© 2021 г. А.А. САРАТОВ, канд. тех. наук (sapfor58@.gmail.com)
(ООО "Интерактивные системы автоматизации проектирования", Тула)

ЖАДНЫЙ АЛГОРИТМ РЕШЕНИЯ КЛАССИЧЕСКОЙ NP-ТРУДНОЙ ЗАДАЧИ ТЕОРИИ РАСПИСАНИЙ МИНИМИЗАЦИИ СУММАРНОГО ЗАПАЗДЫВАНИЯ

Представлен эффективный метод решения классической NP-трудной задачи теории расписаний для одного прибора минимизации суммарного запаздывания $1||\sum T_j$. Предлагается алгоритм решения задачи, основанный на декомпозиции исходной задачи на подзадачи своевременного обслуживания каждого из требований и размещении в конец расписаний тех из них, прирост запаздывания которых в наибольшей степени компенсируется сокращением запаздываний предшествующих требований. Трудоемкость алгоритма не превышает $O\left(n^2\right)$ операций, где n – количество требований.

Ключевые слова: теория расписаний, один прибор, минимизация суммарного запаздывания, жадные алгоритмы.

DOI: 10.31857/S0005231021110064

Рассматривается классическая NP-трудная в обычном смысле задача теории расписания, в которой необходимо обслужить множество требований $N = \{1, \ldots, , n\}$ на одном приборе [1]. Прерывания при обслуживании и обслуживание более одного требования в любой момент времени запрещены. Для требования $j \in N$ заданы продолжительность обслуживания $p_j > 0$ и директивный срок окончания обслуживания d_j. Все требования поступают на обслуживание одновременно в момент времени t_0, с которого прибор готов начать обслуживание этих требований. Расписание обслуживания требований 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 $\pi = (j_1, \ldots, j_n)$, тогда $c_{j_1}(\pi) = t_0 + p_{j_1}$ и $c_{j_k}(\pi) = c_{j_{k-1}} + p_{j_k}$ 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. Если обслуживание требования i предшествует обслуживанию требования j, то для этого будем использовать запись $(i \to j)_\pi$. Если

обслуживание требования i переносится на более ранний или поздний срок, то будем называть это смещением влево или вправо.

К настоящему времени опубликовано большое число работ, посвященных подходам к решению данной задачи [1, 2]. Все предложенные решения не позволяют получить точное решение задачи за приемлемое для практического использования время. Здесь предлагается метод синтеза оптимального расписания, основанный на декомпозиции исходной задачи на подзадачи своевременного обслуживания каждого из требований и размещении в конец расписаний тех из них, прирост запаздывания которых в наибольшей степени компенсируется сокращением запаздываний предшествующих требований.

Поскольку в оптимальном расписании π^* любое перемещение требования j и любая парная перестановка требований i, j не могут привести к уменьшению $F(\pi)$, то для всех $i, j \in \pi^*$ должно выполняться условие:

$$(1)\quad \begin{cases} \nabla T'_j(\pi) \geq \sum\limits_{i=j+1}^{j+k} \nabla T'_i(\pi), & j > i \\ \nabla T''_j(\pi) \leq \sum\limits_{i=j-k}^{j-1} \nabla T''_i(\pi), & j < i \end{cases} \wedge \begin{cases} \nabla T_{ij}(\pi) \geq \sum\limits_{v=i+1}^{j-1} \nabla T'_v(\pi), & p_i > p_j \\ \nabla T_{ji}(\pi) \leq \sum\limits_{v=i+1}^{j-1} \nabla T''_v(\pi), & p_i < p_j, \end{cases}$$

где

$\nabla T'_j(\pi) \nabla T''_j(\pi)$ — изменение запаздывания требования j при перемещении j в очереди на k позиций соответственно вправо и влево;

$\sum_{j+1}^{j+k} \nabla T'_i(\pi)$ — суммарное уменьшение запаздывания требований $i = \{j+1, \ldots, j+k\} \in N$ при их перемещении в очереди на k позиций влево;

$$(2)\quad \sum_{j+1}^{j+k} \nabla T'_i(\pi) = \sum_{j+1}^{j+k} \left\{ \max\left[0, \left(t_0 + \sum_{m=0}^{i} p_m - d_i\right)\right] - \max\left[0, \left(t_0 + \sum_{m=0}^{i} p_m - p_j - d_i\right)\right] \right\};$$

$\sum_{j-k}^{j-1} \nabla T''_i(\pi)$ — суммарное увеличение запаздывания требований $i = \{j-k, \ldots, j-1\} \in N$ при их перемещении в очереди на k позиций вправо;

$$(3)\quad \sum_{j-k}^{j-1} \nabla T''_i(\pi) = \sum_{j-k}^{j-1} \left\{ \max\left[0, \left(t_0 + \sum_{m=0}^{i} p_m + p_j - d_i\right)\right] - \max\left[0, \left(t_0 + \sum_{m=0}^{i} p_m - d_i\right)\right] \right\};$$

$\nabla T_{ij}(\pi)$ — увеличение запаздывания требования i при замещении j;

$\nabla T_{ji}(\pi)$ — уменьшение запаздывания требования j при замещении i;

$\sum_{i+1}^{j-1} {}_{\nabla}T'_v(\pi)$ — суммарное уменьшение запаздывания требований $v = \{i+1, \ldots, j-1\}$ при перестановке требований i, j и $p_i > p_j$;

$\sum_{i+1}^{j-1} {}_{\nabla}T''_v(\pi)$ — суммарное увеличение запаздывания требований $v = \{i+1, \ldots, j-1\}$ при перестановке требований i, j и $p_i < p_j$.

Множества перемещаемых требований формул (2), (3) будем называть частичными расписаниями $\pi^{\wedge}$ и $\pi^{\wedge\wedge}$;

$$\pi^{\wedge} = \{j+1, \ldots, j+k\} \in N, \quad \pi^{\wedge\wedge} = \{j-k, \ldots, j-1\} \in N.$$

Выражение (1) при $k = 1, \ldots, n-1$, является эффективным инструментом оценки оптимальности построенного расписания, ибо вычисляется за $0\left(n^2\right)$ операций.

Для обеспечения эффективного процесса синтеза π^* на основе формулы (1) необходим быстрый алгоритм формирования частичных расписаний $\pi^{\wedge}$ и $\pi^{\wedge\wedge}$, эквивалентных по соотношению суммарных смещений $\sum_{i+1}^{i+k} {}_{\nabla}T'_i(\pi)$, $\sum_{i-k}^{i-1} {}_{\nabla}T''_i(\pi)$ с суммарными смещениями в расписании π^*. Такая эквивалентность может быть достигнута в частичных расписаниях $\pi^{\wedge}$, $\pi^{\wedge\wedge}$, отвечающих (1) при $k = 1$. Истинность выражения (1) при $k = 1$ отражает известный факт, что в оптимальном расписании перестановка двух смежных требований не может уменьшить их суммарного запаздывания и является необходимым условием для проверки оптимальности π^*. Очередность обслуживания требований $j \in \pi^{\wedge}$ определяется соотношениями их параметров d_j, p_j, поэтому расписания, отвечающие условию (1) при $k = 1$, будем называть локально оптимальными.

Рассмотрим метод построения расписания π^* начиная от последнего требования к первому, используя процедуры формирования $\pi^{\wedge}$.

Алгоритм A построения $\pi^{\wedge}$ имеет вид.

1. Принимаем время старта $t = t_0$.
2. Моделируем попарно перестановки требований $i \to j$, $j \to i$ и вычисляем сумму ${}_{\nabla}T_{ij} = T_i(\pi) + T_j(\pi)$. Если при $i \to j$ суммарное смещение меньше, чем при $j \to i$, то i помечается, как приоритетное требование i^* и продолжает участвовать в перестановках, а j исключается. Если суммарные смещения для $i \to j$ и $j \to i$ равны, то помечается требование с меньшим d_j.
3. Исключаем i^* из N и помещаем в N^*, добавляя список справа.
4. Принимаем время старта $t = t_0 + p_i$.
5. Повторяем шаги 2–4, пока $N \neq \varnothing$.

Для канонических DL примеров [1] алгоритм A достаточен для синтеза оптимального расписания. Рассмотрим работу алгоритма A на одном из таких примеров (рис. 1).

В канонических LG расписаниях [1] первые $n/2-1$ требований не опаздывают, и при построении таких расписаний можно использовать соотношения прироста запаздываний от перемещения требований вправо с уменьшением суммарных запаздываний требований, перемещаемых влево на освоившиеся места (2).

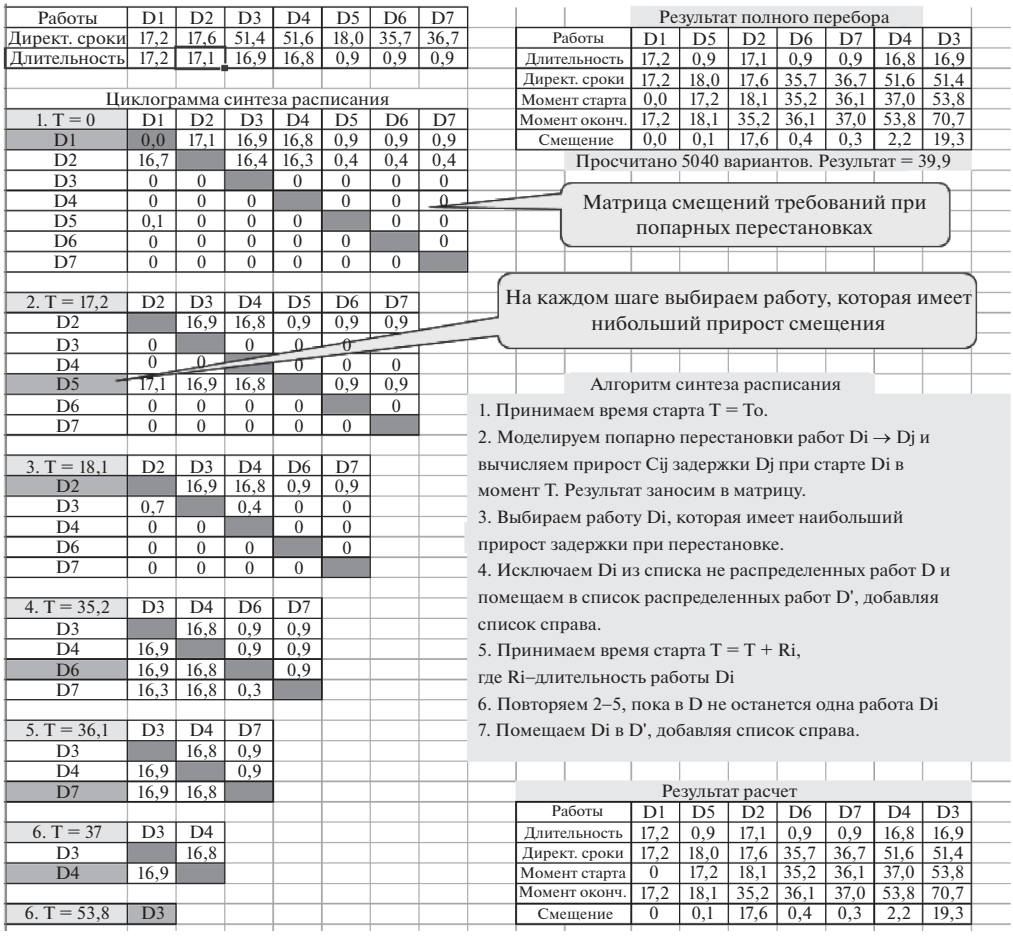

Рис. 1. Решение канонического DL-примера.

Обозначим как π_j расписание, состоящее из $\pi^\wedge$ и следующего за ним требования j, а суммарное запаздывание требований π_j — как $F(\pi_j)$.

Если принять условие (1) достаточным для проверки оптимальности $\pi^\wedge$, то в расписании $\pi_j = (\pi^\wedge, j)$ с минимальным суммарным смещением $F(\pi_j)$ требование $j \in \pi_j$ будет последним в оптимальном расписании π^*. Исходя из этого, алгоритм построения π^* имеет вид:

1. Для каждого требования j формируем расписание $\pi_j = (\pi^\wedge, j)$, в котором j размещается в конце очереди, а остальные работы упорядочены согласно алгоритму A. Вычисляем суммарные смещения $F(\pi)$.
2. Выбираем расписание $\pi_j = (\pi^\wedge, j)$ с минимальным $F(\pi)$.
3. Исключаем j из N и помещаем в список последних требований $N^* \in \pi^*$, добавляя список слева.
4. Повторяем шаги 1, 2, 3, пока в N останутся только успевающие работы.
5. Объединяем N и N^*.
6. Конец.

1	02	03	05	04	01	
p	20,00	18,10	16,00	18,00	20,10	
d	52,25	53,70	54,25	53,75	52,10	
t	0	20,00	38,10	54,10	72,10	
t'	20,00	38,10	54,10	72,10	92,20	
T	0	0	0	18,35	40,10	58,45

7	01	04	05	03	
p	20,10	18,00	16,00	18,10	
d	52,10	53,75	54,25	53,70	
t	0	20,10	38,10	54,10	
t'	20,10	38,10	54,10	72,20	
T	0	0	0	18,50	18,50

2	01	03	05	04	02	
p	20,10	18,10	16,00	18,00	20,00	
d	52,10	53,70	54,25	53,75	52,25	
t	0	20,10	38,20	54,20	72,20	
t'	20,10	38,20	54,20	72,20	92,20	
T	0	0	0	18,45	39.95	58,40

8	01	03	05	04	
p	20,10	18,10	16,00	18,00	
d	52,10	53,70	54,25	53,75	
t	0	20,10	38,20	54,20	
t'	20,10	38,20	54,20	72,20	
T	0	0	0	18,45	18,45

3	D1	D2	D5	D4	D3	
p	20,10	20,00	16,00	18,00	18,10	
d	52,10	52,25	54,25	53,75	53,70	
t	0,00	20,10	40,10	56,10	74,10	
t'	20,10	40,10	56,10	74,10	92,20	
T	0,00	0,00	1,85	20,35	38,50	60,70

9	D1	D3	D4	D5	
p	20,10	18,10	18,00	16,00	
d	52,10	53,70	53,75	54,25	
t	0,00	20,10	38,20	56,20	
t'	20,10	38,20	56,20	72,20	
T	0,00	0,00	2,45	17,95	20,40

4	D1	D2	D5	D3	D4	
p	20,10	20,00	16,00	18,10	18,00	
d	52,10	52,25	54,25	53,70	53,75	
t	0,00	20,10	40,10	56,10	74,20	
t'	20,10	40,10	56,10	74,20	92,20	
T	0,00	0,00	1,85	20,50	38,45	60,80

10	D3	D5	D1	
p	18,10	16,00	20,10	
d	53,70	54,25	52,10	
t	0,00	18,10	34,10	
t'	18,10	34,10	54,20	
T	0,00	0,00	2,10	2,10

5	D1	D2	D4	D3	D4	
p	20,10	20,00	18,00	18,10	16,00	
d	52,10	52,25	53,75	53,70	54,25	
t	0,00	20,10	40,10	58,10	76,20	
t'	20,10	40,10	58,10	76,20	92,20	
T	0,00	0,00	4,35	22,50	37,95	64,80

11	D1	D5	D3	
p	20,10	16,00	18,10	
d	52,10	54,25	53,70	
t	0,00	20,10	36,10	
t'	20,10	36,10	54,20	
T	0,00	0,00	0,50	0,50

6	D3	D4	D5	D1	
p	18,10	18,00	16,00	20,10	
d	53,70	53,75	54,25	52,10	
t	0,00	18,10	36,10	52,10	
t'	18,10	36,10	52,10	72,20	
T	0,00	0,00	0,00	20,10	20,10

12	D1	D3	D5	
p	20,10	18,10	16,00	
d	52,10	53,70	54,25	
t	0,00	20,10	38,20	
t'	20,10	38,20	54,20	
T	0,00	0,00	0,00	0,00

Итог	D1	D3	D5	D4	D2
p	20,10	18,10	16,00	18,00	20,00
d	52,10	53,70	54,25	53,75	52,25
t	0,00	20,10	38,20	54,20	72,20
t'	20,10	38,20	54,20	72,20	92,20
T	0,00	0,00	0,00	18,45	39,95

Рис. 2. Синтез оптимального расписания для LG примера.

Рассмотрим пример. Пусть имеется 5 требований $N = \{D_1, D_2, D_3, D_4, D_5\}$ с параметрами: $p_1 = 20{,}1$, $p_2 = 20$, $p_3 = 18{,}1$, $p_4 = 18$, $p_5 = 16$, $d_1 = 52{,}1$, $d_2 = 52{,}25$, $d_3 = 53{,}7$, $d_4 = 53{,}75$, $d_5 = 54{,}25$. Необходимо составить расписание с минимальным суммарным запаздыванием (T_i) требований. Процедура синтеза расписания (рис. 2) будет включать 12 шагов:

1. Переносим D_1 в конец расписания, а остальные требования выстраиваем по результатам перестановок $i \to j$, $j \to i$ (алгоритм A). Вычисляем суммарное запаздывание $T = T_4 + T_1 = 18{,}35 + 40{,}1 = 58{,}45$.
2. Выполняем п. 1 для $D_2, \ldots, D_5$ (шаги $2, \ldots, 5$).
3. Выбираем требование (D_2), с наименьшим $T = 58{,}4$, и переносим D_2 из N в N^*.
4. Выполняем пп. 1–3 для $N = \{D_1, D_3, D_4, D_5\}$ (шаги $6, \ldots, 9$). Переносим D_4 из N в N^*.
5. Выполняем пп. 1–3 для $N = \{D_1, D_3, D_5\}$ (шаги $10, \ldots, 12$). Переносим D_5 из N в N^*.
6. Оставшиеся требования $N = \{D_1, D_3\}$ имеют $T = 0$. Переносим D_1, D_3 из N в N^*.
7. Конец.

Полученное расписание $N^* = \{D_1, D_3, D_5, D_4, D_2\}$ имеет минимальное суммарное запаздывание $T = T_1 + T_3 + T_5 + T_4 + T_2 = 18{,}45 + 39{,}95 = 58{,}4$.

Испытания предложенного метода проводились на канонических DL и LG примерах с количеством требований от 5 до 26. Во всех случаях были получены оптимальные решения.

СПИСОК ЛИТЕРАТУРЫ

1. *Лазарев А.А., Гафаров Е.Р.* Теория расписаний. Задачи и алгоритмы. М.: Изд-во МГУ, 2011, 222 с.
2. *Лазарев А.А., Архипов Д.И.* Оценка абсолютной погрешности и полиномиальной разрешимости для классической NP-трудной задачи теории расписаний // Доклады АН. 2018. Т. 480. № 5. С. 523—527.
3. *Саратов А.А.* Согласование производственных циклов методом взаимных штрафов // Автоматизация процессов управления. 2019. № 1. С. 66–73.

Статья представлена к публикации членом редколлегии А.А. Лазаревым.

Поступила в редакцию 27.01.2021

После доработки 28.05.2021

Принята к публикации 30.06.2021