УДК 519.854.2

Алгоритмы решения NP-трудной проблемы минимизации суммарного запаздывания для одного прибора

©2006 г. Лазарев А.А.[1,2], Кварацхелия А.Г., Гафаров Е.Р.

Москва, Вычислительный Центр им. А.А. Дородницына РАН

Поступила в редакцию 2006

Аннотация

В работе рассматривается классическая NP-трудная в обычном смысле проблема теории расписаний минимизации суммарного запаздывания для одного прибора $1 \mid\mid \sum T_j$. Для NP-трудного случая задачи предложена процедура его разбиения на частные подслучаи, для которых приводятся полиномиальные и псевдополиномиальные алгоритмы решения, трудоемкости не превышающей $O(n^2 \sum p_j)$.

1 Введение

В работе рассматривается NP-трудная в обычном смысле проблема теории расписаний минимизации суммарного запаздывания для одного прибора $1 \mid\mid \sum T_j$. Необходимо обслужить n требований на одном приборе. Прерывания при обслуживании и обслуживание более одного требования в любой момент времени запрещены. Для требования $j \in N = \{1, 2, \ldots, n\}$ заданы продолжительность

[1]Работа выполнена в рамках научной школы академика Ю.И. Журавлёва, НШ-5833.2006.1

[2]Partially supported by DAAD (Deutsche Akademische Austauschdienst): A0206237/Ref. 325

обслуживания $p_j \in \mathbb{Z}^+$ и директивный срок окончания обслуживания d_j, где N - множество требований. Задан момент освобождения прибора t_0, с которого прибор готов начать обслуживание требований. Расписание обслуживания требований π строится с момента времени t_0 и однозначно задается перестановкой элементов множества N.

Требуется построить расписание π^* обслуживания требований множества N, при котором достигается минимум функции $F(\pi) = \sum_{j=1}^{n} \max\{0, c_j(\pi) - d_j\}$, где $c_j(\pi)$ - момент завершения обслуживания требования j при расписании π. Пусть $\pi = (j_1, j_2, \ldots, j_n)$, тогда $c_{j_1}(\pi) = t_0 + p_{j_1}$ и $c_{j_k}(\pi) = c_{j_{k-1}}(\pi) + p_{j_k}$ для $k = 2, 3, \ldots, n$. Величина $T_j(\pi) = \max\{0, c_j(\pi) - d_j\}$ называется *запаздыванием* требования j при расписании π, а $F(\pi)$ – *суммарным запаздыванием* требований при расписании π.

Исследуемая проблема является NP-трудной в обычном смысле [1]. Лаулер (Lawler [2]) предложил псевдополиномиальный алгоритм решения общего случая проблемы трудоемкости $O(n^4 \sum p_j)$. Шварц и др. (Szwarc et all [3]) построили алгоритмы решения проблемы, которые были протестированы для примеров $n < 600$ (тестовые примеры Поттса и Вассенхова (Potts and van Wassenhove [4])). Исследование приближенных алгоритмов решения проблемы было проведено в работе [5], где построены примеры, на которых известные приближенные алгоритмы находят решение с относительной погрешностью порядка размерности примера n.

В работе исследуется частный случай проблемы, когда параметры требований удовлетворяют условиям:

$$\begin{cases} p_1 \geq p_2 \geq \ldots \geq p_n, \\ d_1 \leq d_2 \leq \ldots \leq d_n. \end{cases} \tag{1}$$

Предлагается процедура разбиения множества требований

произвольного примера проблемы, удовлетворяющего условиям (1), и последующего его решения с учетом количества подмножеств, полученных с помощью процедуры разбиения.

Мы полагаем, что классу примеров, параметры требований которых удовлетворяют (1), принадлежат "наиболее трудоемкие"для решения примеры. В частности, "плохие"(в оригинале, bad) примеры из работы [5] удовлетворяют ограничениям (1).

2 Подходы к решению проблемы

В работах [2, 3, 4, 6] применялся декомпозиционный подход к решению примеров общего случая проблемы (без ограничений (1)). При этом исходный пример проблемы разбивается на множество подпримеров, и процесс решения сводится к обходу дерева полученных подпримеров.

Без ограничения общности предположим, что требования множества N пронумерованы в порядке неубывания директивных сроков $d_1 \leq d_2 \leq \ldots \leq d_n$, если $d_k = d_{k+1}$, то $p_k \leq p_{k+1}$.

Процедура разбиения множества требований на подмножества

0. $k := 1$, $\alpha_k := 1$;

1. **for** $j = 2, 3, \ldots, n$ **do:**

if $d_j - d_{\alpha_k} > p_j$ **then** $\beta_k := j - 1$; $k := k + 1$; $\alpha_k := j$;

2. $\beta_k = n$.

Данная процедура однозначно разбивает исходное множество N на k подмножеств, $k \leq n$, за $O(n)$ операций. В результате работы процедуры будут выделены подмножества требований

$M_1, M_2, \ldots, M_k$, где $M_i = \{\alpha_i, \alpha_i + 1, \ldots, \beta_i\}$, $\alpha_1 = 1, \beta_k = n$. При этом, $M_1 \bigcup M_2 \bigcup \ldots \bigcup M_k = N$ и $M_i \bigcap M_j = \emptyset$, если $i \neq j$.

Например, для множества из трех требований с параметрами: $p_1 = 10, p_2 = 10, p_3 = 2$, $d_1 = 7, d_2 = 9, d_3 = 10$, будет получено разбиение на подмножества $M_1 = \{1, 2\}$ и $M_2 = \{3\}$.

Через $x = \langle \{p_j, d_j\}_{j \in N}, t_0 \rangle$ будем обозначать пример проблемы, для которого требуется построить оптимальное расписание. Известно, что для произвольных примеров $x_1 = \langle \{p_j, d_j\}_{j \in N}, t \rangle$ и $x_2 = \langle \{p_j, d_j + C\}_{j \in N}, t + C \rangle$, где C - константа, множества оптимальных расписаний и оптимальные значения целевой функции совпадают. Такие примеры x_1 и x_2 будем называть *равными примерами*.

В следующих разделах предлагаются алгоритмы решения случая (1) в зависимости от значения k (количества подмножеств, выделенных процедурой разбиения).

3 Случай $k = 1$

В случае $k = 1$ параметры требований исходного примера удовлетворяют соотношениям

$$\begin{cases} p_1 \geq p_2 \geq \ldots \geq p_n, \\ d_1 \leq d_2 \leq \ldots \leq d_n, \\ d_n - d_1 \leq p_n, \end{cases} \tag{2}$$

т. е. $d_1 \leq d_2 \leq \ldots \leq d_n \leq d_1 + p_n$.

3.1 Доказательство NP-трудности случая $k = 1$

Проблема Четно-Нечетное Разбиение (ЧНР). Задано упорядоченное множество из $2n$ положительных целых чисел $B = \{b_1, b_2, \ldots, b_{2n}\}$, $b_i > b_{i+1}$, $1 \leq i \leq 2n - 1$. Требуется определить,

существует ли разбиение множества B на два подмножества B_1 и B_2, такое, что $\sum_{b_i \in B_1} b_i = \sum_{b_i \in B_2} b_i$ и для каждого $i = 1, 2, \ldots, n$ подмножество B_1 (следовательно, и B_2) содержит в точности один элемент из пары $\{b_{2i-1}, b_{2i}\}$.

Обозначим $\delta_i = b_{2i-1} - b_{2i},\ i = 1, \ldots, n,\ \delta = \sum_{i=1}^{n} \delta_i$.

Построим модифицированный пример Четно-Нечетного Разбиения.

$$\begin{cases} a_{2n} = M + b, \\ a_{2i} = a_{2i+2} + b,\ i = n-1, \ldots, 1, \\ a_{2i-1} = a_{2i} + \delta_i,\ i = n, \ldots, 1, \end{cases}$$

где $b \gg n\delta,\ M \geq n^3 b$.

Очевидно, выполняется $a_i > a_{i+1},\ \forall i = 1, 2, \ldots, 2n-1, \delta_i = b_{2i-1} - b_{2i} = a_{2i-1} - a_{2i},\ i = 1, \ldots, n$.

Лемма 1 *Исходный пример* **ЧНР** *имеет решение "ДА" тогда и только тогда, когда модифицированный пример* **ЧНР** *имеет решение "ДА".*

Приведем полиномиальную схему сведения модифицированного примера **ЧНР** к частному случаю (2) задачи $\mathbf{1} \mid\mid \sum \boldsymbol{T_j}$. Количество требований $(2n + 1)$. Требования обозначим следующим образом $V_1, V_2, V_3, V_4, \ldots, V_{2i-1}, V_{2i}, \ldots, V_{2n-1}, V_{2n}, V_{2n+1}$, $N = \{1, 2, \ldots, 2n, 2n+1\}$. Для упрощения записи будем использовать следующие обозначения $p_{V_i} = p_i,\ d_{V_i} = d_i,\ T_{V_i} = T_i,\ c_{V_i} = c_i,\ i = 1, \ldots, 2n+1$. Пример, удовлетворяющий следующим ограничениям, будем называть *каноническим LG примером.*

$$
\begin{cases}
p_1 > p_2 > \ldots > p_{2n+1}, \\
d_1 < d_2 < \ldots < d_{2n+1}, \\
d_{2n+1} - d_1 < p_{2n+1}, \\
p_{2n+1} = M = n^3 b, \\
p_{2n} = p_{2n+1} + b = a_{2n}, \\
p_{2i} = p_{2i+2} + b = a_{2i},\ i = n-1, \ldots, 1, \\
p_{2i-1} = p_{2i} + \delta_i = a_{2i-1},\ i = n, \ldots, 1, \\
d_{2n+1} = \sum_{i:=1}^{n} p_{2i} + p_{2n+1} + \frac{1}{2}\delta, \\
d_{2n} = d_{2n+1} - \delta, \\
d_{2i} = d_{2i+2} - (n-i)b + \delta,\ i = n-1, \ldots, 1, \\
d_{2i-1} = d_{2i} - (n-i)\delta_i - \varepsilon\delta_i,\ i = n, \ldots, 1,
\end{cases}
\tag{3}
$$

где $b = n^2\delta$, $0 < \varepsilon < \frac{\min_i \delta_i}{\max_i \delta_i}$.

Директивные сроки примера (3) представлены на рис. 1.

Необходимо отметить, что канонические примеры DL из работы [1] не удовлетворяют случаю (3). Первые три неравенства показывают, что (3) является подслучаем (2).

Лемма 2 *Для случая (3) при любом расписании количество запаздывающих требований равно или n, или $(n+1)$.*

Расписание $(V_{1,1}, V_{2,1}, \ldots, V_{i,1}, \ldots, V_{n,1}, V_{2n+1}, V_{n,2}, \ldots, V_{i,2}, \ldots, V_{2,2}, V_{1,2})$ будем называть *каноническим LG* расписанием, где $\{V_{i,1}, V_{i,2}\} = \{V_{2i-1}, V_{2i}\},\ i = 1, 2, \ldots, n$.

Теорема 1 *Для случая (3) все оптимальные расписания являются каноническими или могут быть преобразованы к каноническим расписаниям применением правила EDD к $(n+1)$ требованиям, обслуживаемым в начале расписания.*

При расписании π_{EDD} требования множества $\{\pi_{EDD}\}$ упорядочены в порядке неубывания директивных сроков.

Теорема 2 *Решение примера ЧНР будет "ДА" тогда и только тогда, когда при оптимальном каноническом расписании* $c_{2n+1}(\pi) = d_{2n+1}$.

Доказательство утверждений представлено в работе [7].

3.2 Алгоритм решения для случая $k = 1$

Далее в работе будем использовать следующие обозначения. Пусть $d_j(t) = d_j - d_n + t - t_0,\ j \in N$, при этом, с учетом (1), для любого $t \in \mathbb{R}$ выполняется $d_1(t) \leq d_2(t) \leq \ldots \leq d_n(t)$. Пусть $\pi_l(t)$ и $F_l(t)$ оптимальное расписание и соответствующее ему значение суммарного запаздывания для примера с множеством требований $N_l = \{l, \ldots, n\}$, моментом начала обслуживания равным 0, и директивными сроками окончания обслуживания $d_j(t)$, $j = l, \ldots, n$, $l = n, \ldots, 1$.

Согласно определению величин $d_j(t)$, пример $\langle\{p_j, d_j(t)\}_{j\in N}, 0\rangle$ эквивалентен примеру $\langle\{p_j, d_j\}_{j\in N}, d_n - t + t_0)\rangle$. Также, пример $\langle\{p_j, d_j(t)\}_{j\in N}, 0\rangle$ при $t = d_n$ совпадает с исходным примером $\langle\{p_j, d_j\}_{j\in N}, t_0)\rangle$.

Приведем алгоритм решения примеров, параметры требований которых удовлетворяют условиям (2).

Алгоритм B-1

1: $\pi_n(t) := (n)$, $F_n(t) := \max\{0, p_n - t + t_0\}$;
2: **for** $k = n-1, n-2, \ldots, 1$ **do**
3: $\quad \pi^1 := (k, \pi_{k+1}(t - p_k))$; $\pi^2 := (\pi_{k+1}(t), k)$;
4: $\quad F(\pi^1) := \max\{0, p_k - d_k(t)\} + F_{k+1}(t - p_k)$;
5: $\quad F(\pi^2) := F_{k+1}(t) + \max\{0, \sum_{j=k}^{n} p_j - d_k(t)\}$;

6: $F_k(t) := \min\{F(\pi^1), F(\pi^2)\}$; $\pi_k(t) := \arg\min\{F(\pi^1), F(\pi^2)\}$;

7: **end for**

В Алгоритме B-1 на каждом шаге для всех возможных точек $t \in [0, \sum p_j]$ строятся расписания $\pi_l(t)$ путем включения требования l на первое или последнее место в ранее построенные расписания $\pi_{l+1}(t - p_l)$ или $\pi_{l+1}(t)$, соответственно, и выбирается наилучшее из двух получившихся расписаний.

Теорема 3 *Расписание $\pi_1(d_n)$ является оптимальным для случая (2). Алгоритм B-1 строит расписание $\pi_1(d_n)$ за $O(n \sum p_j)$ операций.*

При реализации Алгоритма B-1 необходимо учитывать тот факт, что величина d_n должна быть целочислена. Поэтому, если $d_n \notin \mathbb{Z}$, предлагается построить и решить с помощью Алгоритма *B-1* следующий пример: $\langle \{p_j, d'_j\}_{j \in N}, t'_0 \rangle$, $d'_j = d_j - \Delta$, $j \in N$, $t'_0 = t_0 - \Delta$, где $\Delta = d_n - \lfloor d_n \rfloor$, где $\lfloor a \rfloor$ есть целая часть числа a. В этом случае величина d'_n является целочисленной. Исходный и построенный пример эквивалентны между собой, следовательно, расписание $\pi_1(d'_n)$ является оптимальным для исходного примера $\langle \{p_j, d_j\}_{j \in N}, t_0 \rangle$.

3.3 Алгоритм B-1 модифицированный.

Кусочно-линейную функцию $F_l(t)$ можно представить в виде: $F_l(t) = -\alpha_i t + b_i, t \in [t_i, t_{i+1}]$, $i = 1, 2, \ldots, m$, где t_i – точки "излома" функции, а α_i – количество запаздывающих требований при оптимальном частичном расписании $\pi_l(t)$ на интервале $[t_i, t_{i+1}]$.

Алгоритм B-1 модифицированный перебирает не все целочисленные точки t из интервала $[0, \sum_{j=1}^{n} p_j]$, а лишь точки "излома" графика функции $F_l(t)$. Трудоемкость алгоритма полиномиально зависит от числа точек "излома".

3.4 Случай $k = 1$ и $d_n - d_1 \leq 1$

Предполагается, что параметры требований удовлетворяют условиям:

$$\begin{cases} d_1 \leq d_2 \leq \ldots \leq d_n, \\ d_n - d_1 \leq 1, \\ p_1, \ldots, p_n \in \mathbb{Z}^+. \end{cases} \tag{4}$$

Построение оптимального расписания по алгоритму ***C-1*** происходит "с конца". Сначала в интервале $[\lfloor d_{max} \rfloor + 1, t_0 + \sum_{j=1}^{n} p_j]$ упорядочиваются требования по неубыванию директивных сроков. Затем из "оставшихся"требований перебираются всевозможные пары требований "покрывающие"интервал $[d_{min}, d_{max}]$ и выбирается пара, на которой достигается минимальное значение целевой функции. Оставшиеся требования упорядочиваются в произвольном порядке. Алгоритм ***C-1*** за $O(n^2)$ операций для случая (4) находит оптимальное расписание.

4 Алгоритм решения в случае $1 < k < n$

В случае $1 < k < n$ параметры требований удовлетворяют условиям

$$\begin{cases} d_1 \leq d_2 \leq \ldots \leq d_n, \\ p_1 \geq p_2 \geq \ldots \geq p_n, \\ d_{\beta_1} - d_{\alpha_1} \leq p_{\beta_1}, \qquad \alpha_1 = 1, \\ d_{\beta_2} - d_{\alpha_2} \leq p_{\beta_2}, \qquad \alpha_2 = \beta_1 + 1, \\ \ldots \\ d_{\beta_k} - d_{\alpha_k} \leq p_{\beta_k}, \qquad \beta_k = n. \end{cases} \tag{5}$$

В Алгоритме B-k на каждом шаге для всех возможных точек $t \in [0, \sum p_j]$ строятся расписания $\pi_l(t)$ путем включения требования l

между подмножествами и выбором наилучшего из не более чем $k+1$ частичных расписаний.

Теорема 4 *Расписание $\pi_1(d_n)$ является оптимальным расписанием для случая (5). Алгоритм B-k строит расписание $\pi_1(d_n)$ за $O(k\,n\sum p_j)$ операций.*

Заметим, что если $d_n \notin \mathbb{Z}$, то построим и решим, по аналогии со случаем $k = 1$, пример $\langle \{p_j, d'_j\}_{j\in N}, t'_0\rangle$, где $d'_j = d_j - \Delta$, $j \in N$, $t'_0 = t_0 - \Delta$, $\Delta = d_n - \lfloor d_n \rfloor$.

5 Алгоритм решения в случае $k = n$

В случае $k = n$ параметры требований удовлетворяют следующим условиям $d_j - d_{j-1} > p_j$, $j = 2, 3, \ldots, n$, без ограничений на продолжительность обслуживания требований и их целочисленность.

Для данного случая предлагается Алгоритм *B-n* трудоемкости $O(n^2)$ построения оптимального расписания.

В основе алгоритма лежит процедура ProcB-n (N, t) упорядочивания множества требований $N = \{1, 2, \ldots, n\}$, $d_1 < d_2 < \ldots < d_n$, с момента времени t. Для требования $j^* = \arg\max_{j\in N}\{d_j : p_j = \max_{i\in N} p_i\}$ за время $O(n)$ выбирается позиция $\alpha^* := \arg\min\limits_{\alpha=1,\ldots,n}\{d_j + p_j \le S_\alpha < d_{\alpha+1},\ j \in \{j^*+1, \ldots, \alpha\}\}$, где $S_\alpha := t + p_1 + p_2 + \ldots + p_\alpha$. Задача сводится к аналогичному упорядочиванию двух множеств $N' := \{1, \ldots, \alpha^*\} \setminus \{j^*\}$ с момента времени t и $N'' := \{\alpha^*+1, \ldots, n\}$ с момента времени $t'' := S_{\alpha^*}$.

Теорема 5 *Алгоритм B-n строит оптимальное расписание для случая $k = n$ за $O(n^2)$ операций.*

Список литературы

[1] Du J. and Leung J. Y.-T. *Minimizing total tardiness on one processor is NP-hard* // Math. Oper. Res.. 1990. № 15.

[2] Lawler E.L. *A pseudopolynomial algorithm for sequencing jobs to minimize total tardiness* // Ann. Discrete Math.. 1977. № 1.

[3] Szwarc W., Grosso A. and Della Croce F. *Algorithmic paradoxes of the single machine total tardiness problem* // Journal of Scheduling. 2001. № 4.

[4] Potts C.N. and van Wassenhove L.N. *A decomposition algorithm for the single machine total tardiness problem* // Oper. Res. Lett.. 1982. № 1.

[5] Della Croce F., Grosso A., Paschos V. *Lower bounds on the approximation ratios of leading heuristics for the single-machine total tardiness problem* // Journal of Scheduling. 2004. № 7.

[6] Lazarev A., Kvaratskhelia A., Tchernykh A. *Solution algorithms for the total tardiness scheduling problem on a single machine* // Workshop Proceedings of the ENC'04 International Conference. 2004.

[7] Лазарев А.А., Гафаров Е. Р., *Доказательство NP-трудности частного случая задачи минимизация суммарного запаздывания для одного прибора* $1||\sum T_j$. // Известия РАН: Теория и системы управления. 2006. (№ 3, в печати).

Вычислительный Центр им. А.А. Дородницына РАН
Лазарев Александр Алексеевич, Кварацхелия Александр Гонерович,
Гафаров Евгений Рашидович
119991, г. Москва, ул. Вавилова, 40,
телефон 135 6238, e-mail: Alexandr.Lazarev@ksu.ru

Список рисунков.

Лазарев А.А., Кварацхелия А.Г., Гафаров Е.Р. Алгоритмы решения NP-трудной проблемы минимизации суммарного запаздывания для одного прибора. В работе рассматривается классическая NP-трудная в обычном смысле проблема теории расписаний минимизации суммарного запаздывания для одного прибора $1 \mid\mid \sum T_j$. Для NP-трудного случая задачи предложена процедура его разбиения на частные подслучаи, для которых приводятся полиномиальные и псевдополиномиальные алгоритмы решения, трудоемкости не превышающей $O(n^2 \sum p_j)$.

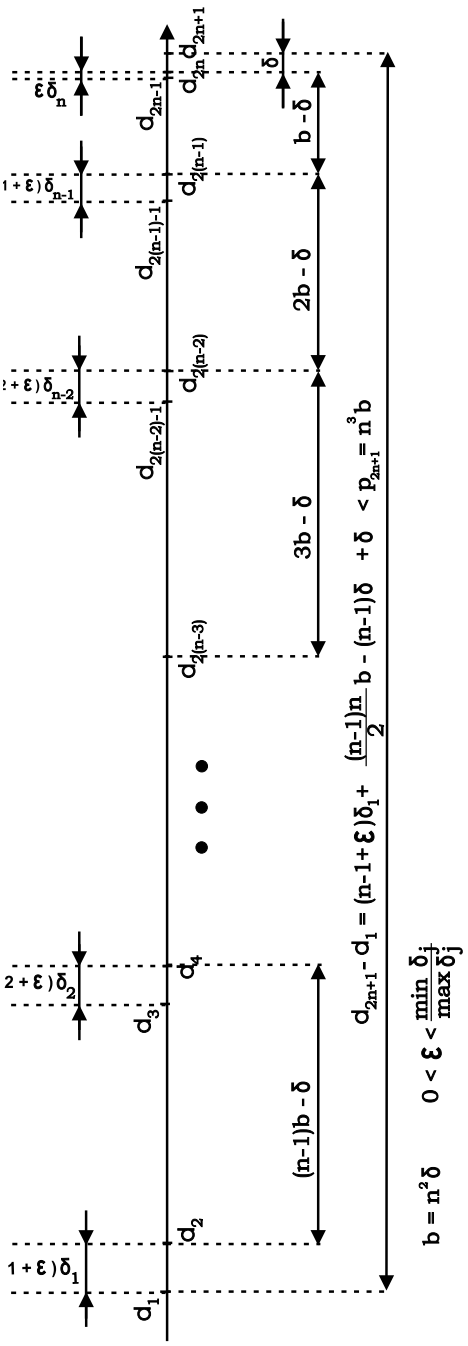

Рис. 1: Директивные сроки канонического LG примера.

Лазарев А.А., Кварацхелия А.Г., Гафаров Е.Р. Алгоритмы решения NP-трудной проблемы минимизации суммарного запаздывания для одного прибора. В работе рассматривается классическая NP-трудная в обычном смысле проблема теории расписаний минимизации суммарного запаздывания для одного прибора $1 \mid\mid \sum T_j$. Для NP-трудного случая задачи предложена процедура его разбиения на частные подслучаи, для которых приводятся полиномиальные и псевдополиномиальные алгоритмы решения, трудоемкости не превышающей $O(n^2 \sum p_j)$.

Lazarev A.A., Kvaratskhelia A.G., Gafarov E.R.
Algorithms of decision of NP-hard problem minimizing total tardiness for single machine.