bin-P}(n_1, \ldots, n_s) = Opt_{\text{Bin-P}}(Round(I)) \leq Opt_{\text{Bin-P}}(I).$$

To prove (ii), consider an arbitrary set of indices $T \in \{T_1, T_2, \ldots, T_r\}$. Let $x_T = (x_1, \ldots, x_n)$ be the corresponding description of the set of indices assigned to this bin for *Round*(I). We can bound $\sum_{j \in T} q_j$ as follows:

$$\sum_{j \in T} q_j \leq \sum_{i=1}^{s} x_i l_{i+1} = \sum_{i=1}^{s} x_i l_i + \sum_{i=1}^{s} x_i (l_{i+1} - l_i) \leq 1 + \sum_{i=1}^{s} x_i (l_{i+1} - l_i). \quad (4.62)$$

Since $l_i > \varepsilon$ for every $i \in \{1, \ldots, s\}$, the number of pieces in a bin is at most $\left\lfloor \frac{1}{\varepsilon} \right\rfloor$, i.e.,

$$\sum_{i=1}^{s} x_i \leq \left\lfloor \frac{1}{\varepsilon} \right\rfloor . \quad (4.63)$$

Let, for $i = 1, \ldots, s$, a_i be the fraction of the bin T filled by values of size l_i. Obviously,

$$x_i \leq \frac{a_i}{l_i} \quad (4.64)$$

for every $i \in \{1, 2, \ldots, s\}$. Inserting (4.64) into (4.62) we obtain

$$\begin{aligned}
\sum_{j \in T} q_j &\underset{(4.62)}{\leq} 1 + \sum_{i=1}^{s} x_i (l_{i+1} - l_i) \\
&\underset{(4.64)}{\leq} 1 + \sum_{i=1}^{s} \frac{a_i}{l_i} (l_{i+1} - l_i) \\
&= 1 + \sum_{i=1}^{s} \left[a_i \cdot \frac{l_{i+1}}{l_i} - a_i \right] \\
&= 1 + \sum_{i=1}^{s} a_i \cdot \left(\frac{l_{i+1}}{l_i} - 1 \right) \\
&= 1 + \sum_{i=1}^{s} a_i \cdot \varepsilon = 1 + \varepsilon \cdot \sum_{i=1}^{s} a_i = 1 + \varepsilon
\end{aligned}$$

问题2： dual PTAS for BIN-P (3)

Step 3: Use the above h-dual PTAS to design an h-dual PTAS for the general BIN-P.

- 你能解释算法4.3.6.4吗？
  对输入做了怎样的处理？
- 它为什么是BIN-P的h-dual PTAS？

Algorithm 4.3.6.4 (Bin-PTAS).

Input: (I, ε), where $I = (q_1, q_2, \ldots, q_n)$, $0 \le q_1 \le q_2 \le \cdots \le q_n \le 1$, $\varepsilon \in (0, 1)$.

Step 1: Find i such that $q_1 \le q_2 \le \ldots \le q_i \le \varepsilon \le q_{i+1} \le q_{i+2} \le \cdots \le q_n$.

Step 2: Apply BP-PTA$_\varepsilon$ on the input $(q_{i+1}, \ldots, q_n)$. Let $T = (T_1, \ldots, T_m)$ be the output BP-PTA$_\varepsilon(q_{i+1}, \ldots, q_n)$.

Step 3: For every i such that $\sum_{j \in T_i} q_j \le 1$ pack one of the small pieces from $\{q_1, \ldots, q_i\}$ into T_i until $\sum_{j \in T_i} q_j > 1$ for all $j \in \{1, 2, \ldots, n\}$.
If there are still some small pieces to be assigned, take a new bin and pack the pieces there until this bin is overfilled. Repeat this last step several times, if necessary.

Proof. First, we analyze the time complexity of Bin-PTAS. Step 1 can be executed in linear time. (If one needs to sort the input values, then it takes $O(n \log n)$ time.) Following Lemma 4.3.6.3 the application of BP-PTA$_\varepsilon$ on the input values larger than ε runs in time polynomial according to n. Step 3 can be implemented in linear time.

Now we have to prove that for every input (I, ε), $I = (q_1, \ldots, q_n)$, $\varepsilon \in (0, 1)$,

(i) $cost(\text{Bin-PTAS}(I, \varepsilon)) \leq Opt_{\text{Bin-P}}(I)$, and
(ii) every bin of Bin-PTAS(I, ε) has a size of at most $1 + \varepsilon$.

The condition (ii) is obviously fulfilled because BP-PTA$_\varepsilon$ is an h-dual ε-approximation algorithm, i.e., the bins of BP-PTA$_\varepsilon(q_{i+1}, \ldots, q_n)$ have a size of at most $1 + \varepsilon$. One can easily observe that the small pieces $q_1, \ldots, q_i$ are added to BP-PTA$_\varepsilon(q_{i+1}, \ldots, q_n)$ in Step 3 in such a way that no bin has a size greater than $1 + \varepsilon$.

To prove (i) we first observe that (Lemma 4.3.6.3)

$$Opt_{\text{Bin-P}}(q_{i+1}, \ldots, q_n) \geq cost(\text{BP-PTA}_\varepsilon(q_{i+1}, \ldots, q_n)).$$

Now, if one adds a new bin in Step 3 of Bin-PTAS, then it means that all bins have sizes larger than 1. Thus, the sum of the capacities (sizes) of these bins is larger than its number and so any optimal solution must contain one bin more. $\square$

问题3：PTAS for MS

- BIN-P和MS是如何相互转化的？

$$Opt_{\text{Bin-P}}\left(\frac{p_1}{d}, \frac{p_2}{d}, \ldots, \frac{p_n}{d}\right) \le m \Leftrightarrow Opt_{\text{MS}}(I, m) \le d.$$

- 掌握BIN-P的解法之后，如何求解MS？
  你能解释算法4.3.6.7吗？

Algorithm 4.3.6.7.

Input: $((I,m),\varepsilon)$, where $I = (p_1, \ldots, p_n)$, for some $n \in \mathbb{N}$, $p_1, \ldots, p_n$, m are positive integers, and $\varepsilon > 0$.

Step 1: Compute $ATLEAST := \max\left\{\frac{1}{m}\sum_{i=1}^{n} p_i, \max\{p_1, \ldots, p_n\}\right\}$;
Set $LOWER := ATLEAST$;
$UPPER := 2 \cdot ATLEAST$;
$k := \lceil \log_2(4/\varepsilon) \rceil$.

Step 2: **for** $i = 1$ **to** k **do**
  do begin $d := \frac{1}{2}(UPPER + LOWER)$;
    call Bin-PTAS$_{\varepsilon/2}$ on the input $\left(\frac{p_1}{d}, \frac{p_2}{d}, \ldots, \frac{p_n}{d}\right)$;
    $c := cost\left(\text{Bin-PTAS}_{\varepsilon/2}\left(\frac{p_1}{d}, \ldots, \frac{p_n}{d}\right)\right)$
    if $c > m$ **then** $LOWER := d$
      else $UPPER := d$
  end

Step 3: Set $d^* := UPPER$;
call Bin-PTAS$_{\varepsilon/2}$ on the input $\left(\frac{p_1}{d^*}, \ldots, \frac{p_n}{d^*}\right)$.

Output: Bin-PTAS$_{\varepsilon/2}\left(\frac{p_1}{d^*}, \ldots, \frac{p_n}{d^*}\right)$.

问题4：(近似)算法复习

- 你还记得这些解法吗？它们的要点分别是什么？
  - 伪多项式算法
  - 分支-界限算法
  - 局部搜索算法
  - 松弛算法
  - 贪心算法
  - PTAS
  - 稳定性
  - 对偶近似算法