Автоматика и телемеханика, № 10, 2010

© 2010 г. **Ф. ВЕРНЕР, д-р философии**
(Факультет математики университета Отто фон Герике,
Магдебург, Германия),
С. А. КРАВЧЕНКО, канд. физ.-мат. наук
(Объединенный институт проблем информатики НАН Беларуси, Минск)

ПОСТРОЕНИЕ ОПТИМАЛЬНЫХ РАСПИСАНИЙ ДЛЯ ОБСЛУЖИВАЮЩИХ СИСТЕМ С МНОЖЕСТВОМ СЕРВЕРОВ

Рассматривается задача оптимального обслуживания множества требований на множестве идентичных параллельных приборов. Перед выполнением необходима загрузка, которая осуществляется сервером. Причем задано множество одинаковых серверов, каждый из которых годится для выполнения загрузки. Рассматриваются вопросы вычислительной сложности данной задачи при условии, что необходимо минимизировать время завершения обслуживания всех требований. Для случая с одинаковыми длительностями обслуживания и одинаковыми длительностями загрузки предлагается полиномиальный алгоритм. Для задачи с единичными временами загрузки при условии, что количество приборов на единицу превосходит количество серверов, предлагается псевдополиномиальный алгоритм. Доказывается NP-трудность в сильном смысле для задачи с фиксированным числом приборов и серверов при условии минимизации максимальной задержки. Обобщаются некоторые известные алгоритмы с фиксированными временами обслуживания на соответствующие задачи с сервером при фиксированных временах загрузки. Кроме того, сделана оценка эффективности для двух списочных алгоритмов при условии минимизации общего времени обслуживания.

1. Введение

В работе рассматривается задача оптимального обслуживания n требований на множестве, состоящем из m параллельных идентичных приборов. Каждое требование должно пройти обслуживание на одном из приборов. Прерывания в процессе обслуживания запрещаются. Перед началом обслуживания требование необходимо загрузить на прибор при помощи сервера (робота). Загрузка может быть выполнена только в случае, если имеются свободный сервер и свободный прибор. Если в текущий момент все приборы и все серверы заняты обслуживанием и загрузкой, то загрузка нового требования не может быть начата. Таким образом, для того чтобы требование начало загружаться, необходимо наличие свободного сервера и прибора. Предполагается, что время перевода сервера от одного прибора к другому пренебрежимо мало. В общем случае в системе имеется k серверов для выполнения требуемой загрузки.

В ряде работ рассматривались обслуживающие модели с одним сервером. Анализ вычислительной сложности описанных задач с одним сервером и классическими критериями оптимальности рассматривался в [1]. В данной работе проводится анализ вычислительной сложности описанной задачи. В зависимости от критерия оптимальности, количества приборов и ограничений на времена выполнения и времена загрузки либо предлагается эффективный алгоритм решения, либо доказывается NP-трудность.

Эвристический алгоритм поиска по лучу (beam search) для задачи оптимального обслуживания на параллельных приборах с одним сервером был предложен

в [2]. В [3] предложен алгоритм опережающего просмотра (look-ahead heuristic) для случая непрерывно прибывающих требований. В [4] рассматривались задачи на параллельных приборах с одним сервером при условии минимизации быстродействия и при условии минимизации вынужденных простоев. Был предложен псевдополиномиальный алгоритм для случая с двумя приборами и единичными временами загрузки. Кроме того, анализировались некоторые полиномиально разрешимые случаи и эвристические алгоритмы. В [5] исследовался ряд вопросов, связанных с вычислительной сложностью задачи с одним сервером. В частности, доказано, что задача минимизации суммы моментов завершения обслуживания требований с единичными длительностями загрузки является NP-трудной в сильном смысле. В [6] можно найти достаточно полный обзор результатов, полученных для задач теории расписаний с учетом загрузок.

Задачи оптимального обслуживания на параллельных приборах с разгрузочными серверами рассматривались в [7]. В [8] рассматривалась задача оптимального обслуживания с одним сервером на множестве параллельных предписанных (dedicated) приборов и последовательностно-зависимых времен загрузки, т.е. время загрузки для данного требования определяется предшествующим требованием, обслуженным на том же приборе.

Некоторые задачи, рассматриваемые в литературе, являются родственными задачам с сервером. Например, задача помех на приборах (the machine interference problem) состоит в определении оптимального количества приборов, которые должны быть назначены для выполнения одной операции таким образом, чтобы минимизировать операционные помехи, возникающие при попытке обслуживания несколькими приборами одного требования [9].

Существует несколько работ, посвященных серверам и автоматизированным объектам в гибких производственных системах. В [10] исследовали вычислительную сложность задач, возникающих в роботизированных гибких производствах. Некоторые результаты по вычислительной сложности задач, возникающих при построении оптимальных расписаний для роботизированных гибких участков, можно найти в [11–13]. Для более детального ознакомления можно обратиться к [14, 15].

Другой подобной задачей является задача, при которой выполнение требований возможно лишь при наличии нескольких ресурсов. В [16] рассматривается задача, в которой необходимо минимизировать сумму взвешенных моментов завершения обслуживания требований. В [17] рассматривается задача с двумя идентичными приборами и одним сервером, в которой учитывается время движения сервера между приборами.

В данной работе обобщаются некоторые результаты из [1], при этом рассматривается обслуживающая система с k серверами, $k \geqslant 1$. Статья организована следующим образом. Формулировка задачи и используемые обозначения приводятся в разделе 2. В разделах 3 и 5 исследуются в основном задачи построения оптимальных по быстродействию расписаний. В частности, в разделе 3 исследуются вопросы вычислительной сложности, а в разделе 5 анализируются две эвристики построения списочных расписаний. В разделе 4 известные алгоритмы для задач с параллельными приборами и одинаковыми временами обслуживания требований обобщаются на случай соответствующих задач с серверами при условии одинаковых времен загрузки. В разделе 6 предлагается задача для дальнейших исследований.

2. Формулировка задачи

Пусть имеется множество $N = \{1, \ldots, n\}$ требований, которые необходимо обслужить на множестве $m > 1$ идентичных параллельных приборов, далее обозначенных через $M_1, \ldots, M_m$. Для каждого требования $j \in N$ известно время обслуживания на приборе p_j и время загрузки на прибор s_j. Обслуживание может выполняться на

любом из приборов, и до начала обслуживания необходима предварительная загрузка требования на прибор. Любой прибор одновременно может выполнять не более одного требования. Загрузка и выполнение требования не могут прерываться. Далее предполагается, что $s_j \geqslant 1$ и $p_j \geqslant 0$. Отметим, что здесь нулевое время выполнения интерпретируется как пренебрежимо малая величина. В зависимости от типа рассматриваемой задачи в постановке могут присутствовать также неотрицательный директивный срок d_j, срок завершения D_j, моменты поступления r_j и веса w_j.

Для обозначения задач, рассматриваемых далее, воспользуемся символикой, предложенной в [18], при которой каждая задача описывается в виде $\alpha \mid \beta \mid \gamma$. В предложенной символике α служит для обозначения обслуживающей системы, β описывает характеристики требований, и γ указывает целевую функцию, которую необходимо минимизировать. В принятых обозначениях α включает символ вида Sk, который указывает на наличие k серверов, если же используется символ S, то это значит, что количество серверов в каждом конкретном примере задачи может быть произвольным. К примеру, $\alpha = Pm, S2$ означает, что рассматривается задача с двумя серверами и m параллельными приборами, при этом значение m является фиксированным. В качестве параметра β также можно брать случай с единичными длительностями $p_j = 1$ и единичными временами загрузки $s_j = 1$. Если же времена загрузки (длительности) равны между собой, то это обозначается как $s_j = s$ ($p_j = p$). Если множество $\beta = \emptyset$, то времена загрузки и длительности могут принимать произвольные значения.

В качестве параметра γ будем рассматривать следующие критерии оптимальности. Если $\gamma = C_{\max}$, то необходимо минимизировать время выполнения расписания, т.е. минимизируется $C_{\max} = \max\limits_{j=1,\ldots,n}\{C_j\}$, при этом C_j обозначает время завершения обслуживания требования $j \in N$. Критерий $L_{\max} = \max\limits_{j=1,\ldots,n}\{C_j - d_j\}$ обозначает минимизацию максимальной задержки. Если $\gamma = \sum w_j C_j$, то необходимо минимизировать взвешенную сумму моментов завершения всех требований. Если на месте γ стоит прочерк, т.е. $\gamma = -$, то это значит, что необходимо построить допустимое расписание.

Одним из наиболее быстрых алгоритмов для задач на параллельных приборах является алгоритм упорядочения по списку. В таком алгоритме требования распределяются равномерно по приборам согласно списку. Полученное списочное расписание полностью определяется перестановкой требований $\pi = (\pi_1, \ldots, \pi_n)$, задающей последовательность, в которой сервер производит загрузку. Другими словами, взяв i-е требование π_i, определяем свободный прибор M_j, на который требование π_i должно быть загружено по возможности раньше.

3. Вычислительная сложность задач с несколькими серверами

В [19] было показано, что задача $P2, S1 \mid s_j = 1 \mid C_{\max}$ является NP-трудной в обычном смысле. Несложно доказать аналогичный результат для любого фиксированного числа приборов и серверов.

Исследуем вычислительную сложность задачи $Pm, Sk \mid s_j = 1 \mid C_{\max}$, где множество требований необходимо обслужить на m приборах с k серверами. Заметим, что число требований не является фиксированным, т.е. оно может изменяться в каждом конкретном примере. Загрузка каждого требования занимает единицу времени. Необходимо минимизировать длину построенного расписания.

Теорема 1. Задача $Pm, Sk \mid s_j = 1 \mid C_{\max}$ является NP-трудной в обычном смысле.

Доказательство. Поскольку задача $P2 \mid\mid C_{\max}$ является NP-трудной в обычном смысле, будем рассматривать случай $m > 2$. Рассмотрим задачу $Pm, Sk \mid s_j =$

$= 1 \mid C_{\max}$ и предположим, что $m-2 = ak+b$ для некоторых положительных целых a и $0 \leqslant b < k$. Известно, что обе задачи $P2, S2 \mid s_j = 1 \mid C_{\max}$ и $P2, S1 \mid s_j = 1 \mid C_{\max}$ являются NP-трудными в обычном смысле, см. [19]. Значит, следующая задача распознавания является NP-полной: существует ли решение для задачи $P2, S2 \mid s_j = = 1 \mid C_{\max}$ (или задачи $P2, S1 \mid s_j = 1 \mid C_{\max}$ соответственно), при котором для заданного значения C выполняется $C_{\max} \leqslant C$?

Понятно, что названные задачи распознавания остаются NP-полными при допущении, что сумма всех времен выполнения и времен загрузок является нечетным числом (которое возникает, если все единичные промежутки времени, за исключением первого единичного интервала, до момента времени C на обоих приборах заняты загрузкой и выполнением). Теперь, если $k = 1$, задача $P2, S1 \mid s_j = 1 \mid C_{\max}$ может быть сведена к задаче $Pm, Sk \mid s_j = 1 \mid C_{\max}$ добавлением $m-2$ требований с $p_1 = C, p_2 = C+1, \ldots, p_{m-2} = C+m-3$. Однако, если $k > 1$, задача $P2, S2 \mid s_j = = 1 \mid C_{\max}$ может быть сведена к задаче $Pm, Sk \mid s_j = 1 \mid C_{\max}$ добавлением k требований с $p_j = C+a-1$, k требований с $p_j = C+a-2, \ldots, k$ требований с $p_j = C$ и b требований с $p_j = C-1$.

Теперь покажем, что задача $Pm, S(m-1) \mid s_j = 1 \mid C_{\max}$ может быть решена для случая с фиксированным числом приборов за псевдополиномиальное время. Этот результат является обобщением результата, опубликованного в [4] для задачи $P2, S1 \mid s_j = 1 \mid C_{\max}$.

Известно, что задача $Pm \mid\mid C_{\max}$ может быть решена псевдополиномиальным алгоритмом даже в случае, когда все приборы становятся доступными в разное время. Далее будем обозначать этот алгоритм через PP. Применим алгоритм PP к некому частичному расписанию σ и некому множеству требований $N' \subset N$, другими словами, начиная с некого частичного расписания σ решаем задачу на параллельных приборах без серверов, пытаясь при этом обслужить все требования из множества N' наиболее быстрым способом, т.е. так, чтобы длина построенного расписания была минимальной.

Нижняя граница оптимального значения целевой функции для задачи $Pm, S(m- - 1) \mid s_j = 1 \mid C_{\max}$ получается из неравенства $C_{\max} \geqslant \max\{n, \tilde{C}_{\max}\}$, где $\tilde{C}_{\max}$ обозначает оптимальное значение целевой функции для задачи $Pm \mid\mid C_{\max}$, при условии, что один прибор становится доступным в момент времени 1, а все остальные приборы доступны с момента времени 0, причем времена выполнения n требований равны $p'_j = p_j + 1$, где p_j обозначает время выполнения требования $j \in N$ в задаче $Pm, S(m-1) \mid s_j = 1 \mid C_{\max}$.

Будем помечать требование $j \in N$ как допустимое для некого частичного расписания σ, если после обслуживания требования j не все m приборов завершают работу одновременно.

Перед первым шагом алгоритма все требования готовы к обслуживанию, и частичное расписание σ_0 описывается ситуацией, когда первые $m-1$ приборов готовы к работе в момент 0, а последний прибор может начинать обслуживание в момент 1. В алгоритме n шагов. На каждом шаге i некое новое требование обрабатывается и полученное расписание σ_i рассматривается как частичное расписание для задачи на параллельных приборах без серверов с модифицированными временами выполнения $p'_j = p_j + 1$ всех требований.

Алгоритм 1.

Шаг i, $i = 1, \ldots, n$:

а. Выбираем наименьшее требование (т.е. требование с минимальным модифицированным временем выполнения) такое, что после добавления этого требования к частичному расписанию σ_{i-1} и применения алгоритма PP к полученному частичному расписанию σ_i и ко всем необслуженным требованиям с модифици-

рованными временами обслуживания $p'_j = p_j + 1$ получающееся быстродействие не превосходит значения $C_{\max} = \max\{n, \tilde{C}_{\max}\}$.

В случае невозможности такого выбора берем произвольное требование.

b. Получаем новое расписание σ_i из расписания σ_{i-1}, исключаем выбранное требование из дальнейшего рассмотрения.

Временная сложность алгоритма 1 оценивается величиной $O\left(n^3\left(\sum\limits_{i=1}^{n} p'_i\right)^{m-1}\right)$. Чтобы узнать значение $\tilde{C}_{\max}$, необходимо использовать алгоритм PP, сложность которого $O\left(n\left(\sum\limits_{i=1}^{n} p'_i\right)^{m-1}\right)$. В п. **a.** шага i необходимо рассмотреть не более чем n требований и применить алгоритм PP к каждому из полученных расписаний. Описанная процедура потребует $O\left(n\left(\sum\limits_{i=1}^{n} p'_i\right)^{m-1}\right)$ шагов. Следовательно, временная сложность п. **a.** шага i оценивается значением $O\left(n^2\left(\sum\limits_{i=1}^{n} p'_i\right)^{m-1}\right)$. Поскольку алгоритм 1 состоит из n шагов, общая сложность составит $O\left(n^3\left(\sum\limits_{i=1}^{n} p'_i\right)^{m-1}\right)$.

Теорема 2. Алгоритм 1 строит оптимальное расписание для задачи $Pm, S(m-1) \mid s_j = 1 \mid C_{\max}$.

Доказательство. Если на подшаге **a.** каждого шага i алгоритма 1 можно выбрать допустимое требование, то выполняется $C_{\max} = \tilde{C}_{\max}$. Предположим, что на некотором шаге i не существует допустимого требования, т.е. для частичного расписания σ_{i-1} один прибор доступен с момента времени T_1, а остальные приборы доступны с момента T_2 и все необработанные требования имеют модифицированные времена выполнения $T_2 - T_1$.

Если количество необработанных требований не превосходит m, то алгоритм строит оптимальное расписание, поскольку в этом случае выполняется $C_{\max} = \tilde{C}_{\max}$.

Предположим, что количество необработанных требований превосходит m. Рассмотрим интервал $[T_1, T_2]$ и соответствующее расписание σ_{i-1}. Заметим, что все требования, выполняющиеся в интервале $[T_1, T_2]$, имеют модифицированные времена выполнения, которые не меньше значения $T_2 - T_1$. Выберем из них требование с максимальным значением p'_i и рассмотрим шаг j, на котором это требование было назначено на обработку. Поскольку на этом шаге алгоритм делал выбор между требованием i и требованием с модифицированным временем обработки $T_2 - T_1$, можно заключить, что для расписания σ_{j-1} один прибор доступен с момента времени T'_1, другие приборы доступны с момента T'_2, кроме того, выполняется $T'_2 - T'_1 = T_2 - T_1$.

И опять в интервале $[T'_1, T'_2]$ выполняются только требования с модифицированным временем, превосходящим значение $T'_2 - T'_1$. Приведенные рассуждения можно повторять до момента, когда будет рассматриваться частичное расписание σ_j, для которого выполняется $T'_1 = 0$. По определению расписания σ_0 один прибор доступен с момента 1. Следовательно, описанная ситуация возможна лишь тогда, когда выполняется $T_2 - T_1 = 1$.

Покажем теперь, что в частичном расписании σ_{i-1} все серверы работают без простоев. Предположим, что в интервале $[T'_1, T'_1 + 1]$ работает меньше чем $m-1$ серверов. Тогда существуют, по крайней мере, два требования, которые выполняются в $[T'_1, T'_1 + 1]$ и при этом их загрузка завершается до момента T'_1. Выберем любое из этих требований, скажем z, и рассмотрим шаг q алгоритма 1, на котором выбранное

требование назначается на обработку. После построения расписания σ_{i-1} количество единичных требований превосходит m, следовательно, на шаге q алгоритма 1 должно выбираться не требование z, а некое единичное требование. Итак, данную ситуацию можно исключить из рассмотрения, и, следовательно, в частичном расписании σ_{i-1} все серверы работают без простоев. Значит, в этом случае алгоритм 1 строит расписание с $C_{\max} = n$.

Алгоритм 1 можно применить к задаче $P, S \mid s_j = 1 \mid C_{\max}$, т.е. к случаю с произвольным числом приборов и серверов. Однако полученное расписание может оказаться не оптимальным. Рассмотрим следующий пример задачи $P3, S1 \mid s_j = 1 \mid C_{\max}$. Имеется девять требований с длительностями обработки $p_1 = p_2 = p_3 = 1$, $p_4 = p_5 = p_6 = p_7 = 2$, $p_8 = 3$, $p_9 = 4$. Оптимальное расписание длины $C_{\max} = 10$ можно получить применением списочного алгоритма к последовательности $\pi = (4, 5, 6, 9, 1, 2, 8, 7, 3)$. Алгоритм 1 располагает требования в порядке $\pi' = (4, 5, 6, 7, 8, 9, 1, 2, 3)$. Заметим, что перед загрузкой требования 3 возникает простой. Поэтому полученное значение $C_{\max}$ будет равно 11.

Вопрос о существовании псевдополиномиального алгоритма для задачи $Pm, Sk \mid s_j = 1 \mid C_{\max}$ со значением $k \leqslant m-2$ остается открытым. Однако следующий результат показывает, что для критерия $L_{\max}$ задача с фиксированным числом приборов и серверов не может быть решена за псевдополиномиальное время, если $P \neq NP$.

Теорема 3. Задача $Pm, Sk \mid s_j = 1 \mid L_{\max}$ унарно NP-трудна для любого фиксированного числа приборов m и любого фиксированного числа серверов k при $k < m$.

Доказательство. В [19] было показано, что следующая задача распознавания для модели $P2, S1 \mid s_j = 1 \mid L_{\max}$ является NP-трудной в сильном смысле:

$$
\begin{aligned}
&n = r(2y+1) + 1,\\
&p_j = 2a_j, \quad d_j = r(2y+1), \quad j = 1, \ldots, 3r,\\
&p_j = 1, \quad d_j = k(2y+1) + 1, \quad j = 3r + k, \quad k = 1, \ldots, r,\\
&p_j = 0, \quad d_j = k(2y+1), \quad j = 4r + (k-1)(2y-3) + 1, \ldots, 4r + k(2y-3), \quad k = 1, \ldots, r,\\
&p_j = 0, \quad d_j = r(2y+1) + 1, \quad j = r(2y+1) + 1,
\end{aligned}
$$

где все $3r$ элементов $a_1, \ldots, a_{3r}$ таковы, что $\sum\limits_{i=1}^{3r} a_i = ry$ и выполняется $\dfrac{y}{4} < a_i < \dfrac{y}{2}$ для $i = 1, \ldots, 3r$. Необходимо решить, возможно ли построить расписание, для которого справедливо неравенство $L_{\max} \leqslant 0$.

Для описанной задачи распознавания $P2, S1 \mid s_j = 1 \mid L_{\max}$ рассмотрим следующую задачу распознавания $Pm, Sk \mid s_j = 1 \mid L_{\max}$.

Пусть $m = k + 1 + ak + b$ и справедливо $0 \leqslant b < k$. Тогда для каждого требования j задачи распознавания $P2, S1 \mid s_j = 1 \mid L_{\max}$ положим $p'_j = p_j$, $d'_j = d_j + a + 1$, $j = 1, \ldots, r(2y+1) + 1$. Дополним задачу следующими требованиями.

Добавим ak требований со значениями параметров:

$$
\begin{aligned}
&k \text{ требований с } p'_i = a + r(2y+1) + 1,\ d'_i = a + r(2y+1) + 2,\\
&k \text{ требований с } p'_i = a + r(2y+1),\ d'_i = a + r(2y+1) + 2,\\
&\ldots\\
&k \text{ требований с } p'_j = 2 + r(2y+1),\ d'_i = a + r(2y+1) + 2.
\end{aligned}
$$

Далее добавим b требований с $p'_j = 1 + r(2y+1)$ и $d'_j = a + r(2y+1) + 2$. Добавим также $k - b$ требований с $p'_i = 0$, $d'_i = a + 1$ и $(k-1)(r(2y+1)+1)$ требований со

значениями параметров:

$$(k-1)(2y+1) \text{ требований с } p'_i = 0,\ d'_i = (2y+1) + a + 1,$$
$$(k-1)(2y+1) \text{ требований с } p'_i = 0,\ d'_i = 2(2y+1) + a + 1,$$
$$\dots$$
$$(k-1)(2y+1) \text{ требований с } p'_i = 0,\ d'_i = r(2y+1) + a + 1,$$
$$(k-1) \text{ требований с } p'_i = 0,\ d'_i = r(2y+1) + a + 2.$$

Покажем, что существует решение задачи распознавания $Pm, Sk \mid s_j = 1 \mid L_{\max}$ тогда и только тогда, когда существует решение задачи распознавания $P2, S1 \mid s_j = 1 \mid L_{\max}$.

($\Leftarrow$) Предположим, что существует решение задачи распознавания $P2, S1 \mid s_j = 1 \mid L_{\max}$. Применим списочный алгоритм ко всем добавленным требованиям начиная с требования $j = r(2y+1)+2)$, используя приборы $M_3, M_4, \dots, M_m$ и взяв порядок, заданный в тексте. Получаем задачу оптимального обслуживания первых $r(2y+1)+1$ требований на двух приборах в интервале $[a+1, a+2+r(2y+1)]$, которая может быть решена аналогично задаче $P2, S1 \mid s_j = 1 \mid L_{\max}$.

($\Rightarrow$) Предположим, что существует решение для задачи распознавания $Pm, Sk \mid s_j = 1 \mid L_{\max}$. Тогда первые $ak+k$ дополнительных требований должны выполняться на $m-k$ приборах, чтобы не нарушить директивные сроки. Остальные требования должны выполняться на $k+1$ приборах в интервале времени $[a+1, a+2+r(2y+1)]$.

Отметим, что в каждом интервале $[a+1+(z-1)(2y+1), a+1+z(2y+1)]$, $z = 1, \dots, r$, серверы должны работать без простоев. Следовательно, сумма всех времен выполнения в каждом интервале равна $2y+1$. Кроме того, требования при выполнении не могут накладываться друг на друга, поскольку в этом случае для одного из требований будет нарушен директивный срок. Поскольку точно три требования с ненулевыми длительностями должны выполняться в каждом интервале, их всегда можно выполнить на двух приборах. Таким образом будет построено решение для $P2, S1 \mid s_j = 1 \mid L_{\max}$.

Если число приборов является произвольным, то задача $P, S1 \mid s_j = 1 \mid C_{\max}$ является унарно NP-трудной [4]. Из этого факта непосредственно получается

Теорема 4. Задача $P, Sk \mid s_j = 1 \mid C_{\max}$ является унарно NP-трудной.

Доказательство. Рассмотрим задачу 3-разбиения: дано множество целых чисел $A = \{a_1, \dots, a_{3m}\}$ с условиями $\sum\limits_{i=1}^{3m} a_i = mB$, $\dfrac{B}{4} < a_i < \dfrac{B}{2}$ для $1 \leqslant i \leqslant 3m$. Существует ли разбиение множества A на m попарно не пересекающихся 3-элементных множеств $A_1, \dots, A_m$ таких, что для каждого j выполняется равенство $\sum\limits_{a_i \in A_j} a_i = B$?

Данный пример задачи о 3-разбиении можно полиномиально свести к следующему варианту задачи $P, Sk \mid s_j = 1 \mid C_{\max}$ с $3m$ требованиями и m приборами, когда $p_i = ma_i - 1$, $s_i = 1$, $i = 1, \dots, 3k$: можно ли построить расписание, для которого выполняется $C_{\max} \leqslant mB + z - 1$?

Если времена загрузки не единичные, но равные между собой, можно сформулировать следующий результат [19]: задача $P2, S1 \mid s_j = s \mid C_{\max}$ является унарно NP-трудной. Из этого результата можно получить теорему.

Теорема 5. Задача $Pm, Sk \mid\mid C_{\max}$ является унарно NP-трудной для каждого $k < m$.

Доказательство. В [19] было доказано, что задача $P2, S1 \mid s_j = s \mid C_{\max}$ является унарно NP-трудной, т.е. следующая задача распознавания является NP-полной: существует ли решение для задачи $P2, S1 \mid s_j = s \mid C_{\max}$, при котором выполняется $C_{\max} \leqslant C$? Рассмотрим задачу $Pm, Sk \mid\mid C_{\max}$ и предположим,

что $m = k + 1 + ak + b$, $0 \leqslant b < k$. Пусть $d = b$, если $b \neq 0$, и $d = k$, если $b = 0$. Пусть $D = C$, если $b = 0$, и $D = C + 1$, если $b \neq 0$.

Задачу $P2, S1 \mid s_j = s \mid C_{\max}$ можно свести к задаче $Pm, Sk \mid\mid C_{\max}$, добавив к условию задачи $P2, S1 \mid s_j = s \mid C_{\max}$ множество требований, которые можно описать следующим образом: k требований с $s_j = 1$, $p_j = D + 1$; k требований с $s_j = 1$, $p_j = D + 2$; $\ldots$; k требований с $s_j = 1$, $p_j = D + a$; b требований с $s_j = 1$, $p_j = D + 1$; $k - d$ требований с $s_j = D + 1$, $p_j = 0$ и $d - 1$ требований с $s_j = D$, $p_j = 0$.

Теперь обратимся к задаче с единичными длительностями выполнения требований. В [19] показано, что задача $P, S1 \mid p_j = 1 \mid C_{\max}$ полиномиально разрешима. Если количество серверов больше одного, то задача усложняется.

Теорема 6. Задача $P, Sk \mid p_j = 1 \mid C_{\max}$ является бинарно NP-трудной для каждого фиксированного значения $k > 1$.

Доказательство. Положим $m = n$, где n означает число требований. Задача сводится к делению множества из n требований на k подмножеств таким образом, чтобы максимальная сумма времен загрузки для каждого множества была как можно больше. Таким образом, получается известная задача упаковки (bin packing problem) [20].

4. Полиномиальный алгоритм для случая с одинаковыми значениями длительностей и одинаковыми значениями загрузок

В [21] были разработаны полиномиальные алгоритмы для задач $P \mid r_j, p_j = p, D_j \mid -$, $P \mid r_j, p_j = p, D_j \mid \sum C_j$ и $P \mid r_j, p_j = p \mid \sum w_j C_j$. В [22] был предложен полиномиальный алгоритм для задачи $P \mid r_j, p_j = p \mid \sum f_j(C_j)$, где f_j – неубывающая функция такая, что для любых индексов i и j функция $f_i - f_j$ является монотонной. В [23] был предложен полиномиальный алгоритм для задачи $P \mid r_j, p_j = p, D_j \mid \max \varphi_j(C_j)$, где φ_j является неубывающей функцией.

В этом разделе покажем, что полиномиальные алгоритмы, разработанные в [21–23], можно обобщить на задачи $P, S \mid r_j, p_j = p, s_j = s, D_j \mid -$, $P, S \mid r_j, p_j = p, s_j = s, D_j \mid \sum C_j$, $P, S \mid r_j, p_j = p, s_j = s \mid \sum w_j C_j$ $P, S \mid r_j, p_j = p, s_j = s \mid \sum f_j(C_j)$, $P, S \mid r_j, p_j = p, s_j = s, D_j \mid \max \varphi_j(C_j)$ соответственно.

Так как все обобщения делаются аналогичным образом, приведем обобщение только для задачи $P, S \mid r_j, p_j = p, s_j = s, D_j \mid -$.

Утверждение. Оптимальное расписание для задачи $P, S \mid r_j, p_j = p, s_j = s, D_j \mid F$, где $F = F(C_1, \ldots, C_n)$ – любая неубывающая функция, можно искать в классе расписаний, для которых каждое требование выполняется в одном из следующих интервалов длины p :

$$\{[r_j + vp + ws, r_j + vp + ws + p + s[\mid v, w \in \{0, 1, 2, \ldots\}\}.$$

Доказательство. Предположим утверждение неверно. Выберем требование, которое начинается раньше всех требований, интервалы выполнения которых не принадлежат множеству $\{[r_j + vp + ws, r_j + vp + ws + p + s[\mid v, w \in \{0, 1, 2, \ldots\}\}$. Обозначим это требование через k и момент начала его обработки через t_k. Возможны три случая:

1) либо некое требование l завершается в момент t_k;
2) либо существует требование l, чья загрузка завершается в момент t_k;
3) либо нет требований, завершающих обслуживание или загрузку в момент t_k.

В случаях 1 и 2 получаем, что требование k не является требованием с самым ранним началом выполнения. В случае 3 получаем, что расписание не является оптимальным. Таким образом, утверждение верно.

Положим $D = \max_j\{D_j\}$ и $r = \min_j\{r_j\}$. Напомним, что m означает число приборов. Обозначим через e число серверов.

Рассмотрим следующее множество интервалов:

$$I = \{[r_j + vp + ws, r_j + vp + ws + p + s \; [\mid j \in \{1, 2, \ldots\};$$
$$v, w \in \{\ldots, -2, -1, 0, 1, 2, \ldots\};\; r_j + vp + ws \geqslant r;\; r_j + vp + ws + p + s \leqslant D\}.$$

Перенумеруем различные интервалы из множества I по порядку их расположения на числовой оси. Обозначим полученное множество через I', т.е.

$$I' = \{I_i \mid i \in \{1, \ldots, z\}\},$$

и для каждого I_i обозначим через $r(I_i)$ левый конец интервала I_i, а через $d(I_i)$ правый конец интервала I_i. Выберем максимальное значение y такое, что $I_{i+1} \bigcap \ldots \bigcap I_{i+y} \neq \emptyset$ для любых $i \in \{0, \ldots, z - y\}$. Пусть h будет максимальным индексом таким, что выполняется $I_{i+1} \bigcap \ldots \bigcap I_{i+h} \neq \emptyset$ для любых $i \in \{0, \ldots, z - h\}$ и $r(I_{i+h}) - r(I_{i+1}) \leqslant s$. Кроме того, пусть x_{ji} равно количеству требования j, выполняемых в интервале I_i.

Рассмотрим систему:

$$(1) \qquad \sum_{i=1}^{z} x_{ji} = p, \quad j = 1, \ldots, n,$$

$$(2) \qquad \sum_{j=1}^{n} x_{j,i+1} + \sum_{j=1}^{n} x_{j,i+2} + \ldots + \sum_{j=1}^{n} x_{j,i+y} \leqslant mp, \quad i = 0, \ldots, z - y,$$

$$(3) \qquad \sum_{j=1}^{n} x_{j,i+1} + \sum_{j=1}^{n} x_{j,i+2} + \ldots + \sum_{j=1}^{n} x_{j,i+h} \leqslant ep, \quad i = 0, \ldots, z - h,$$

$$(4) \qquad x_{ji} = 0, \quad \text{если} \quad r(I_i) < r_j \quad \text{или} \quad d(I_i) > D_j \quad i = 1, \ldots, z, \quad j = 1, \ldots, n,$$

$$(5) \qquad x_{ji} \geqslant 0, \quad i = 1, \ldots, z, \quad j = 1, \ldots, n.$$

Покажем, что задача (1)–(5) эквивалентна задаче $P, S \mid r_j, p_j = p, s_j = s, D_j \mid -$.

Теорема 7. Любое допустимое расписание для задачи $P, S \mid r_j, p_j = p, s_j = s, D_j \mid -$ можно представить как допустимое решение системы (1)–(5).

Доказательство. Любое допустимое расписание можно описать как вектор x со значениями x_{ji}, равными количеству требования j, выполняющихся в интервале I_i. Заметим, что для x ограничение (1) справедливо, поскольку $\sum\limits_{i=1}^{z} x_{ji}$ является суммарным временем выполнения требования j. Неравенство (2) справедливо, поскольку общее число приборов равно m. Ограничение (3) выполняется, так как общее количество серверов равно e. Ограничение (4) выполняется, поскольку требование j может выполняться только в интервале $[r_j, D_j]$. Ограничение (5) справедливо по определению x.

Теорема 8. Любое допустимое решение системы (1)–(5) *можно преобразовать в допустимое расписание для задачи $P, S \mid r_j, p_j = p, s_j = s, D_j \mid -$.*

Доказательство. Пусть x^* обозначает решение системы (1)–(5). Из [21] следует, что x^* можно рассматривать, как решение для системы (1), (2), (4) и (5). Его можно преобразовать в расписание $\tilde{x}$, в котором все требования выполняются в рамках заданных директивных сроков, каждое требование выполняется лишь одним прибором и каждый прибор выполняет одновременно лишь одно требование.

Остается доказать только, что $\tilde{x}$ удовлетворяет ограничению (3), т.е. что

$$\sum_{j=1}^{n} \tilde{x}_{j,i+1} + \sum_{j=1}^{n} \tilde{x}_{j,i+2} + \ldots + \sum_{j=1}^{n} \tilde{x}_{j,i+h} \leqslant ep \quad \text{для любых} \quad i = 0, \ldots, z-h.$$

Докажем это, используя лемму 1 из [21]. В лемме 1 сказано, что если $e(I_i)$ обозначает количество помеченных копий интервала I_i и

$$v(I_i) = \sum_{j=1}^{n} x^*_{j1} + \ldots + \sum_{j=1}^{n} x^*_{ji},$$

то

$$(e(I_1) + \ldots + e(I_i))p \geqslant v(I_i) > (e(I_1) + \ldots + e(I_i) - 1)p.$$

Итак, получаем

$$(e(I_1) + \ldots + e(I_{k+h}))p \geqslant v(I_{k+h}) > (e(I_1) + \ldots + e(I_{k+h}) - 1)p,$$
$$(e(I_1) + \ldots + e(I_k))p \geqslant v(I_k) > (e(I_1) + \ldots + e(I_k) - 1)p,$$

и, следовательно, выполняется

$$v(I_{k+h}) - v(I_k) > (e(I_1) + \ldots + e(I_{k+h}) - 1)p - (e(I_1) + \ldots + e(I_k))p,$$

т.е.

$$v(I_{k+h}) - v(I_k) > (e(I_{k+1}) + \ldots + e(I_{k+h}) - 1)p.$$

Поскольку

$$v(I_{k+h}) - v(I_k) = \sum_{j=1}^{n} x^*_{j,k+1} + \ldots + \sum_{j=1}^{n} x^*_{j,k+h},$$

получаем

$$\sum_{j=1}^{n} x^*_{j,k+1} + \ldots + \sum_{j=1}^{n} x^*_{j,k+h} > (e(I_{k+1}) + \ldots + e(I_{k+h}) - 1)p.$$

Учитывая

$$(e(I_{k+1}) + \ldots + e(I_{k+h}))p = \sum_{j=1}^{n} \tilde{x}_{j,k+1} + \ldots + \sum_{j=1}^{n} \tilde{x}_{j,k+h},$$

получаем

$$\sum_{j=1}^{n} x^*_{j,k+1} + \ldots + \sum_{j=1}^{n} x^*_{j,k+h} > \sum_{j=1}^{n} \tilde{x}_{j,k+1} + \ldots + \sum_{j=1}^{n} \tilde{x}_{j,k+h} - p.$$

Далее, поскольку из (3) следует справедливость

$$\sum_{j=1}^{n} x^*_{j,i+1} + \sum_{j=1}^{n} x^*_{j,i+2} + \ldots + \sum_{j=1}^{n} x^*_{j,i+h} \leqslant ep,$$

получаем

$$ep > \sum_{j=1}^{n} \tilde{x}_{j,k+1} + \ldots + \sum_{j=1}^{n} \tilde{x}_{j,k+h} - p,$$

другими словами,

$$ep + p > \sum_{j=1}^{n} \tilde{x}_{j,k+1} + \ldots + \sum_{j=1}^{n} \tilde{x}_{j,k+h},$$

и поскольку $\tilde{x}_{j,i} \in \{0, p\}$, получаем, что справедливо

$$ep \geqslant \sum_{j=1}^{n} \tilde{x}_{j,k+1} + \ldots + \sum_{j=1}^{n} \tilde{x}_{j,k+h}.$$

Таким образом, $\tilde{x}$ удовлетворяет ограничению (3).

Итак, доказано, что задача $P, S \mid r_j, p_j = p, s_j = s, D_j \mid -$ решается за полиномиальное время. Аналогично можно показать, что задача $P \mid r_j, p_j = p \mid \sum f_j(C_j)$, где f_j – неубывающая функция такая, что для любых индексов i и j функция $f_i - f_j$ является монотонной, как и задача $P \mid r_j, p_j = p, D_j \mid \max \varphi_j(C_j)$, где φ_j – неубывающая функция, могут быть решены за полиномиальное время.

5. Оценки погрешностей для списочных алгоритмов

Оценим погрешности вычисления точного решения двух списочных алгоритмов для задачи $P, Sk \mid\mid C_{\max}$, т.е. для случая, когда дано произвольное число параллельных приборов и k серверов.

Случай, когда $k = 1$, был рассмотрен в [1, 19]. В частности, в [19] для задачи $P, S1 \mid\mid C_{\max}$ приведены следующие результаты для алгоритма LPT (требования в списке расположены в порядке невозрастания их длительностей) и произвольного списочного алгоритма (требования в списке расположены в произвольном порядке):

(a) алгоритм LPT имеет погрешность $\dfrac{2-1}{m}$;

(b) произвольный списочный алгоритм имеет погрешность $\dfrac{3-2}{m}$.

В [4] было показано, что обе оценки являются точными даже для случая с единичными временами загрузки.

Рассмотрим случай $k \geqslant 2$.

Теорема 9. Для алгоритма LPT справедлива оценка:

$$\frac{C_{\max}^{LPT}}{C_{\max}^{*}} \leqslant 2 + \frac{k-1}{k} - \frac{k}{m},$$

где $C_{\max}^{LPT}$ обозначает длину расписания, построенного по алгоритму LPT, и $C_{\max}^{*}$ обозначает минимальную длину расписания.

Доказательство. Рассмотрим любое расписание, построенное алгоритмом LPT для задачи $P, Sk \mid\mid C_{\max}$. Для этого расписания время загрузки s_j для требования $j \in \{1, \ldots, n\}$ разбивается на k частей, скажем s_j^1, s_j^2, $\ldots$ и s_j^k, где s_j^r обозначает общее время, когда сервер выполняет загрузку требования j и r серверов работают

одновременно. Обозначим последнее выполняемое требование через a. Тогда справедлива оценка:

$$
\begin{aligned}
C_{\max}^{LPT} &\leqslant \left(\sum_j s_j^k\right) \Big/ k + \\
&+ \left(\sum_j p_j + \sum_j s_j - s_a^1 - s_a^2 - \ldots - s_a^{k-1} - \sum_j s_j^k - p_a\right) \Big/ m + \\
&+ s_a^1 + s_a^2 + \ldots + s_a^{k-1} + p_a = \\
&= \left(\sum_j p_j + \sum_j s_j\right) \Big/ m - \left(s_a^1 + s_a^2 + \ldots + s_a^{k-1} + \sum_j s_j^k + p_a\right) \Big/ m + \\
&+ \left(\sum_j s_j^k\right) \Big/ k + s_a^1 + s_a^2 + \ldots + s_a^{k-1} + p_a = \\
&= \left(\sum_j p_j + \sum_j s_j\right) \Big/ m - \left(s_a^1 + s_a^2 + \ldots + s_a^{k-1} + \sum_j s_j^k + p_a\right) \Big/ m + \\
&+ \left(\sum_j s_j^k/k + s_a^1/k + \ldots + s_a^{k-1}/k + p_a/k\right) + \\
&+ \left(s_a^1 + s_a^2 + \ldots + s_a^{k-1} + p_a\right) \frac{k-1}{k} = \\
&= \left(\sum_j p_j + \sum_j s_j\right) \Big/ m + \\
&+ \left(\sum_j s_j^k/k + s_a^1/k + \ldots + s_a^{k-1}/k + p_a/k\right) \left(1 - \frac{k}{m}\right) + \\
&+ \left(s_a^1 + s_a^2 + \ldots + s_a^{k-1} + p_a\right) \frac{k-1}{k}.
\end{aligned}
$$

Учитывая, что

$$
\left(\sum_j p_j + \sum_j s_j\right) \Big/ m \leqslant C_{\max}^*,
$$

$$
\sum_j s_j^k/k + s_a^1/k + \ldots + s_a^{k-1}/k + p_a/k \leqslant C_{\max}^*
$$

и

$$
s_a^1 + s_a^2 + \ldots + s_a^{k-1} + p_a \leqslant C_{\max}^*,
$$

получаем

$$
C_{\max}^{LPT} \leqslant C_{\max}^* \left(2 + \frac{k-1}{k} - \frac{k}{m}\right).
$$

Покажем, что данная оценка является точной. Рассмотрим следующий пример задачи $P, Sk \mid\mid C_{\max}$ с $(m-k)m + k(k-1) + 1$ требованиями, предполагая, что k является делителем m. Имеется $m(m-k)$ требований с $s_j = 0$ и $p_j = 1$, $k(k-1)$ требований с $s_j = m/k$ и $p_j = 0$ и одно требование с $s_j = m$ и $p_j = 0$. Алгоритмом LPT все требования упорядочиваются следующим образом: сначала все требования с $p_j = 1$, затем все требования с $s_j = m/k$ и затем требование с $s_j = m$. Длина полученного расписания равна $m - k + (k-1)m/k + m$, а длина оптимального расписания – m, что получается, если требование с $s_j = m$ выполнить первым, а затем k раз повторить следующую процедуру: выполнять $k-1$ требований с $s_j = m/k$ и $(m-k)m/k$ требований с $p_j = 1$.

Рассмотрим произвольный список требований.

Теорема 10. Для произвольной последовательности списочный алгоритм имеет оценку

$$\frac{C^{LS}_{\max}}{C^{*}_{\max}} \leqslant 3 - \frac{k+1}{m},$$

где $C^{LS}_{\max}$ означает длину расписания для произвольного списка и $C^{*}_{\max}$ означает оптимальную длину расписания.

Доказательство. Рассмотрим любое списочное расписание для задачи $P, Sk \mid\mid C_{\max}$. Используя обозначения из предыдущего доказательства, получим оценку:

$$C^{LS}_{\max} \leqslant \left(\sum_j s^k_j\right) \Big/ k + \left(\sum_j p_j + \sum_j s_j - \sum_j s^k_j - p_a\right) \Big/ m + p_a =$$

$$= \left(\sum_j p_j + \sum_j s_j\right) \Big/ m + (1 - k/m) \left(\sum_j s^k_j\right) \Big/ k + p_a \left(1 - \frac{1}{m}\right).$$

Учитывая, что

$$\left(\sum_j p_j + \sum_j s_j\right) \Big/ m \leqslant C^{*}_{\max}, \quad \sum_j s^k_j / k \leqslant C^{*}_{\max} \quad \text{и} \quad p_a \leqslant C^{*}_{\max},$$

получаем

$$C^{LS}_{\max} \leqslant C^{*}_{\max} \left(3 - \frac{k+1}{m}\right).$$

Покажем, что полученная оценка точная. Рассмотрим пример задачи $P, Sk \mid\mid C_{\max}$ с количеством требований $zm(m-k-1) + k + 1$, где z – произвольное положительное число. Имеется $zm(m-k-1)$ требований с $s_j = 0$ и $p_j = 1$, k требований с $s_j = zm - 1$ и $p_j = 0$ и одно требование с $s_j = 1$ и $p_j = zm - 1$. По произвольному списку можно обработать сначала требования с $p_j = 1$, затем все требования с $s_j = zm - 1$ и наконец требование с $s_j = 1$. В результате получим расписание длины $z(m-k-1) + zm - 1 + zm$, в то время как длина оптимального расписания zm. Оптимальное расписание получается при обработке требований с $s_j = 1$ в первую очередь, а затем k требований с $s_j = zm - 1$ и всех остальных требований.

Итак, получаем значение

$$\frac{C^{LS}_{\max}}{C^{*}_{\max}} = 3 - \left(k + 1 + \frac{1}{z}\right)/m,$$

которое стремится к $3 - \dfrac{k+1}{m}$, если $z \to \infty$.

6. Заключение

В данной работе использовались критерии $\max \varphi_j(C_j)$, где φ_j является неубывающей функцией, и $\sum f_j(C_j)$, где f_j является неубывающей функцией такой, что для любых индексов i и j функция $f_i - f_j$ является монотонной. Для дальнейших исследований было бы интересно исследовать вопрос о вычислительной сложности задачи $P, S \mid r_j, p_j = p, s_j = s \mid \sum U_j$, где $U_j = 0$, если $C_j \leqslant d_j$, и $U_j = 1$, если $C_j > d_j$.

СПИСОК ЛИТЕРАТУРЫ

1. *Hall N.G., Potts C.N., Sriskandarajah C.* Parallel machine scheduling with a common server // Discret. Appl. Math. 2000. V. 102. P. 223–243.
2. *Koulamas C.P.* Scheduling on two parallel machines for minimizing machine interference. Working Paper 1993.
3. *Koulamas C.P., Smith M.L.* Look-ahead scheduling for minimizing machine interference // Int. J. Product. Res. 1988. V. 26. P. 1523–1533.
4. *Kravchenko S.A., Werner F.* Parallel machine scheduling problems with a single server // Math. Comput. Model. 1997. V. 26. P. 1–11.
5. *Brucker P., Dhaenens-Flipo C., Knust S. et. al.* Complexity results for parallel machine problems with a single server // J. Schedul. 2002. V. 5. P. 429–457.
6. *Allahverdi A., Ng C.T., Cheng T.C.E., Kovalyov M.Y.* A survey of scheduling problems with setup times or costs // Eur. J. Oper. Res. 2008. V. 187. P. 985–1032.
7. *Ou J., Qi X., Lee C.-Y.* Parallel machine scheduling with multiple unloading servers // J. Scheduling. 2009. doi: 10.1007/s10951-009-0104-1.
8. *Huang S., Cai L., Zhang X.* Parallel dedicated machine scheduling problem with sequence-dependent setups and a single server // Comput. & Indust. Engin. 2009. doi: 10.1016/j.cie.2009.10.003.
9. *Aronson J.E.* Two heuristics for the deterministic, single operator, multiple machine, multiple run cyclic scheduling problem // J. Oper. Management. 1984. V. 4. P. 159–773.
10. *Wilhelm W.E., Sarin S.C.* A structure for sequencing robot activities in machine loading applications // Int. J. Product. Res. 1985. V. 23. P. 47–64.
11. *Hall N.G., Kamoun H., Sriskandarajah C.* Scheduling in robotic cells: classification, two and three machine cells // Oper. Res. 1997. V. 45. P. 421–439.
12. *Hall N.G., Kamoun H., Sriskandarajah C.* Scheduling in robotic cells, complexity and sready state analysis // Eur. J. Oper. Res. 1998. V. 109. P. 43–65.
13. *Kamoun H., Hall N.G., Sriskandarajah C.* Scheduling in robotic cells: heuristics and cell design // Oper. Res. 1999. V. 47. P. 821–835.
14. *Ganesharajah T., Hall N.G., Sriskandarajah C.* Design and operational issues in AGV-served manufacturing systems // Ann. Oper. Res. 1998. V. 76. P. 109–154.
15. *Sethi S.P., Sriskandarajah C., Sorger G., et. al.* Sequencing of parts and robot moves in a robotic cell // Int. J. Flexible Manufactur. Syst. 1992. V. 4. P. 331–358.
16. *Dobson G., Kamarkar U.S.* Simultaneous resource scheduling to minimize weighted flow time // Oper. Res. 1988. V. 37. P. 1523–1533.

17. *Sahney V.K.* Single-server, two-machine sequencing with switching time // Oper. Res. 1972. V. 20. P. 24–26.
18. *Graham R.L., Lawler E.L., Lenstra J.K., et. al.* Optimization and approximation in deterministic machine scheduling: a survey // Ann. Discret. Math. 1979. V. 5. P. 287–326.
19. *Hall N.G., Potts C.N., Sriskandarajah C.* Parallel machine scheduling with a common server // The Fifth Int. Workshop Project Management and Scheduling, Abstracts. 1997. P. 102–106.
20. *Tanaev V.S., Gordon V.S., Shafransky Y.M.* Scheduling Theory, Single-Stage Systems. Kluwer Academic Publisher, 1994.
21. *Brucker P., Kravchenko S.A.* Scheduling jobs with equal processing times and time windows on identical parallel machines // J. Schedul. 2008. V. 11. P. 229–237.
22. *Kravchenko S.A., Werner F.* On a parallel machine scheduling problem with equal processing times // Discret. Appl. Math. 2009. V. 157. P. 848–852.
23. *Kravchenko S.A., Werner F.* On a parallel machine scheduling problem with equal processing times // Preprint 26/07, Fakultät für Mathematik, Otto-von-Guericke-Universität Magdeburg, 2007.

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

Поступила в редакцию 12.01.2010