ДИСКРЕТНЫЙ АНАЛИЗ И ИССЛЕДОВАНИЕ ОПЕРАЦИЙ
Май—июнь 2009. Том 16, № 3. С. 74–98

УДК 519.854.2

О СВОЙСТВАХ ОПТИМАЛЬНЫХ РАСПИСАНИЙ В ЗАДАЧЕ FLOW SHOP С ПРЕРЫВАНИЯМИ И ПРОИЗВОЛЬНЫМ РЕГУЛЯРНЫМ КРИТЕРИЕМ *)

Д. А. Чемисова

Аннотация. Исследуются свойства оптимальных расписаний в NP-трудной задаче flow shop с разрешением прерываний операций на минимум произвольной регулярной функции от моментов окончания операций. Показано, что оптимальное расписание любого примера такой задачи может быть построено жадным алгоритмом при подходящем задании приоритетов операций на машинах. Дана оценка на число прерываний в любом жадном (в частности, в оптимальном) расписании. Доказано также, что длина оптимального (по заданному регулярному критерию) расписания всегда совпадает с суммой длин операций из некоторого подмножества. Результаты данной работы обобщают известные ранее результаты для задачи с классическим критерием на минимум длины расписания.

Ключевые слова: теория расписаний, прерывание, оптимальное расписание, регулярный критерий.

Введение

Задача flow shop является одной из первых цеховых задач теории расписаний, рассмотренных в литературе. В наиболее простом варианте — с критерием минимум длины расписания и без каких-либо ограничений на расписание кроме запрета на прерывание операций — она была рассмотрена Джонсоном ещё в 1954 г. [9] (с тех пор этот вариант задачи называется задачей Джонсона). Неудивительно, что данный вариант задачи flow shop является на настоящий момент наиболее исследованным. Однако и в этом варианте осталось немало открытых вопросов. Наиболее полный (хотя и несколько устаревший) обзор результатов по этой задаче можно найти в [4].

Но стоит только сделать небольшой (на первый взгляд) шаг в сторону от классической постановки задачи и снять запрет на прерывание

*) Исследование выполнено при финансовой поддержке Российского фонда фундаментальных исследований (проект 08–01–00370).

операций (в стандартной трёхпольной системе обозначений [4] эта задача записывается как $F|pmtn|C_{\max}$), как мы сталкиваемся с практически неизученной областью теории расписаний. Первые исследования этой задачи, выполненные Гонзалезом и Сани в 1978 г. [8], показали, что задача NP-трудна в сильном смысле уже в случае трёх машин. Этот факт был бы нисколько не удивителен (мало ли известно NP-трудных задач?), если бы вслед за этим отрицательным результатом появились (как это обычно бывает) и положительные результаты — какие-либо алгоритмы, точные или приближённые, с худшими или лучшими показателями эффективности и точности. Но дело в том, что за тридцать лет, прошедшие с момента первой публикации по этой задаче, не только не появилось каких-либо новых интересных результатов, но практически не было получено **вообще ничего**. До недавнего времени не было известно даже какого-либо элементарного (но конечного) алгоритма точного решения этой задачи.

Объяснением этого феномена может служить тот очевидный факт, что оптимальное расписание для такой задачи должно иметь заведомо более сложную структуру по сравнению с таковой для задачи без прерываний операций. В структуре решения этой задачи процесс выполнения каждой операции уже не обязательно представляется непрерывным интервалом времени, а может быть разбит на сколь угодно много (сколь угодно мелких) частей, не связанных во времени. И хотя в некоторых случаях разрешение прерывания операций не является препятствием для решения задачи*, в большинстве случаев ситуация обратная.

Лишь сравнительно недавно появились нетривиальные результаты по анализу свойств оптимальных расписаний задачи $F|pmtn|C_{\max}$. Так, в [2] обнаружены интересные структурные свойства оптимальных расписаний, такие как «суперцелочисленность оптимума». Оказалось, что несмотря на разрешение прерываний операций (могущее привести к появлению сколь угодно малых «дробных» частей операций), длина оптимального расписания всегда совпадает с суммой длин целых операций из некоторого подмножества.

Помимо исследования свойств оптимальных решений в [2] приведён алгоритм решения задачи $F|pmtn|C_{\max}$. Показано, что оптимальное решение этой задачи может быть получено обычным жадным алгоритмом при надлежащем задании приоритетов операций на каждой машине. И

*И даже более того, является облегчающим фактором. Например, известно, что задачи $R|pmtn|C_{\max}$ и $O|pmtn|C_{\max}$ с прерываниями операций на минимум длины расписания в системах с параллельными машинами и в системе open shop не только алгоритмически разрешимы, но и эффективно разрешимы (см. [10] и [7] соответственно), в то время как аналогичные задачи без прерывания операций NP-трудны.

хотя об эффективности этого алгоритма можно даже не говорить, но его конечность безусловно гарантирована.

Наконец, в [2] получена оценка сверху на минимальное число прерываний в оптимальном расписании, гарантированное для любого примера с заданным числом машин m и числом работ n. (Естественно, что эта оценка выражается в виде функции от m и n.) Приведённая оценка примерно в n раз улучшала аналогичную оценку из [3].

Упомянутые выше результаты получены для задачи flow shop с «наиболее удобным» для анализа критерием — минимум длины расписания. Что же касается задач с другими критериями (и с разрешением прерываний операций), то здесь ситуация с поиском эффективных алгоритмов, по-видимому, ещё более безнадёжна. Как показал анализ сложности, проведённый в работах [5] и [6], задачи $F2|pmtn|L_{\max}$ и $F2|pmtn|\sum C_j$ (на минимум *максимального временного смещения* и *суммы времён окончания работ*) являются NP-трудными в сильном смысле уже в случае двух машин. Однако аналогичная задача $F2|pmtn|C_{\max}$ на минимум длины расписания полиномиально разрешима [9].

Такая трудная ситуация, сложившаяся на фронте цеховых задач с прерываниями операций, послужила стимулом для написания настоящей работы. Представлялось интересным выяснить, насколько полученные в [2] результаты применимы к задачам flow shop с другими критериями оптимальности. В нашей работе рассматривается задача flow shop с произвольным *регулярным критерием* (на минимум произвольной *регулярной*, т. е. неубывающей по каждому из аргументов и непрерывной слева, *функции* от моментов завершения операций).

Следует отметить, что все классические целевые функции (такие как длина расписания, сумма времён окончания работ, суммарное запаздывание работ, суммарное взвешенное опоздание работ и др.) являются регулярными функциями от своих аргументов — моментов окончания работ. Если же учесть, что момент завершения каждой работы является регулярной функцией от моментов завершения её операций (и равняется их максимуму) и что суперпозиция регулярных функций вновь даёт регулярную функцию, то мы приходим к выводу, что все классические оптимизируемые характеристики расписаний (такие как длина расписания, сумма времён окончания работ и др.) представимы в виде регулярных функций от моментов завершения операций. В то же время далеко не всякая функция от моментов завершения операций представима в виде функции от моментов завершения работ. Таким образом, переходя от классических целевых функций к регулярным функциям от моментов

завершения операций, мы существенно расширяем область применимости наших результатов. О каких же результатах идёт речь?

Будет показано, что свойство суперцелочисленности длины оптимального расписания выполняется для задачи flow shop с любым регулярным критерием от моментов завершения операций (а отсюда и для всех классических критериев). Далее, мы покажем, что выведенная в [2] оценка на число прерываний в оптимальном расписании, полученная для задачи на минимум длины расписания, с небольшой поправкой справедлива и для других регулярных критериев от моментов завершения операций. Более точно, для любой задачи с регулярным критерием и для любого примера этой задачи найдётся такое оптимальное расписание, в котором число прерываний операций не превосходит объявленной оценки.

Наконец, в работе будет приведён алгоритм точного решения задачи flow shop с произвольным регулярным критерием. Хотя ясно, что число таких критериев бесконечно (как минимум, оно континуально), что число допустимых расписаний с прерываниями операций для каждого конкретного примера также континуально и что решение задач с разными критериями требует построения, вообще говоря, различных оптимальных расписаний, тем не менее утверждаем, что при заданном входе задачи мы способны одним единственным запуском нашего алгоритма построить конечное множество допустимых расписаний, среди которых для любой задачи с регулярным критерием найдётся хотя бы одно оптимальное расписание. Кроме того, для каждой задачи с конкретной регулярной целевой функцией и для конкретного её примера наш алгоритм способен за конечное время вычислить оптимум и отыскать оптимальное расписание (удовлетворяющее упомянутой выше оценке на число прерываний) при условии, что имеется конечная процедура вычисления значений целевой функции от таких расписаний.

Работа организована следующим образом. В разд. 1 вводятся исходные понятия и обозначения. Разд. 2 акцентирует внимание читателей на различии между свойствами оптимальных решений для задачи flow shop с критерием «минимум длины расписания» и свойствами, которые выполнимы для любой задачи flow shop с произвольным регулярным критерием от моментов окончания операций (будем называть её *задачей* $\mathcal{D}$). Разд. 3 посвящён анализу свойств оптимальных расписаний задачи $\mathcal{D}$. В разд. 4 описывается жадный алгоритм построения допустимых расписаний задачи $\mathcal{D}$ и исследуются свойства получаемых расписаний. В частности, показывается, что одно из строящихся расписаний является оптимальным. Приводится оценка на число прерываний в таком распи-

сании. В разд. 5 проводится анализ свойств критических путей в жадном расписании. В результате анализа устанавливается свойство «суперцелочисленности» длины оптимального расписания.

1. Исходные понятия и обозначения

Будем обозначать множество перестановок индексов $\{1,\dots,n\}$ через $\mathcal{P}_n$; $\mathcal{I}_{m,n}$ — множество индивидуальных задач (примеров) задачи Джонсона с параметрами m и n.

В формулируемой ниже задаче $\mathcal{D}$ требуется найти оптимальное расписание выполнения заданного семейства операций $\mathcal{O}$. Понятие *операции* является базовым для этой задачи и в данной статье не определяется. Тем не менее, такие понятия как *расписание* и *прерывание операции* нуждаются в исходных определениях.

Определение 1. Часть операции, непрерывно выполняемую в заданном расписании в некотором интервале времени, будем называть *фрагментом* операции. Произвольный фрагмент длины ε будем называть *ε-фрагментом*, 0-фрагмент будем называть *нулевым*, а максимальный по включению фрагмент — *максимальным фрагментом.*

Расписание для операции $O_{ij}\in\mathcal{O}$ задаётся парой $(k_{ij},\mathcal{T}_{ij})$, где k_{ij} — число максимальных фрагментов, на которые разбивается операция, а

$$\mathcal{T}_{ij}=\big\{\big[s_{ij}^k,c_{ij}^k\big]\,|\,k=1,\dots,k_{ij}\big\}$$

— семейство интервалов времени, выделяемых для выполнения этих фрагментов $\big(0\leqslant s_{ij}^1\leqslant c_{ij}^1\leqslant\dots\leqslant s_{ij}^{k_{ij}}\leqslant c_{ij}^{k_{ij}}\big)$. (Как будет видно из последующей формулировки задачи $\mathcal{D}$, в допустимом расписании суммарная длина фрагментов операции должна равняться её длине: $\sum\limits_{k=1}^{k_{ij}}\big(c_{ij}^k-s_{ij}^k\big)=p_{ij}$.) Через $|O'|, s_S(O')$ и $c_S(O')$ будем обозначать длину фрагмента O' и моменты его начала и окончания в заданном расписании S; $s_S(O_{ij})=s_{ij}^1$ и $c_S(O_{ij})=c_{ij}^{k_{ij}}$ обозначают моменты начала и окончания операции O_{ij}. *Полное расписание* S определяется как совокупность расписаний по всем операциям:

$$S\doteq\{(k_{ij},\mathcal{T}_{ij})\,|\,O_{ij}\in\mathcal{O}\}.$$

Величину $C_{\max}(S)=\max\limits_{i,j}\,c_S(O_{ij})$ будем называть *длиной расписания* S.

Будем говорить, что операция O_{ij} имеет *прерывания* в заданном расписании S, если расписание выполнения операции состоит из двух и более максимальных фрагментов. Момент окончания каждого из макси-

мальных фрагментов операции (за исключением её последнего фрагмента) называется *моментом прерывания* операции. Когда прерывания операций запрещены, для определения расписания достаточно задать лишь моменты начала всех операций: $S \doteq \{s(O_{ij}) \,|\, O_{ij} \in \mathcal{O}\}$.

Точку t будем называть *точкой выполнения* операции O_{ij} в расписании S, если она принадлежит одному из интервалов семейства $\mathcal{T}_{ij}$, и *внутренней точкой* операции O_{ij}, если она является внутренней точкой одного из интервалов семейства $\mathcal{T}_{ij}$.

Замечание 1. В [1] дано более общее определение расписания с прерываниями операций, допускающее разбиение операций на счётное число максимальных фрагментов. (Более того, в таких расписаниях не гарантирована даже возможность перечисления этих фрагментов слева направо числами $1, 2, \ldots$ в тех случаях, когда в расписании имеется хотя бы одна конечная точка накопления фрагментов.) Однако в этой же работе для достаточно общей модели (включающей модель flow shop как частный случай) и для произвольной регулярной целевой функции доказана теорема о *конечности числа прерываний*, согласно которой для любого примера, имеющего оптимальное расписание, существует и оптимальное расписание с конечным числом прерываний. Таким образом, можно заранее ограничиться лишь рассмотрением расписаний с конечным числом прерываний, что мы и сделали в приводимом выше определении.

Задача $\mathcal{D}$. Имеются множества машин $\mathcal{M} = \{M_1, \ldots, M_m\}$ и работ $\mathcal{J} = \{J_1, \ldots, J_n\}$. Каждая работа $J_j \in \mathcal{J}$ имеет m операций O_{ij}, $i = 1, \ldots, m$, которые должны выполняться в предписанном порядке $O_{1j} \prec \cdots \prec O_{mj}$; здесь $O' \prec O''$ означает, что операция O'' может начаться не раньше, чем закончится операция O'. Операция O_{ij} выполняется на машине M_i за время p_{ij} (называемое *длительностью операции*), причём никакие две операции, выполняемые на одной машине, не могут выполняться одновременно в течение ненулевого отрезка времени. Разрешается прерывание любой начатой операции в любой момент времени (с последующим возобновлением и завершением операции без какого-либо штрафа), причём разрешается неограниченное (но конечное) число прерываний любой операции. При этом суммарная длина фрагментов, на которые разбивается операция, должна равняться длительности операции. Требуется построить расписание S, удовлетворяющее перечисленным выше требованиям и минимизирующее заданную регулярную функцию от моментов окончания операций.

Сформулируем задачу (будем называть её *базовой*) на составление расписания выполнения *заданий* из заданного множества $\mathcal{J}$.

Базовая задача. Заданы конечное множество $\mathcal{O}$ операций, которые требуется выполнить без прерываний, и взвешенный орграф $G_p = (\mathcal{O}, U)$; каждой вершине $o_j \in \mathcal{O}$ приписан неотрицательный вес p_j, называемый *длительностью операции* o_j. Расписание $S = \{s_S(o_j) \geqslant 0 \,|\, j \in \mathcal{O}\}$ выполнения операций из множества $\mathcal{O}$ называется *допустимым* (относительно графа G_p), если для любой дуги $(o_i, o_j) \in U$ выполняется неравенство $s_S(o_j) \geqslant s_S(o_i) + p_i$. Требуется построить допустимое расписание минимальной длины.

Определение 2. Допустимое расписание S' для базовой задачи с заданным графом $G_p = (\mathcal{O}, U)$ называется *ранним* относительно графа G_p, если для любого другого расписания S, допустимого относительно G_p, и для любой операции $o_j \in \mathcal{O}$ выполняется соотношение $s_{S'}(o_j) \leqslant s_S(o_j)$.

Определение 3. Пусть $G_p = (\mathcal{J}, U)$ — орграф, вершинам которого приписаны веса. *Путём* $P = (j_1 \to j_2 \to \cdots \to j_k)$ в графе G_p называется такая последовательность вершин $j_i \in \mathcal{J}$, что $(j_i, j_{i+1}) \in U$ для всех $i = 1, \ldots, k-1$. *Длиной* пути $P \subset G_p$ называется сумма весов вершин пути, обозначаемая $|P|$. *Критическим путём* в графе G_p называется путь максимальной длины.

Из календарного планирования известна следующая

Лемма 1. *Для любого взвешенного графа G_p без контуров положительной длины существует единственное расписание, раннее относительно G_p. Оно оптимально для базовой задачи с графом G_p, а его длина совпадает с длиной критического пути в G_p.*

Определение 4. Отношение $\prec$, определённое на множестве пар элементов из V, называется отношением (частичного) *строгого порядка*, если оно иррефлексивно и транзитивно, откуда следует, что для любой пары $v', v'' \in V$ выполняется не более одного из двух отношений: $v' \prec v''$ и $v'' \prec v'$ (т. е. отношение антисимметрично). При этом отношение $\prec$ называется отношением *полного* (*линейного*) *строгого порядка*, если оно определено для любой пары элементов из V.

Будем говорить, что граф $G_p = (V, U)$ задаёт *допустимую схему расписания* для заданного примера $I = (p_{ij}) \in \mathcal{I}_{m,n}$ задачи $\mathcal{D}$, если:

множество вершин V представляет собой объединение непересекающихся подмножеств V_{ij} $(i = 1, \ldots, m;\ j = 1, \ldots, n)$;

вершинам $v \in V$ приписываются неотрицательные веса $p(v)$ такие, что $\sum\limits_{v \in V_{ij}} p(v) = p_{ij}$;

множество U состоит из дуг (v', v''), определённых для следующих пар вершин:

$v' \in V_{i,j},\ v'' \in V_{i+1,j}$ $(i = 1, \ldots, m-1;\ j = 1, \ldots, n)$;

на множестве пар вершин $v', v'' \in V_i \doteq \bigcup\limits_{j=1}^{n} V_{ij}$ $(i = 1, \ldots, m)$ определено отношение $\prec$ линейного строгого порядка. Множество U включает дугу (v', v''), если и только если $v' \prec v''$.

Множество $\mathcal{G}(I)$ состоит из всех взвешенных орграфов $G_p = (V, U)$, задающих допустимые схемы расписаний примера I. (Графы $G_p \in \mathcal{G}(I)$ будем также называть *схемами*.) Для заданной схемы G_p элементы каждого множества V_{ij} будут ассоциироваться с максимальными фрагментами операции O_{ij} в некотором расписании, а веса $p(v)$ вершин $v \in G_p$ — с длинами этих фрагментов. Таким образом, расписание S для базовой задачи с графом G_p может рассматриваться как расписание для примера I задачи $\mathcal{D}$. Справедливо

Утверждение 1. *Для любого примера $I \in \mathcal{I}_{m,n}$ задачи $\mathcal{D}$ и любой схемы $G_p \in \mathcal{G}(I)$ каждое расписание S, допустимое для базовой задачи с графом G_p, является допустимым и для примера I.*

Пусть S — допустимое расписание для заданного примера I задачи $\mathcal{D}$. Оно задаёт разбиение операций на максимальные фрагменты и для каждой машины задаёт порядок выполняемых на ней фрагментов. Пусть O', O'' — максимальные фрагменты, выполняемые на одной машине, $O' \neq O''$. Будем говорить, что фрагмент O' *строго предшествует* фрагменту $O'' \neq O'$ в расписании S (обозначается $O' \prec_S O''$), если O' начинается раньше, чем заканчивается O''. Нетрудно видеть, что отношение $\prec_S$ является отношением «почти линейного» строгого порядка на множестве фрагментов операций, выполняемых на одной машине: оно транзитивно и определено для каждой пары различных фрагментов O', O'', за исключением пар нулевых фрагментов O', O'', выполняемых одновременно. Чтобы получить из $\prec_S$ отношение линейного порядка $(\prec)$ на множестве максимальных фрагментов машины M_i, достаточно на каждом (максимальном по включению) подмножестве её нулевых максимальных фрагментов, выполняемых одновременно, выбрать какой-либо вариант линейного порядка. Далее определим взвешенный граф $G_p(S) = (V(S), U(S))$, вершинами которого являются все максимальные фрагменты операций в расписании S. Вершине $v \in V(S)$ приписывается вес, равный длине данного фрагмента. Для пары фрагментов $O', O'' \in V(S)$ определяем дугу $(O', O'') \in U(S)$ в двух случаях

если $O' \in O_{i,j},\ O'' \in O_{i+1,j}$ для некоторых i, j;

если O' и O'' выполняются на одной машине и $O' \prec O''$.

Нетрудно видеть, что таким образом определённый граф $G_p(S)$ принадлежит $\mathcal{G}(I)$ (т. е. задаёт допустимую схему расписания исходного примера I) и расписание S является допустимым решением базовой задачи с графом $G_p(S)$. Из этого замечания и утверждения 1 следует, что множества допустимых расписаний примера I исходной и базовой задач (по отношению ко всей совокупности графов $G_p \in \mathcal{G}(I)$) совпадают. Ясно, что оптимальное расписание задачи $\mathcal{D}$ с заданным входом I следует искать среди ранних относительно семейства графов $G_p \in \mathcal{G}(I)$ расписаний. Таким образом, решение исходной задачи сводится к решению совокупности базовых задач относительно семейства графов $G_p \in \mathcal{G}(I)$ и выбору из найденных оптимальных (для базовых задач) решений наилучшего по произвольному регулярному критерию задачи.

Так как любой граф $G_p \in \mathcal{G}(I)$ является ациклическим, то лемма 1 гарантирует существование единственного расписания, раннего относительно G_p. Тем самым справедливо

Утверждение 2. *Для отыскания оптимального решения примера I задачи $\mathcal{D}$ достаточно для каждой схемы $G_p \in \mathcal{G}(I)$ построить раннее расписание базовой задачи с графом G_p и из множества полученных расписаний выбрать оптимальное.*

Понятно, что утверждение 2 ещё не даёт конструктивного способа решения задачи $\mathcal{D}$, поскольку мощность множества $\mathcal{G}(I)$ бесконечна.

2. Различие свойств оптимальных расписаний для разных регулярных критериев

Определение 5. Будем говорить, что в расписании S работы J_{j_1} и J_{j_2} ($j_1 \neq j_2$) выполняются на машине M_i с *перекрёстом*, если найдётся пара фрагментов (O'_{ij_1}, O''_{ij_1}) операции O_{ij_1} и пара фрагментов (O'_{ij_2}, O''_{ij_2}) операции O_{ij_2}, выполняемых в порядке $O'_{ij_1} \prec_S O'_{ij_2} \prec_S O''_{ij_1} \prec_S O''_{ij_2}$.

Определение 6. *Точками переключения* в расписании S называем моменты времени, в которые происходит изменение выполняемой операции хотя бы на одной из машин. (При этом простой машины считается выполнением «пустой» операции.) Таким образом, точками переключения являются моменты прерывания, возобновления, начала и завершения операций.

Определение 7. Момент t будем называть моментом *w-перехода* работы J_j с машины M_i на машину M_{i+1} в расписании S, если $t = c_S(O_{ij}) =$

$s_S(O_{i+1,j})$. При этом работу J_j и операции $O_{ij}, O_{i+1,j}$ будем называть *переходными*.

Определение 8. Допустимое расписание S задачи $\mathcal{D}$ называется *плотным*, если не существует непустого интервала $(t_1, t_2) \subseteq (0, C_{\max}(S))$ и машины M_i, таких что во всём интервале (t_1, t_2) машина M_i простаивает и для каждого $t \in (t_1, t_2)$ множество её операций, готовых к выполнению в момент t, непусто.

Определение 9. Машину M_1 будем называть *первой*, машины M_2, $\ldots, M_{m-1}$ — *средними*, а машины $M_2, \ldots, M_m$ — *непервыми*.

Определение 10. Пусть задано частичное расписание S. Если момент $c_S(O_{ij})$ завершения операции O_{ij} в расписании S не определён, полагаем $c_S(O_{ij}) := \infty$. Определим момент $r_S(O_{ij})$ *начала готовности операции* O_{ij} *к выполнению* в расписании S: $r_S(O_{ij}) = 0$ при $i = 1$; $r_S(O_{ij}) = c_S(O_{i-1,j})$ при $i > 1$. Будем говорить, что операция O_{ij} *готова к выполнению* в момент времени t, если $r_S(O_{ij}) \leqslant t \leqslant c_S(O_{ij})$.

В [2] доказана следующая лемма о свойствах оптимальных расписаний для задачи Flow Shop с разрешением прерываний и критерием быстродействия.

Лемма 2 [2]. *Для любого примера $I \in \mathcal{I}_m$ задачи $F|pmtn|C_{\max}$ существует оптимальное расписание S, удовлетворяющее свойствам $(1')$–$(8')$.*

$(1')$ *Расписание S плотное.*

$(2')$ *Всякая операция O_{ij}, чей фрагмент выполняется на машине M_i между двумя фрагментами (O' и O'') другой операции ($O_{ij'}$), становится готовой к выполнению не ранее чем с момента завершения фрагмента O'.*

$(3')$ *Каждое прерывание на средней машине M_i происходит в момент w-перехода некоторой работы J_j с машины M_{i-1} на машину M_i.*

$(4')$ *На машинах M_1 и M_m все операции выполняются без прерываний.*

$(5')$ *На машине M_m работы выполняются в порядке их окончания на M_{m-1}.*

$(6')$ *На машине M_1 работы выполняются в порядке их начала на M_2.*

$(7')$ *Расписание лексикографически минимизирует вектор-функцию $(F_1(C), F_2(C))$, где $F_1(C) = C_{\max}$, $F_2(C) = \sum\limits_{i=2}^{m-1} \sum\limits_{j=1}^{n} c(O_{ij})$ — сумма времён окончания операций на средних машинах, а C — вектор моментов окончания операций.*

$(8')$ *Каждая нулевая операция O_{ij} на средней машине M_i выполняется в момент w-перехода работы J_j с M_{i-1} на M_i.*

Следует отметить, что не все из приведённых в этой лемме свойств справедливы для любого регулярного критерия. В частности, рассмотрим задачу $F|pmtn|\sum C_{ij}$ с критерием «минимум суммы времён окончания операций» (будем ссылаться на неё как на задачу $\mathcal{B}$) и проведём её сравнение с классической задачей Flow Shop на минимум длины расписания на простейших двухмашинных входах.

Пример задачи $\mathcal{B}$, опровергающий свойство (4′) для машины M_m, состоит из двух работ и двух машин. У первой работы длительности операций на машинах M_1 и M_2 равны 1 и 5, у второй — 2 и 1 соответственно. Оптимум, равный 15, достигается на единственном расписании, в котором операция O_{21} на второй машине имеет неизбежное прерывание (рис. 1).

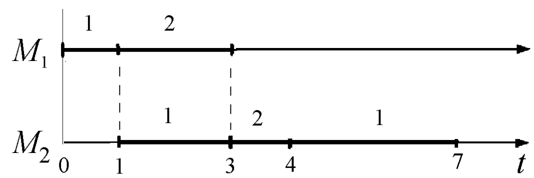


Рис. 1. Пример, опровергающий свойство (4′) для машины M_2

Поскольку свойство (5′) в лемме 2 сформулировано так, что предполагает выполненным свойство (4′), то приведённый выше пример также опровергает и свойство (5′).

Свойство (6′) опровергается примером из трёх работ на двух машинах с векторами длительностей (0,10), (3,8) и (7,4). Его оптимальное расписание показано на рис. 2.

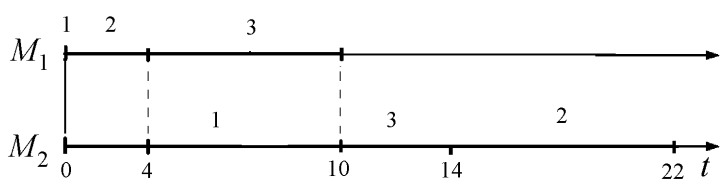


Рис. 2. Пример, опровергающий свойство (6′) для машины M_1

Для доказательства того, что никакое оптимальное расписание данного примера не обладает свойством (6′), достаточно перебрать все комбинации приоритетов работ на машинах M_1 и M_2 (таких комбинаций 36), для каждой построить «жадное» расписание (алгоритмом, описанным в разд. 4), из них оставить только оптимальные и убедиться в справедливости декларируемого свойства. (Согласно следствию 2 из разд. 4 все

оптимальные расписания любого примера данной задачи могут быть получены таким способом.)

3. Анализ свойств оптимальных расписаний в задаче flow shop с прерываниями и произвольным регулярным критерием

В [1] сформулирована общая задача GP с ξ операциями $\{O_1, \dots, O_\xi\}$, с допущением прерываний операций и k регулярными целевыми функциями $F_1(C)$, $\dots, F_k(C)$, зависящими от вектора $C(S) = (c(O_1), \dots, c(O_\xi))$ моментов завершения операций в заданном расписании S. Доказана

Теорема 1 [1]. *Для любого примера задачи GP с непустым множеством допустимых решений существует допустимое расписание S, лексикографически минимизирующее вектор-функцию $(F_1(C(S)), \dots, F_k(C(S)))$.*

Так как задача $\mathcal{D}$ является частным случаем задачи GP и её целевая функция регулярна, из теоремы 1 следует существование оптимального расписания задачи $\mathcal{D}$.

Определение 11. Расписание S для примера I называется *ранним*, если оно раннее относительно графа $G_p(S)$.

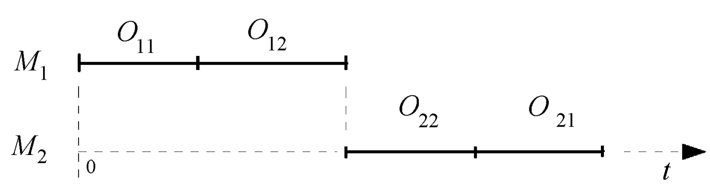


Рис. 3. Раннее расписание, не являющееся плотным

Замечание 2. Всякое плотное расписание S является ранним. Обратное неверно: на рис. 3 изображено расписание S выполнения двух работ на двух машинах, раннее относительно графа $G_p(S)$, но при этом неплотное.

Определение 12. *Несущественными фрагментами* в заданном расписании S будем называть все нулевые максимальные фрагменты, за исключением первого фрагмента каждой нулевой операции.

Замечание 3. Из любого допустимого расписания можно удалить все несущественные фрагменты без ущерба для качества расписания,

т. е. а) с сохранением допустимости расписания; б) без увеличения моментов окончания операций; в) с уменьшением числа прерываний операций. Поэтому мы можем предполагать, что все рассматриваемые далее расписания не содержат несущественных фрагментов. В то же время, мы обязаны учитывать нулевые операции: хотя они не оказывают влияния на длину расписания, они могут играть существенную роль при подсчёте числа прерываний операций.

Лемма 3. *Пусть* $F_1(C)$ *— регулярная функция из условия задачи* $\mathcal{D}$, $F_2(C) = \sum\limits_{i,j} c\,(O_{ij})$ *— сумма моментов окончания всех операций,* C *— вектор моментов окончания операций. Тогда всякое расписание* S *для заданного примера задачи* $\mathcal{D}$, *лексикографически минимизирующее вектор-функцию* $(F_1(C), F_2(C))$, *удовлетворяет перечисленным ниже свойствам.*

(1*) *Расписание* S *плотное.*

(2*) *Всякая операция* O_{ij}, *чей фрагмент выполняется на машине* M_i *между двумя фрагментами* (O' *и* O'') *другой операции* ($O_{ij'}$), *становится готовой к выполнению не ранее чем с момента завершения фрагмента* O': $r_S(O_{ij}) \geqslant c_S(O')$. *Как следствие, в* S *нет перекрёстов.*

(3*) *Каждое прерывание на непервой машине* M_i *происходит в момент* w-*перехода некоторой работы* J_j *с машины* M_{i-1} *на машину* M_i.

(4*) *На машине* M_1 *все операции выполняются без прерываний.*

(5*) *Каждая нулевая операция* O_{ij} *на непервой машине* M_i *выполняется в момент* w-*перехода работы* J_j *с* M_{i-1} *на* M_i, *а на машине* M_1 *— в момент* 0.

Доказательство. Рассмотрим расписание S, лексикографически минимизирующее вектор-функцию $F(C) = (F_1(C), F_2(C))$. Свойства (1*) и (5*) легко следуют из минимальности вектор-функции. Действительно, если бы какая-то нулевая операция O_{ij} на непервой машине M_i выполнялась позже момента t окончания операции $O_{i-1,j}$, то, перенеся её выполнение на момент t, мы могли бы уменьшить компоненту $F_2(C)$. Аналогично, если бы нарушилась плотность расписания S в непустом интервале времени (t_1, t_2) (когда какая-то машина M_i простаивает, в то время как имеется готовая для выполнения операция O_{ij}), мы могли бы перенести в этот интервал непустой завершающий фрагмент операции O_{ij}, уменьшив при этом как время её завершения, так и компоненту $F_2(C)$, что в обоих случаях противоречит выбору расписания S.

Пусть не выполняется свойство (2*), т. е. фрагмент O'_{ij} операции O_{ij} выполняется между фрагментами O' и O'' операции $O_{ij'}$ и при этом $r_S(O_{ij}) < c_S(O')$. Из свойства (5*) следует, что O_{ij} — ненулевая опе-

рация. Следовательно, по замечанию 3 её фрагмент O'_{ij} также ненулевой. Без ограничения общности можем считать, что $c_S(O_{ij}) < c_S(O_{ij'})$ и что O'_{ij} — последний фрагмент операции O_{ij}. В противном случае при $c_S(O_{ij'}) < c_S(O_{ij})$ мы приходим к искомой ситуации, поменяв ролями операции O_{ij} и $O_{ij'}$. Наконец, если предположить, что имеет место равенство $c_S(O_{ij'}) = c_S(O_{ij})$, то хотя бы одна из двух операций должна будет завершиться нулевым фрагментом. В силу замечания 3 этот фрагмент является единственным фрагментом нулевой операции. Но операция O_{ij}, как мы выяснили, ненулевая, а операция $O_{ij'}$ также не может быть нулевой, так как имеет два разделённых фрагмента O' и O''. Таким образом, случай равенства $c_S(O_{ij'}) = c_S(O_{ij})$ невозможен.

Определим $\varepsilon \doteq \min\{|O'|, |O'_{ij}|, c_S(O') - r_S(O_{ij})\}$. Тогда $\varepsilon > 0$, и, поменяв местами последние ε-подфрагменты фрагментов O'_{ij} и O', мы уменьшим время окончания операции O_{ij} как минимум на ε (с сохранением допустимости расписания), что приведёт к лексикографическому уменьшению вектора F и противоречит выбору расписания S. Свойство (2*) доказано.

Докажем свойство (3*). Пусть в некоторый момент t на непервой машине M_i произошло прерывание операции $O_{ij'}$. Так как по свойству плотности расписания S сразу после t машина M_i не может простаивать, то в момент t на M_i начинает выполняться фрагмент O' некоторой операции $O_{ij''}$. Если $O_{ij''}$ — нулевая операция, то по свойству (5*) момент t является моментом w-перехода работы $J_{j''}$ с M_{i-1} на M_i, и получаем свойство (3*). Если O' — ненулевой фрагмент операции $O_{ij''}$, то из свойства (2*) вытекает, что $r_S(O_{ij''}) \geqslant t$, откуда непосредственно следует (3*).

Докажем теперь свойство (4*). Предположим от противного, что на M_1 в расписании S существует прерывание операций. Среди таковых найдём операцию O_{1j}, заканчивающуюся позже других, и пусть O' и O'' — её предпоследний и последний фрагменты (по замечанию 3 ненулевые). Пусть $\mathcal{O}'$ — множество операций, имеющих хотя бы один ненулевой фрагмент между O' и O''. Так как в S нет перекрёстов, то каждая операция из $\mathcal{O}'$ завершается также между O' и O''. Таким образом, если фрагмент O' присоединить к началу фрагмента O'', то все фрагменты операций из $\mathcal{O}'$, находящиеся между O' и O'', сдвинутся влево, что повлечёт строгое уменьшение времён завершения этих операций, а следовательно, и строгое уменьшение значения функции $F_2(C)$. При этом функция $F_1(C)$ в силу её регулярности также не возрастёт. Получаем противоречие с выбором расписания S. Лемма 3 доказана.

Будем говорить, что расписание S обладает свойством (6*), если оно лексикографически минимизирует вектор-функцию $(F_1(C), F_2(C))$. Будем ссылаться на совокупность свойств (1*)–(4*) как на $(*)$, а на совокупность (1*)–(6*) как на $(**)$.

Поскольку функция F_2 является целевой функцией задачи $\mathcal{B}$, из леммы 3 вытекает

Следствие 1. *Всякое оптимальное расписание для заданного примера I задачи $\mathcal{B}$ обладает свойствами* $(**)$.

Поскольку обе функции $F_1(C)$ и $F_2(C)$ регулярны, а задача $\mathcal{D}$ является частным случаем задачи GP, из теоремы 1 следует существование расписания S, минимизирующего вектор $(F_1(C), F_2(C))$. Отсюда с учётом леммы 3 вытекает

Лемма 4. *Для любого примера задачи $\mathcal{D}$ существует оптимальное расписание, обладающее свойствами* $(**)$.

4. Жадный алгоритм $\mathcal{A}_g$ и его свойства

Описываемый в этом разделе алгоритм $\mathcal{A}_g$ относится к семейству алгоритмов жадного типа. Свойства алгоритма $\mathcal{A}_g$ будут использоваться нами в алгоритме точного решения задачи $\mathcal{D}$. Вначале сформулируем общие принципы работы жадного алгоритма при построении допустимого расписания в задаче $\mathcal{D}$, после чего уточним специфику применения такого алгоритма в нашем случае.

Общие принципы работы жадного алгоритма при построении допустимого расписания в задаче $\mathcal{D}$.

1. Временна́я «прогонка» расписания. Искомое *жадное* расписание S_g строится последовательно во времени: для всё большего t оно становится полностью определённым в интервале $[0, t]$. При этом момент t изменяется не непрерывно, а дискретно (скачкообразно), перебегая от точки переключения τ_k (см. определение 6) к следующей $(\tau_{k+1} > \tau_k)$. Процесс построения расписания заканчивается при $t = C_{\max}(S_g)$.

2. Необратимость алгоритма. Частичное расписание, определённое на k-м шаге алгоритма в интервале $[0, \tau_k]$, не изменяется на последующих шагах алгоритма.

3. Плотность расписания. Формально определение точки переключения τ_k (см. определение 6) включает в себя моменты перехода машины:

(a) от выполнения одной операции к выполнению другой (при этом первая либо прерывается в момент τ_k, либо завершается);

(b) после прерывания выполняемой операции — к простою;
(c) после завершения операции — к простою;
(d) от простоя — к выполнению прерванной ранее операции;
(e) от простоя — к началу новой операции.

Однако для рассматриваемой нами задачи Flow Shop исключается появление некоторых из описанных выше ситуаций в *жадном расписании*, построенном при помощи жадного алгоритма. В частности, полностью исключаются ситуации (b) и (d), а ситуация (e) (с началом некоторой операции O_{ij}) возможна лишь в момент окончания операции $O_{i-1,j}$. Все перечисленные ограничения проистекают из требования *плотности* строящегося расписания S_g (см. определение 8).

4. Приоритетность. В каждый момент времени t на машине M_i выполняется наиболее приоритетная операция из множества $\mathcal{O}_i(S_g,t)$ операций, готовых к выполнению на этой машине в момент t (см. определение 10). При этом *отношение приоритетности* на множестве всех операций каждой машины M_i задаётся априори (т. е. до построения расписания) и остаётся неизменным в течение всего времени работы алгоритма. Отсюда следует, что замена выполняемой на машине M_i операции может происходить лишь в моменты изменения множества $\mathcal{O}_i(S_g,t)$, а точнее, лишь в двух случаях: (a) когда завершается какая-то операция машины M_i; (b) когда в множестве $\mathcal{O}_i(S_g,t)$ появляется новая операция с наибольшим приоритетом (что совпадает по времени с моментом завершения какой-то операции на машине M_{i-1}). Таким образом, множество всех точек переключения в жадном расписании совпадает с множеством моментов завершения операций в этом расписании (откуда, в частности, следует, что число точек переключения не превосходит числа операций). Из свойства приоритетности также вытекает следующее свойство.

5. Причинная обусловленность прерываний. Прерывание операции O_{ij} в жадном расписании возможно лишь в ситуации (a) и лишь в момент w-перехода какой-либо работы $J_{j'}$ ($j' \neq j$) с машины M_{i-1} на машину M_i.

Специфика жадного алгоритма $\mathcal{A}_g(\pi^1,\dots,\pi^m;I)$ состоит в том, что для заданного примера $I \in \mathcal{I}_{m,n}$ с помощью перестановок $\pi^1,\dots,\pi^m \in \mathcal{P}_n$ задаются строгие приоритетные порядки выполнения операций на машинах $M_1,\dots,M_m$: действительно, каждая перестановка π^i определяет перестановку номеров работ $\{1,\dots,n\}$, а поскольку каждая работа J_j в модели Flow Shop представлена на каждой машине M_i единственной операцией O_{ij}, тем самым определяется приоритетный порядок на множестве операций машины M_i. Порядок является *строгим*: из любых двух

операций одна всегда приоритетнее другой.

Лемма 5. *Для любых входа* $I \in \mathcal{I}_{m,n}$ *и перестановок* $\pi^1, \dots, \pi^m \in \mathcal{P}_n$ *алгоритм* $\mathcal{A}_g(\pi^1, \dots, \pi^m; I)$ *строит расписание* S_g, *обладающее свойствами* (∗) *из леммы* 3.

Доказательство. Из принципа 3 следует плотность расписания S_g (свойство (1*)), а из принципа 5 — свойство (3*).

Докажем свойство (2*). Если операция $O_{ij'}$ была прервана в момент t, то до момента её возобновления на M_i могут выполняться лишь более приоритетные, чем $O_{ij'}$, операции (по принципу 4). Если бы какая-то из этих операций ($O_{ij''}$) имела момент готовности $r(O_{ij''}) < t$, то это означало бы, что в интервале времени $(t-\varepsilon, t)$ выполняется не самая приоритетная операция ($O_{ij'}$), что противоречило бы принципу 4. Противоречие доказывает свойство (2*).

Далее, поскольку все работы готовы к выполнению на M_1, начиная с момента 0, машина M_1 от начала до конца работает без простоев и выполняет работы согласно их априорному приоритетному порядку π^1 без прерываний (свойство (4*)).

Лемма 6. *Для любого примера* $I \in \mathcal{I}_{m,n}$ *задачи* $\mathcal{D}$ *и любого расписания* S, *обладающего свойствами* (∗∗) *из леммы* 3, *существуют перестановки* $\pi^1, \dots, \pi^m \in \mathcal{P}_n$ *такие, что строящееся согласно алгоритму* $\mathcal{A}_g(\pi^1, \dots, \pi^m; I)$ *расписание* S_g *совпадает с* S.

Доказательство. Рассмотрим оптимальное расписание S, обладающее свойствами (∗∗). На множестве операций машины M_k определим отношение «$\lhd_S$» следующим образом: $O_{kj'} \lhd_S O_{kj''}$, если $j' \neq j''$ и выполняется одно из соотношений

$$c_S(O_{kj'}) \leqslant s_S(O_{kj''}), \tag{1}$$

$$s_S(O_{kj''}) < s_S(O_{kj'}) \leqslant c_S(O_{kj'}) < c_S(O_{kj''}). \tag{2}$$

Так как в расписании S нет перекрёстов (по свойству (2*)) и нет несущественных фрагментов (замечание 3), то для любой пары операций $\{O_{kj'}, O_{kj''}\}$ имеет место одно из отношений: $O_{kj'} \lhd_S O_{kj''}$ либо $O_{kj''} \lhd_S O_{kj'}$, а для пар нулевых операций, выполняемых одновременно, имеют место оба отношения (поскольку для каждой пары выполняется (1)). Нетрудно убедиться, что $\lhd_S$ является отношением линейного нестрогого порядка на множестве операций машины M_k, что позволяет нам упорядочить все операции машины M_k в цепочку: $O_{k\pi^k_1} \lhd_S O_{k\pi^k_2} \lhd_S \cdots \lhd_S O_{k\pi^k_n}$ для некоторой перестановки $\pi^k = (\pi^k_1, \dots, \pi^k_n)$. (При наличии нескольких

нулевых операций машины M_k, выполняемых одновременно в расписании S, перестановка π^k определяется неоднозначно. Однако для наших целей в качестве π^k годится любая из перестановок, согласованная с отношением $\lhd_S$.) Используя определённые таким образом перестановки $\pi^1, \ldots, \pi^m$ в алгоритме $\mathcal{A}_g(\pi^1, \ldots, \pi^m; I)$, построим жадное расписание S_g и докажем его оптимальность.

Докажем индукцией по $k = 1, \ldots, m$, что на машине M_k расписание S_g совпадает с S. По лемме 5 расписание S_g обладает свойствами $(*)$. Из свойств (1*) и (4*) следует, что операции машины M_1 выполняются в расписании S_g без прерываний и без простоя машины согласно их приоритетам, определённым перестановкой π^1. Поскольку π^1 соответствует порядку выполнения работ на машине M_1 в расписании S, совпадение расписаний S и S_g на M_1 доказано, что обеспечивает базис индукции.

Пусть $1 < k \leqslant m$ и совпадение расписаний S и S_g на машинах M_i $(i < k)$ доказано. Докажем, что S и S_g совпадают на машине M_k. Пусть это не так, и пусть t' — наименьший момент времени, начиная с которого расписания не совпадают. Это несовпадение может реализовываться одним из двух способов: (a) какая-то нулевая операция $O_{kj''}$, выполняемая в момент t' в одном из расписаний, выполняется в другом в момент $t'' > t'$; (b) в непустом интервале (t', t'') в одном из расписаний выполняется ненулевая операция $O_{kj'}$, а в другом — другая операция $O_{kj''}$ (либо машина простаивает).

Пусть имеет место случай (a). Поскольку на M_{k-1} расписания S и S_g совпадают, а в расписании S по свойству (5*) нулевая операция $O_{kj''}$ выполняется в наиболее ранний возможный момент (в момент окончания операции $O_{k-1,j''}$), то $t' = c_S(O_{k-1,j''}) = c_{S_g}(O_{k-1,j''}) = c_S(O_{kj''}) < t'' = c_{S_g}(O_{kj''})$. Так как S_g — жадное расписание, в интервале (t', t'') машина M_k не простаивает, причём выполняться на M_k в этом интервале в расписании S_g могут лишь операции, более приоритетные чем $O_{kj''}$. Поскольку нулевая операция $O_{kj''}$ не может быть связана с более приоритетной операцией $O_{kj'}$ соотношением (2), по определению отношения $\lhd_S$ должно выполняться соотношение (1), т. е. в интервале (t', t'') на машине M_k в расписании S_g могут выполняться лишь те операции, которые заканчиваются в расписании S не позже момента t'. Однако все ненулевые операции на M_k, заканчивающиеся в S не позже момента t', и в расписании S_g заканчиваются не позже t' (поскольку по предположению расписания совпадают в интервале $[0, t')$). Нулевые же операции не могут заполнить собой ненулевой отрезок (t', t''). Следовательно, случай (a) невозможен.

Предположим, что имеет место случай (b). Легко понять, что ни в одном из расписаний S_g и S машина M_k не может простаивать в интервале (t', t''). Она не может простаивать в S_g по свойству плотности расписания, так как в течение всего интервала (t', t'') множество операций, готовых к выполнению на M_k, непусто (оно содержит по крайней мере ту ненулевую операцию, которая в этом интервале выполняется в расписании S). С другой стороны, M_k не может простаивать и в расписании S (при условии, что в S_g в это время выполняется ненулевая операция O_{kj}), поскольку в этом случае мы могли бы без помех уменьшить время окончания операции O_{kj} в расписании S, что противоречит свойству (6*) расписания S.

Предположим теперь, что на машине M_k в интервале (t', t'') в расписаниях S_g и S выполняются соответственно фрагменты $O'_{kj'}$ и $O'_{kj''}$ ненулевых операций разных работ. Поскольку обе операции, очевидно, готовы к выполнению в момент t' в обоих расписаниях, а жадный алгоритм из множества готовых операций выбирает наиболее приоритетную (по принципу 4), отсюда следует, что $O_{kj'} \lhd_S O_{kj''}$, а значит, выполняется одно из соотношений (1), (2). Предположим, что выполняется (1). Тогда $c_S(O_{kj'}) \leqslant s_S(O_{kj''}) \leqslant t'$, откуда следует, что операция $O_{kj'}$ завершается в расписании S до момента t'. Так как в интервале $[0, t')$ расписания S и S_g совпадают, в расписании S_g операция $O_{kj'}$ завершается до момента t' и её выполнение в непустом интервале (t', t'') невозможно.

Предположим далее, что выполняется (2). Поскольку мы знаем, что часть операции $O_{kj'}$ (в объёме, не меньшем $|O'_{kj'}| = t'' - t'$) выполняется в расписании S после фрагмента $O'_{kj''}$, после чего следует ненулевой завершающий фрагмент операции $O_{kj''}$, то получаем ситуацию, описываемую в свойстве (2*): фрагмент $O'_{kj'}$ операции $O_{kj'}$ выполняется между двумя фрагментами операции $O_{kj''}$. Отсюда следует (по свойству (2*), которым обладает расписание S), что операция $O_{kj'}$ становится готовой к выполнению не раньше момента t'', тогда как мы знаем, что момент её готовности наступает не позже t'. Противоречие доказывает, что случай (b) также невозможен и ситуация несовпадения расписаний S и S_g в интервале (t', t'') невозможна. Совпадение расписаний S и S_g доказано, что завершает доказательство леммы 6.

Из лемм 4 и 6 следует

Теорема 2. *Для любого примера I задачи $\mathcal{D}$ существуют перестановки $\pi^1, \ldots, \pi^m \in \mathcal{P}_n$ такие, что расписание S_g, строящееся согласно алгоритму $\mathcal{A}_g(\pi^1, \ldots, \pi^m; I)$, является оптимальным.*

Из следствия 1 и леммы 6 вытекает

Следствие 2. *Для любого примера I задачи $\mathcal{B}$, определённой в разд. 2, любое из его оптимальных расписаний может быть получено с помощью алгоритма $\mathcal{A}_g(\pi^1, \ldots, \pi^m; I)$ путём задания надлежащих приоритетов $\pi^1, \ldots, \pi^m$ работ на машинах $M_1, \ldots, M_m$.*

Таким образом, для задачи $\mathcal{B}$ мы можем сформулировать более сильное свойство, нежели для более общей задачи $\mathcal{D}$. А именно,

Следствие 3. *Для любого примера задачи $\mathcal{B}$ не только существует оптимальное расписание, являющееся жадным, но и не существует «нежадных» оптимальных расписаний.*

Такое свойство оказывается очень полезным при доказательстве утверждений типа «Любое оптимальное расписание такого-то примера задачи $\mathcal{B}$ обладает/не обладает такими-то свойствами». Для проверки такого утверждения достаточно для заданного примера перебрать все комбинации приоритетных порядков, для каждого построить жадное расписание, из построенных расписаний выделить те, что минимизируют целевую функцию, и убедиться явным образом в выполнении декларируемых свойств.

Теорема 2 также даёт возможность для построения алгоритма точного решения задачи $\mathcal{D}$.

Следствие 4. *Для нахождения оптимального расписания в задаче $\mathcal{D}$ для заданного примера $I \in \mathcal{I}_{m,n}$ достаточно перебрать все наборы из m перестановок $\pi^1, \ldots, \pi^m \in \mathcal{P}_n$, для каждого набора $(\pi^1, \ldots, \pi^m)$ с помощью алгоритма $\mathcal{A}_g(\pi^1, \ldots, \pi^m; I)$ построить расписание S_g и из множества полученных расписаний выбрать оптимальное.*

Теорема 3. *Для любого примера $I \in \mathcal{I}_{m,n}$ задачи $\mathcal{D}$ с m машинами и n работами существует оптимальное расписание, имеющее не более чем $(m-1)(n-1)$ прерываний.*

Доказательство. Для заданного примера I рассмотрим оптимальное расписание S, строящееся жадным алгоритмом $\mathcal{A}_g(\pi^1, \ldots, \pi^m; I)$ при подходящем наборе приоритетных перестановок $(\pi^1, \ldots, \pi^m)$. (Такой набор существует согласно теореме 2.) Согласно 5-му принципу жадных алгоритмов прерывание какой-либо операции O_{kj} на машине M_k может возникнуть только в момент w-перехода с M_{k-1} на M_k другой работы $J_{j'}$ $(j' \neq j)$ для выполнения на M_k операции $O_{kj'}$. Таким образом, всякая операция $O_{kj'}$ может являться причиной не более одного прерывания и лишь в том случае, если $O_{kj'}$ — не первая операция своей работы. Кроме того, операция, выполняемая на какой-то машине первой, также не может прервать никакую другую операцию. Таким образом, «причина-

ми прерываний» в жадном расписании S для примера I задачи $\mathcal{D}$ могут быть лишь непервые операции непервых машин, откуда и вытекает декларируемая в теореме оценка.

5. Анализ свойств критических путей в жадном расписании

Пусть S — расписание, построенное для заданного примера I задачи $\mathcal{D}$ жадным алгоритмом $\mathcal{A}_g(\pi^1, \dots, \pi^m; I)$; $G_p(S)$ — определённый в разд. 1 граф предшествования фрагментов, построенный по расписанию S; $P = (O^1 \to O^2 \to \dots \to O^\alpha)$ — критический путь в графе $G_p(S)$. По лемме 5 и замечанию 2 расписание S является ранним, и по лемме 1 длина расписания S совпадает с длиной пути P. В дальнейшем критический путь в графе $G_p(S)$ будем называть *критическим путём в расписании* S.

Определение 13. Пусть S — допустимое расписание в задаче Джонсона, P — путь в графе $G_p(S)$. *Составом* пути P будем называть множество $\langle P \rangle$ операций, фрагменты которых входят в P. Путь P будем называть *полным*, если он содержит все фрагменты всех операций $O_i \in \langle P \rangle$.

Заметим, что если путь P полный, то его длина $|P|$ однозначно определяется по его составу. Таким образом, если в $G_p(S)$ существует полный критический путь, то по лемме 1 длина расписания S однозначно определяется по составу такого пути.

Как показывает пример на рис. 4, не в любом плотном расписании существует полный критический путь.

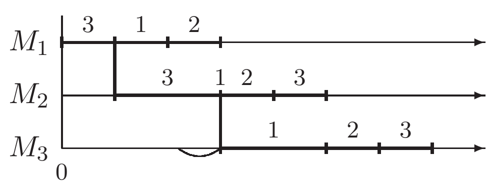


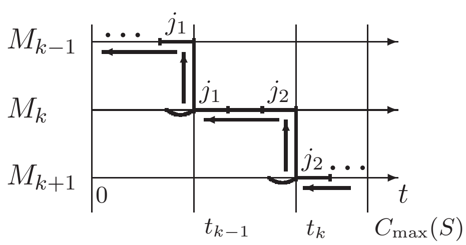
Рис. 4. Плотное расписание, не имеющее полного критического пути

В то же время справедлива

Лемма 7. *В любом допустимом расписании задачи $\mathcal{D}$, обладающем свойствами $(*)$, существует полный критический путь.*

Доказательство. Пусть S — допустимое расписание примера I, обладающее свойствами $(*)$. Определим неубывающую последовательность моментов времени $0 = t_0 \leqslant t_1 \leqslant \dots \leqslant t_m$ обратной индукцией по номеру момента. Полагаем $t_m = C_{\max}(S)$. Пусть моменты $t_k, \dots, t_m$ определены.

В качестве t_{k-1} возьмём ближайший к t_k момент $t \leqslant t_k$, который является либо моментом окончания интервала простоя на машине M_k (см. рис. 5 (а)); выпуклой дугой обозначается интервал простоя), либо моментом прерывания какой-то операции на машине M_k, завершающейся после момента t_k (рис. 5 (б)). Ясно, что на машине M_1 вторая ситуация невозможна, так как S удовлетворяет свойству (4*). Отсюда (ввиду плотности расписания на M_1) следует $t_0 = 0$. Таким образом, совокупность интервалов $\{[t_{k-1}, t_k] \,|\, k = 1, \ldots, m\}$ покрывает всю длину расписания S, и для каждого $k = 1, \ldots, m$ в интервале $[t_{k-1}, t_k]$ машина M_k работает без простоев.



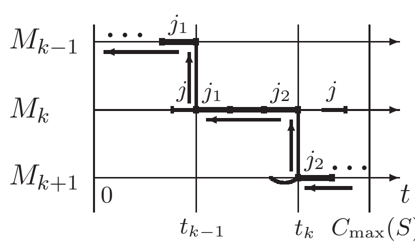


Рис. 5. Возможные варианты звена полного критического пути на машине M_k

Из свойств (1*) и (3*) следует, что каждый момент t_k, $k = 1, \ldots, m-1$, является моментом w-перехода некоторых (возможно, нескольких) работ с M_k на M_{k+1}. Для каждой пары машин (M_k, M_{k+1}) мы выберем одну такую переходную работу (которую будем обозначать j_k) по следующему правилу.

Пусть для момента t_k $(1 < k < m)$ работа j_k определена. Если $t_{k-1} < t_k$, то в качестве j_{k-1} берём произвольную работу из множества переходных с M_{k-1} на M_k в момент t_{k-1}. Если же $t_{k-1} = t_k$, то полагаем $j_{k-1} = j_k$.

Докажем, что в последнем случае работа j_k является переходной в момент t_k с машины M_{k-1} на M_k. Вначале покажем, что операция O_{kj_k} нулевая. По выбору момента t_{k-1} он является либо моментом окончания интервала простоя на машине M_k, либо моментом прерывания некоторой операции $O_{kj'}$. (При этом $j' \neq j_k$, так как O_{kj_k} в момент t_{k-1} завершается, а не прерывается.) В обоих случаях из предположения $|O_{kj_k}| \neq 0$ следует существование в расписании S нулевого максимального фрагмента операции O_{kj_k}, выполняемого в момент t_k, что противоречит замечанию 3.

Итак, O_{kj_k} — нулевая операция, выполняемая в момент t_k. Если t_k — момент окончания простоя на M_k, то равенство $r_S(O_{kj_k}) = t_k$ вытекает

из плотности расписания S. Если же t_k — момент прерывания какой-то операции $O_{kj'}$ ($j' \neq j_k$), то по свойству (2*) получаем $r_S(O_{kj_k}) = t_k$. Таким образом, момент t_k является моментом w-перехода работы j_k с машины M_{k-1} на машину M_k.

В целях унификации наших выкладок к исходному семейству работ добавим две фиктивные работы (с номерами 0 и $n+1$), имеющие нулевые операции на всех машинах. Будем считать, что в расписании S операции работы 0 на всех машинах выполняются первыми, а операции работы $(n+1)$ — последними. Положим $j_0 := 0$; $j_m := n+1$.

Пусть O_{ij}^f и O_{ij}^l обозначают первый и последний максимальные фрагменты операции O_{ij} в расписании S (возможно, они совпадают). По заданной последовательности моментов $t_0 \leqslant t_1 \leqslant \cdots \leqslant t_m$ определим последовательность фрагментов $(O^1, \ldots, O^\alpha)$ в расписании S следующим образом: $P(S) \doteq P_1 \oplus P_2 \oplus \cdots \oplus P_m$, где P_k — последовательность вершин максимального по включению пути в графе $G_p(S)$, ведущего из вершины $O_{kj_{k-1}}^f$ в вершину $O_{kj_k}^l$. Нетрудно убедиться, что каждый из путей P_k ($k = 1, \ldots, m$) существует, поскольку фрагменты $O_{kj_{k-1}}^f$ и $O_{kj_k}^l$ либо совпадают (в случае $t_{k-1} = t_k$), либо (при $t_{k-1} < t_k$) между ними имеет место отношение $O_{kj_{k-1}}^f \prec_S O_{kj_k}^l$. Кроме того, конечная вершина ($O_{kj_k}^l$) пути P_k и начальная вершина (O_{k+1,j_k}^f) пути P_{k+1} принадлежат одной работе, поэтому $(O_{kj_k}^l, O_{k+1,j_k}^f) \in U(S)$. Таким образом, $P(S)$ является путём в графе $G_p(S)$.

Так как в интервале $[t_{k-1}, t_k]$ машина M_k работает без простоев (по выбору моментов t_k), а P_k — максимальный по включению путь из фрагментов операций машины M_k, выполняемых между $O_{kj_{k-1}}^f$ и $O_{kj_k}^l$, то суммарная длина пути P_k равна $t_k - t_{k-1}$. Отсюда длина всего пути $P(S)$ равна $C_{\max}(S)$, т. е. путь $P(S)$ является критическим.

Докажем полноту этого пути, т. е. докажем, что никакая операция не представлена в пути $P(S)$ лишь частью своих фрагментов. Достаточно доказать это для операций непервых машин, поскольку на M_1 операции выполняются без прерываний (по свойству (4*)). Пусть $P(S)$ содержит собственный фрагмент $O' \in C_k$ операции O_{kj}, выполняемый в интервале $[t_{k-1}, t_k]$. Тогда по замечанию 3 фрагмент O' является ненулевым. По правилу выбора момента t_{k-1} никакой фрагмент O'' операции O_{kj} не может заканчиваться строго позже момента t_k. Предположим, что ненулевой фрагмент O'' операции O_{kj} завершается до момента t_{k-1}. В силу плотности расписания S на машине M_k не может быть простоя между фрагментами O'' и O' и, в частности, непосредственно перед моментом

t_{k-1}. Отсюда следует, что момент t_{k-1} был определён как момент прерывания некоторой операции $O_{kj'}$, которая возобновляет своё выполнение после момента t_k (причём, $j' \neq j$, поскольку, как мы выяснили, операция O_{kj} не может иметь фрагментов, выполняемых после t_k). Но в этом случае фрагмент O' операции O_{kj} выполняется между двумя фрагментами операции $O_{kj'}$, и по свойству (2*) операция O_{kj} имеет момент готовности $r_S(O_{kj}) \geqslant t_{k-1}$, что противоречит предположению о выполнении её ненулевого фрагмента O'' до момента t_{k-1}. Лемма 7 доказана.

Из лемм 5, 7 и теоремы 2 вытекают следующие утверждения.

Следствие 5. *Для любого примера I задачи $\mathcal{D}$ и любого набора перестановок $(\pi^1, \ldots, \pi^m)$ расписание S, строящееся жадным алгоритмом $\mathcal{A}_g(\pi^1, \ldots, \pi^m; I)$, содержит полный критический путь.*

Теорема 4. *Длина оптимального расписания любого примера задачи $\mathcal{D}$ с произвольным регулярным критерием от моментов окончания операций совпадает с суммарной длиной операций из некоторого подмножества.*

ЛИТЕРАТУРА

1. **Баптист Ф., Карлье Ж., Керан М., Кононов А. В., Свириденко М., Севастьянов С. В.** Структурные свойства оптимальных расписаний с прерываниями операций // Дискрет. анализ и исслед. операций. — 2009. — Т. 16, № 1. — С. 3–36.
2. **Севастьянов С. В., Чемисова Д. А., Черных И. Д.** О некоторых свойствах оптимальных расписаний в задаче Джонсона с прерываниями // Дискрет. анализ и исслед. операций. Сер. 1. — 2006. — Т. 13, № 3. — С. 83–102.
3. **Танаев В. С., Сотсков Ю. Н., Струсевич В. А.** Теория расписаний. Многостадийные системы. — М.: Наука, 1989. — 328 с.
4. **Chen B., Potts C. N., Woeginger G. J.** A review of machine scheduling: complexity, algorithms and approximability // Handbook of combinatorial optimization. — Boston: Kluwer Acad. Publ., 1998. — Vol. 3. — P. 21–169.
5. **Cho Y., Sahni S.** Preemptive scheduling of independent jobs with release and due times on open, flow and job shops // Oper. Res. — 1981. — Vol. 29. — P. 511–522.
6. **Du J., Leung J. Y.-T.** Minimizing mean flow time in two-machine open shops and flow shops // J. Algorithms. — 1993. — Vol. 14. — P. 341–364.
7. **Gonzalez T., Sahni S.** Open shop scheduling to minimize finish time // J. ACM. — 1976. — Vol. 23. — P. 665–679.
8. **Gonzalez T., Sahni S.** Flowshop and jobshop schedules: complexity and approximation // Oper. Res. — 1978. — Vol. 26, N 1. — P. 36–52.

9. **Johnson S. M.** Optimal two- and three-stage production schedules with setup time included // Naval Research Logistic Quarterly. — 1954. Vol. 1. — P. 61–68.
10. **Lawler E. L., Labetoulle J.** On preemptive scheduling on unrelated parallel processors by linear programming // J. ACM. — 1978. — Vol. 25. — P. 612–619.

Чемисова Дарья Александровна,
e-mail: chemiso@gmail.com

Статья поступила
19 декабря 2008 г.
Переработанный вариант —
11 марта 2009 г.