ДИСКРЕТНЫЙ АНАЛИЗ И ИССЛЕДОВАНИЕ ОПЕРАЦИЙ
Октябрь—декабрь 2007. Серия 1. Том 14, № 4. 3–15

УДК 519.854

АЛГОРИТМ С ОЦЕНКАМИ ДЛЯ ПРОПОРЦИОНАЛЬНОГО СЛУЧАЯ ДВУХПРОЦЕССОРНОЙ ЗАДАЧИ ТЕОРИИ РАСПИСАНИЙ ТИПА FLOW SHOP С МИНИМАЛЬНЫМИ ЗАДЕРЖКАМИ*)

А. А. Агеев

Рассматривается двухпроцессорная задача типа flow shop с минимальными задержками. Известно, что эта задача NP-трудна в сильном смысле даже в случае единичных длительностей операций и может быть решена за полиномиальное время с точностью 2 в общем случае. В ряде работ ставился вопрос о существовании полиномиального алгоритма с лучшей оценкой точности, но он до сих пор остаётся открытым. В этой статье предлагается полиномиальный алгоритм с оценкой точности $\frac{3}{2}$ для частного случая задачи, в котором обе операции каждой работы имеют равные длительности (этот случай задач типа flow shop в литературе обычно принято называть пропорциональным). Анализ алгоритма основан на нетривиальном обобщении нижней границы, установленной в работе В. Ю для случая единичных длительностей операций.

Введение

В двухпроцессорной задаче типа flow shop с минимальными задержками имеются две машины, каждая из которых начиная с момента времени 0 готова к выполнению n независимых работ. Каждая машина способна выполнять не более одной работы в любой момент времени. Работа j состоит из двух операций, причём вторая операция может начаться не ранее чем через l_j единиц времени с момента окончания первой операции. Первая (вторая) операция должна выполняться на машине 1 (машине 2) и выполнение первой (второй) операции занимает a_j (b_j) единиц времени. Предполагается, что a_j, b_j и l_j — неотрицательные целые числа. Требуется найти расписание минимальной длины. Следуя [13], для сформулированной задачи будем использовать обозначение $F2 \mid l_j \mid C_{\max}$.

*)Исследование выполнено при финансовой поддержке Российского фонда фундаментальных исследований (проекты 06–01–00255, 05–01–00960).

Задачи с минимальными задержками возникают, в частности, при оптимизации работы производственных цехов, когда существенное значение имеет время транспортировки из одного цеха в другой, а также в многопроцессорных вычислительных комплексах, где передача данных с выхода одного процессора на вход другого может занимать определённое время.

Обзор известных результатов. Первый результат по задаче $F2 \mid l_j \mid C_{\max}$ — это классический алгоритм Джонсона [6] с временно́й сложностью $O(n \log n)$ для решения задачи с нулевыми минимальными задержками. В [8] исследована задача на одной машине, в которой каждая работа имеет две операции, разделённые по времени минимальной задержкой. Следуя расширению [13] трёхместной системы обозначений, введённой в [5], обозначим эту задачу как $1 \mid l_j \mid C_{\max}$. В [12, 13] показано, что эта задача эквивалентна задаче $F2 \mid l_j \mid C_{\max}$. В [8] показано, что задача $1 \mid l_j \mid C_{\max}$ NP-трудна в обычном смысле. Затем последовательно было установлена NP-трудность в сильном смысле для следующих задач: $F2 \mid l_j \mid C_{\max}$ [9], $F2 \mid l_j, a_j = b_j \mid C_{\max}$ [4], $F2 \mid l_j \in \{0, l\}, a_j = b_j \mid C_{\max}$ [12]. В [13] доказано, что задача $F2 \mid l_j, a_i = b_j = 1 \mid C_{\max}$ NP-трудна в сильном смысле. Для решения задачи $F2 \mid l_j \mid C_{\max}$ Делл Амико [3] предложил несколько алгоритмов с оценкой точности 2 и временно́й сложностью $O(n \log n)$, где n — число работ. Вопрос о существовании полиномиального алгоритма с лучшей оценкой точности формулировался в ряде работ (см., например, [11], но до сих пор остаётся открытым. Также остаётся открытым вопрос, является ли задача $F2 \mid l_j \mid C_{\max}$ APX-трудной.

Двухпроцессорная задача с минимальными задержками тесно связана с двухпроцессорной задачей с жёсткими задержками, которая отличается от первой требованием, что вторая операция каждой работы $j \in J$ должна начинаться точно через l_j единиц времени после окончания выполнения первой операции. В [1, 2] эта задача обозначается через $F2 \mid \text{exact } l_j \mid C_{\max}$. В [2], в частности, показано, что задача $F2 \mid \text{exact } l_j \mid C_{\max}$ может быть решена с оценкой точности 3, а также установлено, что её частный случай $F2 \mid \text{exact } l_j, a_j = b_j \mid C_{\max}$ может быть решён алгоритмом с оценкой точности 2, который с той же оценкой точности решает задачу $F2 \mid l_j, a_j = b_j \mid C_{\max}$. В [2] также показано, что для задачи $F2 \mid \text{exact } l_j, a_j = b_j \mid C_{\max}$ не существует приближённого алгоритма с оценкой точности $1{,}5 - \varepsilon$ при условии, что P≠NP. В [1] установлено, что задача $F2 \mid \text{exact } l_j, a_j = b_j = 1 \mid C_{\max}$ допускает приближённое решение с оценкой точности $\frac{3}{2}$. Так как при доказатель-

стве этого результата использована нижняя оценка, которая справедлива также и для задачи $F2 \mid l_j, a_j = b_j = 1 \mid C_{\max}$, предлагаемый алгоритм обеспечивает приближённое решение и этой задачи с той же оценкой точности $\frac{3}{2}$. Однако данный алгоритм не допускает обобщения даже на случай $a_j \equiv b_j \equiv a$.

Результат данной статьи. Можно показать, что задача $F2 \mid l_j \mid C_{\max}$ может быть решена простым алгоритмом с оценкой точности α, где

$$\alpha = 1 + \frac{\min\{\sum\limits_j a_j, \sum\limits_j b_j\}}{\max\{\sum\limits_j a_j, \sum\limits_j b_j\}}.$$

Заметим, что $\alpha = 2$, если $\sum\limits_j a_j = \sum\limits_j b_j$. Следовательно, этот подход не приводит к улучшению результата Делл Амико даже в пропорциональном случае, т. е. для задачи $F2 \mid l_j, a_j = b_j \mid C_{\max}$, которая является предметом исследования в этой статье. Основной результат данной статьи состоит в том, что задача $F2 \mid l_j, a_j = b_j \mid C_{\max}$ может быть решена с оценкой точности $\frac{3}{2}$, что является существенным улучшением результата Делл Амико для этой задачи. Алгоритм достаточно прост и может быть реализован с временно́й сложностью $O(n^2)$.

1. Определения, обозначения и полезные замечания

Введём основные обозначения и определения, а также сформулируем полезные замечания и предположения, не ограничивающие общность. В задаче $F2 \mid l_j \mid C_{\max}$ задано множество работ $J = \{1, \ldots, n\}$. Каждая работа $j \in J$ состоит из операций $O_{1,j}$ и $O_{2,j}$, длительности которых обозначаются через a_j и b_j соответственно. Мы предполагаем, что a_j и b_j — положительные целые числа при всех $j \in J$. Для каждой работы $j \in J$ операция $O_{2,j}$ отделена от операции $O_{1,j}$ временно́й задержкой в l_j единиц времени. Для каждой работы $j \in J$ обозначим через $\sigma(1,j)$ и $\sigma(2,j)$ начальные моменты операций $O_{1,j}$ и $O_{2,j}$ соответственно. В некоторых случаях работы $j \in J$ мы будем обозначать тройками (a_j, l_j, b_j).

Заметим, что расписание σ допустимо тогда и только тогда, когда

$$\sigma(2,j) \geqslant \sigma(1,j) + a_j + l_j$$

для всех $j \in J$. Для расписания $\sigma = (\sigma(1,1), \sigma(2,1), \ldots, \sigma(2,n))$ обозначим через $C_j(\sigma)$ момент завершения работы j в расписании σ; тогда $C_j(\sigma) = \sigma(2,j) + b_j$. Через $C_{\max}(\sigma) = \max\limits_{j \in J} C_j(\sigma)$ обозначается длина расписания σ, а через $C^*_{\max}$ — длина кратчайшего расписания.

Положим $A = \sum\limits_{j\in J} a_j$, $B = \sum\limits_{j\in J} b_j$ и $L = \max\limits_{j\in J} l_j$. При рассмотрении задачи $F2 \mid l_j, a_j = b_j \mid C_{\max}$ мы будем считать, что для каждой работы $j \in J$ длительности обеих операций равны a_j, следовательно, $A = \sum\limits_{j\in J} a_j = \sum\limits_{j\in J} b_j$.

Пусть σ — расписание задачи $F2 \mid l_j \mid C_{\max}$. Мы говорим, что работа j является *критической* в σ, если $\sigma(2,j) = \sigma(1,j) + a_j + l_j$. Заметим, что если расписание σ не имеет критических работ, т. е. $\sigma(2,j) > \sigma(1,j) + a_j + l_j$ для всех работ $j \in J$, то полагая $\sigma'(j) := \sigma(j)$ и $\sigma'(2,j) := \sigma(2,j) - \min\limits_{j\in J}\{\sigma(2,j) - \sigma(1,j) - a_j - l_j\}$, мы получаем другое допустимое расписание σ' с длиной

$$C_{\max}(\sigma') = C_{\max}(\sigma) - \min_{j\in J}\{\sigma(2,j) - \sigma(1,j) - a_j - l_j\} < C_{\max}(\sigma).$$

Таким образом, любое оптимальное расписание имеет хотя бы одну критическую работу.

Любое допустимое расписание σ порождает пару перестановок (φ, ψ) множества работ J таких, что φ определяет порядок выполнения работ на машине 1, а ψ — на машине 2. Более точно, $\varphi(k)$ ($\psi(k)$) — k-я работа, выполняемая на машине 1 (машине 2). Заметим, что $\varphi^{-1}(j)$ ($\psi^{-1}(j)$), где $j \in J$, есть порядковый номер, под которым работа j выполняется на машине 1 (машине 2). В дальнейшем перестановки φ и ψ будут представляться в виде последовательностей $(\varphi(1), \ldots, \varphi(n))$ и $(\psi(1), \ldots, \psi(n))$. Очевидно, что любое допустимое расписание σ может быть преобразовано в допустимое расписание $\overline{\sigma}$ с $C_{\max}(\sigma) \geqslant C_{\max}(\overline{\sigma})$, причём в расписании $\overline{\sigma}$ работы на обеих машинах выполняются непрерывно в том же порядке, что в расписании σ. Такие расписания мы будем называть *непрерывными*. По определению если σ — непрерывное расписание, заданное перестановками (φ, ψ), то для всякого $k = 2, \ldots, n$

$$\sigma(1, \varphi(k)) = \sigma(1, \varphi(k-1)) + a_{\varphi(k-1)},$$
$$\sigma(2, \psi(k)) = \sigma(2, \psi(k-1)) + b_{\psi(k-1)}.$$

Непрерывное расписание с парой перестановок (φ, ψ) изображено на рис. 1. Непрерывное расписание, в котором машина 1 начинает работу в момент времени 0 и по крайней мере одна работа является критической, будем называть *ранним* расписанием.

Заметим, что для заданной пары перестановок (φ, ψ) раннее расписание однозначно определяется этой парой и имеет минимальную длину

среди всех допустимых расписаний, отвечающих этой паре. Это расписание мы будем обозначать через $[\varphi, \psi]$. Из сказанного следует, что множество всех ранних расписаний содержит оптимальное расписание.

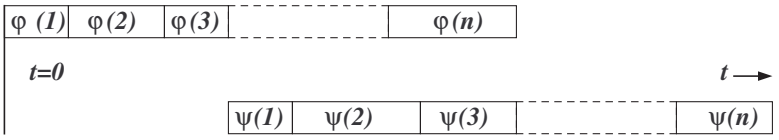

Рис. 1

Заметим, что при заданной паре перестановок (φ, ψ) раннее расписание $\sigma = [\varphi, \psi]$ может быть найдено за линейное время. В самом деле, для нахождения σ достаточно определить момент времени начала работы машины 2, т. е. $x = \sigma(2, \psi(1))$. Так как σ — допустимое непрерывное расписание, при любом $k = 1, \ldots, n$ имеем

$$\sigma(2, \psi(k)) = x + \sum_{i=1}^{k-1} b_{\psi(i)} \geqslant \sigma(1, \varphi(k)) + a_{\varphi(k)} + l_{\varphi(k)}.$$

Таким образом, $x = \min\limits_{1 \leqslant k \leqslant n} \{\sigma(1, \varphi(k)) + a_{\varphi(k)} + l_{\varphi(k)} - \sum_{i=1}^{k-1} b_{\psi(i)}\}$, что может быть найдено за линейное время.

2. Нижняя граница

Прежде чем перейти к описанию алгоритма, установим нижнюю границу для $C^*_{\max}$, которая будет играть решающую роль в доказательстве оценки точности алгоритма. Мы выведем её из нижней границы для задачи $F2 \mid l_j, a_j = b_j = 1 \mid C_{\max}$, доказанной в [12, 13] (лемма 2 в [13]). Мы приводим эту лемму вместе с доказательством.

Лемма 1 [5]. *Для любой конкретной задачи задачи* $F2 \mid l_j, a_j = b_j = 1 \mid C_{\max}$ *справедливо неравенство*

$$C^*_{\max} \geqslant n + 1 + \left\lceil \frac{1}{n} \sum_{j \in J} l_j \right\rceil. \tag{1}$$

Доказательство. Пусть σ — допустимое расписание с перестановками φ и ψ. Тогда для любой работы $j \in J$

$$C_{\max}(\sigma) \geqslant \sigma(1, j) + a_j + l_j + \sum_{k=\psi^{-1}(j)}^{n} b_{\psi(k)}$$

$$= \sum_{k=1}^{\varphi^{-1}(j)} a_{\varphi(k)} + l_j + \sum_{k=\psi^{-1}(j)}^{n} b_{\psi(k)}.$$

Поскольку $a_j = b_j \equiv 1$, имеем

$$\begin{aligned} C_{\max}(\sigma) &\geqslant \frac{1}{n}\Big(\sum_{j\in J}\Big(\sum_{k=1}^{\varphi^{-1}(j)} a_{\varphi(k)} + l_j + \sum_{k=\psi^{-1}(j)}^{n} b_{\psi(k)}\Big)\Big) \\ &= \frac{1}{n}\Big(\sum_{j\in J}\sum_{k=1}^{\varphi^{-1}(j)} a_{\varphi(k)} + \sum_{j\in J}\sum_{k=\psi^{-1}(j)}^{n} b_{\psi(k)}\Big) + \frac{1}{n}\sum_{j\in J} l_j \\ &= \frac{1}{n}\Big(\sum_{j\in J}\varphi^{-1}(j) + \sum_{j\in J}(n - \psi^{-1}(j) + 1)\Big) + \frac{1}{n}\sum_{j\in J} l_j \\ &= \frac{1}{2n}\Big(n(n+1) + n(n+1)\Big) + \frac{1}{n}\sum_{j\in J} l_j = n + 1 + \frac{1}{n}\sum_{j\in J} l_j. \end{aligned}$$

Лемма 1 доказана.

Легко видеть, что рассуждение в доказательстве леммы не может быть обобщено на случай $a_j \equiv b_j$. Тем не менее следующий результат показывает, что данная лемма может быть обобщена и на задачу $F2 \mid l_j, a_j = b_j \mid C_{\max}$.

Лемма 2. *Для любой конкретной задачи задачи* $F2 \mid l_j, a_j = b_j \mid C_{\max}$ *и* $J' \subseteq J$ *справедливо неравенство*

$$C^*_{\max} \geqslant \sum_{j\in J'} a_j + \left\lceil \frac{\sum\limits_{j\in J'} a_j^2}{\sum\limits_{j\in J'} a_j} + \frac{\sum\limits_{j\in J'} a_j l_j}{\sum\limits_{j\in J'} a_j} \right\rceil. \tag{2}$$

Доказательство. В самом деле, поскольку любая нижняя граница для конкретной задачи с уменьшенным числом работ $J' \subseteq J$ является нижней границей для первоначальной конкретной задачи, достаточно доказать лемму лишь для случая $J' = J$, т. е. показать, что

$$\begin{aligned} C^*_{\max} \geqslant A + 1 + \frac{1}{A}\sum_{j\in J} a_j^2 - 1 + \frac{1}{A}\sum_{j\in J} a_j l_j \\ = A + 1 + \frac{1}{A}\sum_{j\in J} a_j(a_j + l_j - 1). \end{aligned}$$

Пусть $\mathcal{I}$ — конкретная задача задачи $F2 \mid l_j, a_j = b_j \mid C_{\max}$. Рассмотрим конкретную задачу $\mathcal{I}_1$ задачи $F2 \mid l_j, a_j = b_j = 1 \mid C_{\max}$, которая получается из $\mathcal{I}$ следующим преобразованием: каждую работу $j \in J$ заменяем a_j идентичными работами $\{\sum\limits_{i=1}^{j-1} a_i + 1, \ldots, \sum\limits_{i=1}^{j-1} a_i + a_j\}$ с параметрами $(1, a_j + l_j - 1, 1)$. Таким образом, множество работ J_1 в $\mathcal{I}_1$ есть

$$J_1 = \bigcup_{j \in J} \Big\{ \sum_{i=1}^{j-1} a_i + 1, \ldots, \sum_{i=1}^{j-1} a_i + a_j \Big\} = \{1, \ldots, A\}.$$

Пусть σ — раннее расписание задачи $\mathcal{I}$ с перестановками φ и ψ. Построим расписание τ задачи $\mathcal{I}_1$, положив

$$\tau(1, i) = \sigma(1, j) + i - \sum_{k=1}^{j-1} a_k - 1, \quad \tau(2, i) = \sigma(2, j) + i - \sum_{k=1}^{j-1} a_k - 1$$

для всех $i \in \{\sum\limits_{k=1}^{j-1} a_k + 1, \ldots, \sum\limits_{k=1}^{j} a_k\}$ и $j \in J$. На рис. 2 изображён пример расписания σ из трёх работ $(4, 4, 4)$, $(3, 7, 3)$ и $(2, 0, 2)$ и расписания τ из 9 работ с параметрами $(1, 7, 1)$, $(1, 9, 1)$ и $(1, 0, 1)$, полученного из расписания σ. По построению длины расписаний σ и τ совпадают. Отсюда следует, что длина кратчайшего расписания в $\mathcal{I}$ не меньше длины кратчайшего расписания в $\mathcal{I}_1$. Теперь заметим, что по построению $\mathcal{I}_1$ — конкретная задача задачи $F2 \mid l_j, a_j = b_j \mid C_{\max}$; применяя к ней лемму 1, имеем $A + 1 + \frac{1}{A} \sum_{j \in J} a_j(a_j + l_j - 1)$, что и требовалось доказать. Лемма 2 доказана.

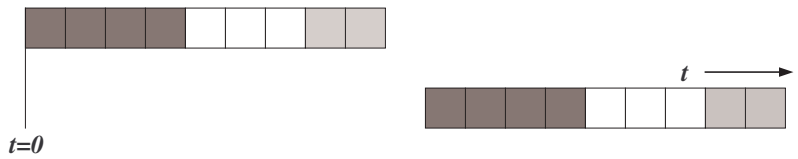

Рис. 2. Пример расписания σ и расписания τ, полученного из σ

3. Описание алгоритма

В этом разделе представим описание нашего алгоритма для решения задачи $F2 \mid l_j, a_j = b_j \mid C_{\max}$. Напомним, что этот случай задачи остаётся NP-трудным в сильном смысле, даже если $l_j \in \{0, l\}$ или $a_j = b_j = 1$ для всех $j \in J$ [12, 13]. Перейдём непосредственно к описанию алгоритма.

Алгоритм `Min_Delay`

Шаг 0. Упорядочим работы в порядке неубывания величин $a_j + l_j$. В целях упрощения записи в дальнейшем мы будем предполагать, что

$$a_1 + l_1 \leqslant a_2 + l_2 \leqslant \ldots \leqslant a_n + l_n. \tag{3}$$

Шаг k ($k = 1, \ldots, n$). Построим расписание $\sigma_k = [\varphi_k, \psi]$, где $\psi = (1, 2, \ldots, n)$, и $\varphi_k = (k{+}1, \ldots, n, 1, \ldots, k)$, если $k \leqslant n-1$, и $\varphi_n = (1, \ldots, n)$. (Расписание σ_k изображено на рис. 3.)

Результат работы алгоритма — расписание σ, имеющее кратчайшую длину среди всех расписаний в множестве $\{\sigma_1, \ldots, \sigma_n\}$.

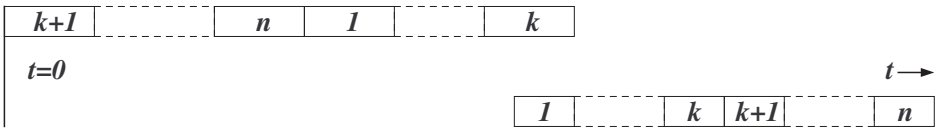

Рис. 3

Заметим, что шаг 0 может быть выполнен за время $O(n \log n)$. Как показано в разделе 2, раннее расписание σ_k для каждого $k = 1, \ldots, n$ может быть построено за линейное время. Таким образом, временна́я сложность алгоритма не превосходит $O(n^2)$.

4. Оценка точности

Начнём с полезного наблюдения, которое будет использоваться при выводе оценок длин расписаний σ_k, построенных алгоритмом `Min_Delay`.

Лемма 3. *Пусть работы множества J занумерованы согласно* (3). *Пусть σ — допустимое непрерывное* (*не обязательно раннее*) *расписание с перестановками $\varphi = (1, 2, \ldots, n)$ и $\psi = (1, 2, \ldots, n)$. Если некоторая работа $j \in J$ является критической в расписании σ, то работа n также является критической.*

Доказательство. Пусть $j \in J$ — некоторая работа. Тогда согласно определению расписания σ

$$C_{\max}(\sigma) \geqslant \sigma(1,1) + \sum_{i=1}^{j} a_i + l_j + \sum_{i=j}^{n} a_j = \sigma(1,1) + a_j + l_j + A$$

и неравенство превращается в равенство тогда и только тогда, когда j — критическая работа в расписании σ. С учётом (3) отсюда следует, что

$$C_{\max}(\sigma) \geqslant \sigma(1,1) + a_n + l_n + A.$$

Таким образом, если некоторая работа j является критической в расписании σ, то $a_j + l_j = a_n + l_n$ и работа n также оказывается критической в расписании σ. Лемма 3 доказана.

Следующая лемма устанавливает важную формулу, выражающую длину расписания σ_k как функцию номера k и входных данных конкретной задачи.

Лемма 4. *При любом* $k = 1, \ldots, n$,

$$C_{\max}(\sigma_k) = \max\{X_k, Y_k\},$$

где $X_k = \sum\limits_{j=k+1}^{n} a_j + a_n + l_n$ *и* $Y_k = A + \sum\limits_{j=k}^{n} a_j + l_k$.

Доказательство. Сначала рассмотрим случай $k = n$. Так как σ_n — раннее расписание, некоторая работа $j \in J$ является критической в σ_n. По лемме 3 отсюда следует, что работа n является критической в расписании σ_n. Тогда по построению расписания σ_n имеем

$$C_{\max}(\sigma_n) = \sum_{j=1}^{n} a_j + a_n + l_n = Y_n.$$

Пусть теперь $1 \leqslant k \leqslant n-1$. Напомним, что

$$\sigma_k = [(k+1, \ldots, n, 1, \ldots, k), (1, \ldots, k, k+1, \ldots, n)].$$

Вследствие допустимости расписания σ_k справедливо неравенство

$$\sigma_k(1,k) + a_k + l_k \leqslant \sigma_k(2,k).$$

Так как $\sigma_k(1,k) + a_k = \sum\limits_{j=1}^{n} a_j$ и $C_{\max}(\sigma_k) = \sigma_k(2,k) + \sum\limits_{j=k}^{n} a_k$, то отсюда следует, что

$$C_{\max}(\sigma_k) \geqslant \sigma_k(1,k) + a_k + l_k + \sum_{j=k}^{n} a_k = A + \sum_{j=k}^{n} a_k + l_k = Y_k.$$

Аналогичным образом показывается, что $C_{\max}(\sigma_k) \geqslant X_k$. Таким образом, имеем

$$C_{\max}(\sigma_k) \geqslant \max\{X_k, Y_k\}. \tag{4}$$

Заметим, что расписание σ_k состоит из двух непрерывных расписаний: расписания τ' множества работ $\{1, \ldots, k\}$ с перестановками $(1, \ldots, k)$,

$(1,\dots,k)$ и расписания τ'' множества работ $\{k+1,\dots,n\}$ с перестановками $(k+1,\dots,n)$, $(k+1,\dots,n)$. Поскольку σ_k — раннее расписание, по крайней мере одно из расписаний τ' и τ'' имеет критическую работу. Так как оба расписания τ' и τ'' удовлетворяют (3), по лемме 3 либо k, либо n — критическая работа в τ' или τ'' соответственно. Следовательно, по крайней мере одна из этих работ является критической в σ_k. Если работа k — критическая в σ_k, то

$$C_{\max}(\sigma_k) = \sigma_k(1,k) + a_k + l_k + \sum_{j=k}^{n} a_k = A + \sum_{j=k}^{n} a_k + l_k = Y_k.$$

Если работа n — критическая в σ_k, то аналогичные выкладки показывают, что $C_{\max}(\sigma_k) = X_k$. Таким образом, мы приходим к неравенству $C_{\max}(\sigma_k) \leqslant \max\{X_k, Y_k\}$, которое вместе с (4) доказывает лемму 4.

Теорема. *Пусть I — конкретная задача рассматриваемой задачи и σ — расписание, полученное алгоритмом* `Min_Delay`. *Тогда*

$$C_{\max}(\sigma) \leqslant \frac{3}{2} C^*_{\max}. \tag{5}$$

Доказательство. Для $k = 1,\dots,n$ положим $\theta(k) = X_k - Y_k$. По лемме 4 имеем $\theta(k) = a_n + l_n - a_k - l_k - A$. Так как $a_k + l_k$ — неубывающая функция по k, то $\theta(k)$ — невозрастающая функция по k. Заметим, что $\theta(n) = -A < 0$. Таким образом, возможны следующие два случая: либо $Y_k \geqslant X_k$ для всех $k = 1,\dots,n$, либо существует номер $r \in \{1,\dots,n-1\}$ такой, что $Y_r < X_r$ и $Y_k \geqslant X_k$ для всех $k = r+1,\dots,n$.

Случай 1. $Y_k \geqslant X_k$ при любом $k = 1,\dots,n$. Тогда по лемме 4 имеем $C_{\max}(\sigma_k) = Y_k$ для всех $k = 1,\dots,n$. По построению σ отсюда следует, что

$$\begin{aligned} C_{\max}(\sigma) &\leqslant \frac{1}{A}\sum_{k=1}^{n} a_k C_{\max}(\sigma_k) = \frac{1}{A}\sum_{k=1}^{n} a_k Y_k = \frac{1}{A}\sum_{k=1}^{n} a_k (A + \sum_{j=k}^{n} a_j + l_k) \\ &= A + \frac{1}{A}\left(\sum_{k=1}^{n}\sum_{j=k}^{n} a_k a_j + \sum_{k=1}^{n} a_k l_k\right) \\ &= A + \frac{1}{2A}\Big(\Big(\sum_{i=1}^{n} a_i\Big)^2 + \sum_{i=1}^{n} a_i^2\Big) + \frac{1}{A}\sum_{k=1}^{n} a_k l_k \\ &= \Big(\frac{A}{2} - \frac{1}{2A}\sum_{i=1}^{n} a_i^2\Big) + \Big(A + \frac{1}{A}\sum_{i=1}^{n} a_i^2 + \frac{1}{A}\sum_{k=1}^{n} a_k l_k\Big), \end{aligned}$$

что по лемме 2 не превосходит $\dfrac{C^*_{\max}}{2} + C^*_{\max} \leqslant \dfrac{3}{2} C^*_{\max}$.

Случай 2. Существует номер $r \in \{1, \ldots, n-1\}$ такой, что $Y_r < X_r$ и $Y_k \geqslant X_k$ при всех $k = r+1, \ldots, n$. По лемме 4 отсюда следует, что $C_{\max}(\sigma_k) = Y_k$ при любом $k = r+1, \ldots, n$ и $C_{\max}(\sigma_k) = X_k$ для всех $k = 1, \ldots, r$. Так как X_k — невозрастающая функция k, то по построению σ имеем

$$C_{\max}(\sigma) \leqslant \min\left\{ \frac{\sum\limits_{k=r+1}^{n} a_k C_{\max}(\sigma_k)}{\sum\limits_{i=r+1}^{n} a_i}, C_{\max}(\sigma_r) \right\} \leqslant \min\{S_1, S_2\}, \qquad (6)$$

где

$$S_1 = \frac{\sum\limits_{k=r+1}^{n} a_k Y_k}{\sum\limits_{i=r+1}^{n} a_i} = \frac{\sum\limits_{k=r+1}^{n} a_k (A + \sum\limits_{j=k}^{n} a_j + l_k)}{\sum\limits_{i=r+1}^{n} a_i}, \qquad (7)$$

$$S_2 = X_r = \sum_{j=r+1}^{n} a_j + a_n + l_n. \qquad (8)$$

Положим $A_r = \sum\limits_{i=r+1}^{n} a_i$. Используя тождество $2 \sum\limits_{k=r+1}^{n} \sum\limits_{j=k}^{n} a_k a_j = A_r^2 + \sum\limits_{i=r+1}^{n} a_i^2$, перегруппируем выражение в правой части из (7) следующим образом:

$$\begin{aligned} S_1 &= A + \frac{1}{A_r}\Big(\sum_{k=r+1}^{n} \sum_{i=r+1}^{n} a_k a_j\Big) + \frac{1}{A_r} \sum_{k=r+1}^{n} a_k l_k \\ &= A + \frac{1}{2A_r}\Big(A_r^2 + \sum_{i=r+1}^{n} a_i^2\Big) + \frac{1}{A_r} \sum_{k=r+1}^{n} a_k l_k \\ &= A - \frac{1}{2A_r} \sum_{i=r+1}^{n} a_i^2 + \frac{A_r}{2} + \frac{1}{A_r} \sum_{i=r+1}^{n} a_i^2 + \frac{1}{A_r} \sum_{k=r+1}^{n} a_k l_k \\ &\leqslant A + \frac{A_r}{2} + \frac{1}{A_r} \sum_{i=r+1}^{n} a_i^2 + \frac{1}{A_r} \sum_{k=r+1}^{n} a_k l_k \\ &= A + \beta\mu + \frac{1}{2}(A_r + \beta) + \beta\Big(\frac{1}{2} - \mu\Big), \end{aligned} \qquad (9)$$

где $\beta = \dfrac{1}{A_r}\sum\limits_{i=r+1}^{n} a_i^2 + \dfrac{1}{A_r}\sum\limits_{j=k}^{n} a_j l_k$ и $\mu = \dfrac{A_r}{A}$.

По лемме 2 справедливо неравенство $A_r + \beta \leqslant C^*_{\max}$ и

$$A + \beta\mu = A + \frac{1}{A}\sum_{i=r+1}^{n} a_i^2 + \frac{1}{A}\sum_{j=k}^{n} a_j l_k \leqslant A + \frac{1}{A}\sum_{i=1}^{n} a_i^2 + \frac{1}{A}\sum_{j=1}^{n} a_j l_k \leqslant C^*_{\max}.$$

Таким образом, из (9) следует, что

$$S_1 \leqslant C_{\max} + \frac{1}{2}C_{\max} + \beta\left(\frac{1}{2} - \mu\right). \tag{10}$$

С другой стороны, с использованием тривиальной нижней границы

$$C^*_{\max} \geqslant \max_{j\in J}\{2a_j + l_j\}$$

из (8) получаем

$$S_2 = 2a_n + l_n + \sum_{j=r+1}^{n-1} a_j \leqslant C^*_{\max}(1+\mu). \tag{11}$$

Теперь сравнивая (10) и (11), видим, что если $\mu \geqslant \frac{1}{2}$, то $S_1 \leqslant \frac{3}{2}C_{\max}$, а если $\mu < \frac{1}{2}$, то $S_2 < \frac{3}{2}C_{\max}$. С учётом (6) это завершает доказательство теоремы.

ЛИТЕРАТУРА

1. **Ageev A. A., Baburin A. E.** Approximation algorithms for UET scheduling problems with exact delays // Oper. Res. Letters 2007. V. 35, N 4. P. 533–540.
2. **Ageev A. A., Kononov A. V.** Approximation algorithms for scheduling problems with exact delays // Approximation and online algorithms. 4th International workshop, WAOA 2006 (Zurich, September 14–15, 2006). Revised papers. Berlin: Springer, 2006. P. 1–14. (Lecture Notes in Comput. Sci.; V. 4368).
3. **Dell'Amico M.** Shop problems with two machines and time lags // Oper. Research 1996. V. 44, N 4. P. 777–787.
4. **Dell'Amico M., Vaessens R. J. M.** Flow and open shop scheduling on two machines with transportation times and machine-independent processing times is NP-hard // Materiali di discussione 141, Dipartimento di Economia Politica, Università di Modena, 1996.

5. **Graham R. L., Lawler E. L., Lenstra J. K., Rinnooy Kan A. H. G.** Optimization and approximation in deterministic sequencing and scheduling: a survey // Annals of Discrete Mathematics. 1979. V. 5. P. 287–326.

6. **Johnson S. M.** Optimal two- and three-stage production schedules with setup times included // Naval Research Logistics Quarterly. 1954. V. 1. P. 61–68.

7. **Johnson S. M.** Discussion: Sequencing n jobs on two machines with arbitrary time lags // Management Science. 1958. V. 5. P. 299–303.

8. **Kern W., Nawijn W. M.** Scheduling multi-operation jobs with time lags on a single machine // Proc. of the 2nd Workshop on Graphs and Combinatorial Optimization. Enschede, 1991.

9. **Lenstra J. K.** Частное сообщение, 1991.

10. **Mitten L. G.** Sequencing n jobs on two machines with arbitrary time lags // Management Science. 1958. V. 5. P. 293–298.

11. **Strusevich V. A.** A heuristic for the two-machine open-shop scheduling with transportation times // Discrete Applied Math. 1999. V. 93, N 2–3. P. 287–304.

12. **Yu W.** The two-machine shop problem with delays and the one-machine total tardiness problem // Ph.D. thesis, Department of Mathematics and Computer Science, Technische Universiteit Eindhoven, 1996.

13. **Yu W., Hoogeveen H., Lenstra J. K.** Minimizing makespan in a two-machine flow shop with delays and unit-time operations is NP-hard // J. Sched. 2004. V. 7, N 5. P. 333–348.

Адрес автора:

Институт математики
им. С. Л. Соболева СО РАН,
пр. Академика Коптюга, 4,
630090 Новосибирск,
Россия.
E-mail: ageev@math.nsc.ru

Статья поступила
24 июля 2007 г.