ДИСКРЕТНЫЙ АНАЛИЗ И ИССЛЕДОВАНИЕ ОПЕРАЦИЙ
Январь—июнь 2000. Серия 2. Том 7, № 1, 9–34

УДК 519.8

ПОЛИНОМИАЛЬНАЯ РАЗРЕШИМОСТЬ ЗАДАЧ КАЛЕНДАРНОГО ПЛАНИРОВАНИЯ СО СКЛАДИРУЕМЫМИ РЕСУРСАМИ И ДИРЕКТИВНЫМИ СРОКАМИ*)

Э. Х. Гимади, В. В. Залюбовский,

С. В. Севастьянов

Исследована задача календарного планирования с ограниченными ресурсами складируемого типа и директивными сроками. Показано, что понятие складируемых ресурсов не сводимо к традиционно рассматриваемым возобновимым и невозобновимым ресурсам. Предложен полиномиальный алгоритм решения задачи. Для мультимодальной модели выделен полиномиально разрешимый случай, когда предложенный алгоритм также находит оптимальное решение.

Введение

В классической задаче календарного планирования с ограниченными ресурсами (ЗКП) необходимо найти расписание выполнения работ с минимальным сроком завершения проекта. При этом должны быть учтены технологические ограничения предшествования, директивные сроки и ограничения на ресурсы. Прерывания работ не допускаются.

Естественным обобщением ЗКП является мультимодальная задача календарного планирования с ограниченными ресурсами (МЗКП). В ней каждая работа может быть выполнена в одной из нескольких мод. Каждая мода представляет собой альтернативный способ выполнения работы, задаваемый комбинацией различных уровней потребления ресурсов и соответствующих длительностей.

В зарубежной литературе (см. обзор [10]) традиционно различаются возобновимые и невозобновимые ограниченные ресурсы. Для *возобновимого* ресурса для каждой работы задается количество ресурса, требуемое в каждый из моментов ее выполнения. Суммарное количество

*) Работа выполнена при финансовой поддержке Российского фонда фундаментальных исследований (код проекта 99–01–00601).

ресурса, требуемого всеми работами, выполняемыми в момент времени t ($t \geqslant 0$), не должно превосходить количества ресурса, имеющегося в наличии в момент t. Типичными представителями этого типа ресурсов являются производственные мощности, оборудование, людские ресурсы.

Невозобновимые ресурсы ограничиваются для всего проекта в целом, позволяя, например, моделировать бюджет проекта. Ограничение на ресурс невозобновимого типа есть ограничение на выбор тех или иных *мод* выполнения работ в мультимодальной постановке задачи. В одномодальной постановке данное ограничение не имеет смысла, если требуется выполнение **всех** работ проекта.*)

В отечественных работах [1–6] используется другая классификация. Ограниченные ресурсы делятся на складируемые (в [5] — сохраняемые) и нескладируемые. Понятие *нескладируемого* ресурса в точности совпадает с описанным выше понятием возобновимого ресурса. В то же время, хотя ресурсные ограничения *складируемого* типа также формулируются в виде баланса между потребляемым и выделяемым количеством ресурса, при этом используется другой механизм подсчета потребляемых и выделяемых ресурсов. А именно для каждой работы и ресурса k задается *интенсивность* $r_{jk}(t)$ потребления ресурса k работой j в момент t от начала ее выполнения. Для вычисления *объема* ресурса k, требуемого для выполнения работы j в течение времени τ, необходимо проинтегрировать функцию $r_{jk}(t)$ по t в интервале $[0, \tau]$. Аналогично для вычисления ресурса k, выделенного к моменту времени τ, необходимо проинтегрировать функцию $q_k(t)$ интенсивности выделения ресурса k в интервале $[0, \tau]$. Для допустимости расписания относительно складируемого ресурса k необходимо, чтобы для каждого момента времени $t \geqslant 0$ суммарный по всем работам объем ресурса, потребляемого к моменту t, не превосходил объема ресурса, выделяемого к моменту t. Примером складируемых ресурсов могут служить материалы с длительным сроком хранения.

Нетрудно заметить, что складируемые ресурсы невозможно учесть с помощью комбинации возобновимых и невозобновимых ресурсов. Более того, недавно введенное понятие *частично возобновимых ресурсов* [9], которое является обобщением возобновимых и невозобновимых ресурсов, также не позволяет описывать ограничения на складируемые ресурсы. В свою очередь, невозобновимый ресурс может рассматриваться как складируемый, весь объем которого доступен с начала выполнения

*) Возможны постановки задач, в которых требуется из заданного «портфеля заказов» выбрать оптимальное (по какому-то критерию) подмножество работ, удовлетворяющее, в числе прочих, ресурсным ограничениям невозобновимого типа. В таких задачах, как правило, вместо минимального срока завершения проекта рассматриваются другие критерии оптимальности.

проекта. (В этом случае ресурс как бы выделяется с ненулевой интенсивностью до момента начала выполнения проекта, и с нулевой интенсивностью после этого момента.)

Складируемые ресурсы интересны по меньшей мере по двум причинам. Во-первых, они более адекватно учитывают специфику некоторых ресурсов, обладающих свойством складируемости. Во-вторых, понятие складируемых ресурсов позволяет расширить возможности для приближенного решения «традиционных» задач календарного планирования, поскольку складируемые ресурсы являются ослаблением возобновимых. Из полученных ниже результатов, в частности, следует, что мы можем за полиномиальное время получать нижние оценки для ЗКП с возобновимыми ресурсами.

В работе [5] для ЗКП с ограничениями на складируемые ресурсы был описан алгоритм без обоснования оптимальности получаемого решения и анализа временной сложности. В [6] исследованы основные свойства расписаний со складируемыми ресурсами и на их основе построен алгоритм, из описания которого можно сделать вывод о его псевдополиномиальной временной сложности. В работах [1–4] для решения ЗКП с длительностями работ из $\mathbf{R}^+$, директивными сроками и ограничениями на складируемые ресурсы предложен асимптотически точный алгоритм, время работы которого зависит от числа дуг u в графе редукции частичного порядка как функция порядка $u \log u$, а абсолютная погрешность стремится к нулю с ростом размерности задачи.

В настоящей статье рассматривается дискретный аналог ЗКП, когда все работы имеют целочисленные длительности, а все функции, определяющие ресурсные ограничения (как-то: количество потребляемого каждой работой возобновимого ресурса, интенсивность потребления складируемого ресурса, а также функции количества и интенсивности выделяемых ресурсов), являются кусочно-постоянными, причем все интервалы постоянства этих функций являются целыми. Приводится полиномиальный алгоритм решения этой задачи в частном случае, когда в модели отсутствуют ресурсные ограничения возобновимого типа. (Такую задачу мы обозначаем PS^σ по аналогии с задачей PS^π, рассматриваемой в [9].) Для мультимодальной постановки этой задачи (MPS^σ) выделяется частный случай, когда построенный нами полиномиальный алгоритм находит оптимальное решение.

Статья организована следующим образом. В разд. 1 приводится математическая формулировка дискретной модели МЗКП, содержащая все упомянутые выше типы ресурсных ограничений. Формулируется также дискретный вариант ЗКП. (В этом случае невозобновимые ресурсы не накладывают никаких ограничений на выбор расписания.) В разд. 2

даются некоторые сведения из теории сетевого планирования, вводится понятие T-позднего расписания, приводятся основные свойства этих расписаний. В разд. 3 вводится в рассмотрение задача PS^σ и формулируется основной результат статьи: полиномиальный алгоритм решения задачи PS^σ. В разд. 4 приводится обоснование алгоритма решения задачи PS^σ на основе использования T-поздних расписаний и дается общая схема этого алгоритма. В разд. 5 описывается процедура проверки допустимости T-позднего расписания и анализируется ее временная сложность. Разд. 6 завершает доказательство основного результата. В разд. 7 приводится обоснование решения частного случая задачи MPS^σ.

1. Математическая модель

Приведем формальные постановки дискретных задач МЗКП и ЗКП. Рассмотрим проект, состоящий из множества работ $J = \{1, \ldots, n\}$. На множестве работ задан частичный порядок, обусловленный технологией реализации проекта. Предполагаем, что этот порядок задан орграфом $G = (V, U)$ (так называемым *графом редукции частичного порядка*), где $V = \{v_j \mid j = 1, \ldots, n\}$ — множество вершин графа, соответствующих работам $j \in J$, U — множество дуг графа, причем дуга (v_i, v_j) входит в U тогда и только тогда, когда работа i непосредственно предшествует работе j. Предполагается, что G не содержит контуров.

Для каждой работы $j \in J$ указаны определенные ресурсные требования. Будем различать три типа ресурсов в зависимости от типа обусловленных ими ограничений: *невозобновимые, возобновимые и складируемые*. Множества видов ресурсов каждого типа будем обозначать соответственно через $\mathscr{K}^\nu$, $\mathscr{K}^\rho$ и $\mathscr{K}^\sigma$.

Работа $j \in J$ может быть выполнена в одной из мод $m \in \{1, \ldots, \mathscr{M}_j\}$. Каждой моде m работы j соответствуют величина p_j^m продолжительности работы, потребность $r_{jk}^m(\tau) \geqslant 0$ в ресурсе $k \in \mathscr{K}^\rho \cup \mathscr{K}^\sigma$ в интервале $[\tau - 1, \tau)$ от момента начала выполнения работы ($\tau = 1, \ldots, p_j^m$) и потребность r_{jk}^m в невозобновимом ресурсе $k \in \mathscr{K}^\nu$.

Для каждого возобновимого ресурса $k \in \mathscr{K}^\rho$ суммарное по всем работам, выполняемым в интервале $[t-1, t)$, количество потребляемого ресурса k не может превосходить заданной величины $q_k(t)$. Для каждого складируемого ресурса $k \in \mathscr{K}^\sigma$ вид ограничения другой: суммарное по всем работам, выполняемым в интервале $[0, t)$, потребление ресурса k не может превосходить суммарного количества ресурса $\sum_{t'=1}^{t} q_k(t')$, выделенного к моменту t, где $q_k(t')$ — количество ресурса k, выделяемого в интервале $[t'-1, t')$. Наконец, для каждого невозобновимого ресурса

$k \in \mathscr{K}^{\nu}$ суммарная по всем работам потребность в этом ресурсе не может превосходить заданной величины q_k.

Для некоторых $j \in J_{\mathrm{dir}} \subseteq J$ задан директивный срок $d_j \in \mathbb{Z}^+$ завершения работы j.

Под *допустимым* расписанием S понимается такое назначение для каждой работы $j \in J$ неотрицательного целочисленного момента s_j начала ее выполнения и такой моды $m(j)$, что выполняются, во-первых, описанные выше ресурсные ограничения, во-вторых, ограничения, обусловленные отношением предшествования: для любых работ i и j таких, что $(v_i, v_j) \in U$, выполняется соотношение $s_i + p_i^{m(i)} \leqslant s_j$; в-третьих, соблюдаются директивные сроки: $s_j + p_j^{m(j)} \leqslant d_j$ для всякой работы $j \in J_{\mathrm{dir}}$. Прерывания работ не разрешаются. Задача заключается в нахождении допустимого расписания с минимальным временем завершения проекта $C_{\max}(S)$.

Для каждой работы j будем считать известным множество $\mathrm{Pred}(j)$ ее непосредственных предшественников, т. е. считаем, что граф G задан набором списков дуг, входящих в каждую вершину $v_j \in V$, $j = 1, \ldots, n$. Через $\mathrm{Succ}(j) = \{j' \in J \mid j \in \mathrm{Pred}(j')\}$ будем обозначать множество *непосредственных последователей* работы $j \in J$. Определим также множества работ $J_{\min} = \{j \in J \mid \mathrm{Pred}(j) = \varnothing\}$ и $J_{\max} = \{j \in J \mid \mathrm{Succ}(j) = \varnothing\}$.

Формально рассматриваемая задача может быть записана следующим образом: минимизировать время завершения проекта

$$C_{\max}(S) = \max_{j \in J}(s_j + p_j^{m(j)}) \tag{1}$$

при условиях

$$s_j + p_j^{m(j)} \leqslant d_j, \quad j \in J_{\mathrm{dir}}; \tag{2}$$

$$s_i + p_i^{m(i)} \leqslant s_j, \quad i \in \mathrm{Pred}(j),\ j \in J; \tag{3}$$

$$\sum_{j \in J(t)} r_{jk}^{m(j)}(t - s_j) \leqslant q_k(t), \quad k \in \mathscr{K}^{\rho},\ t \in \mathbb{Z}^+ \setminus \{0\}; \tag{4}$$

$$\sum_{t'=1}^{t} \sum_{j \in J(t')} r_{jk}^{m(j)}(t' - s_j) \leqslant \sum_{t'=1}^{t} q_k(t'), \quad k \in \mathscr{K}^{\sigma},\ t \in \mathbb{Z}^+ \setminus \{0\}; \tag{5}$$

$$\sum_{j \in J} r_{jk}^{m(j)} \leqslant q_k, \quad k \in \mathscr{K}^{\nu}; \tag{6}$$

$$s_j \in \mathbb{Z}^+, \quad j \in J; \tag{7}$$

$$m(j) \in \{1, \ldots, \mathscr{M}_j\}, \quad j \in J. \tag{8}$$

Здесь $J(t) = \{j \mid s_j < t \leqslant s_j + p_j^{m(j)}\}$ — множество работ, выполняемых в единичном интервале $[t-1, t)$ при расписании S; $\mathbb{Z}^+ = \{0, 1, 2, \ldots\}$.

Множество ограничений (2) гарантирует соблюдение директивных сроков. Неравенства (3) соответствуют ограничениям предшествования. Соотношения (4)–(6) обеспечивают соблюдение ограничений по возобновимым, складируемым и невозобновимым ресурсам соответственно. Условия (7)–(8) отражают требования на переменные s_j и $m(j)$.

Нетрудно заметить, что ресурсные ограничения (4)–(6) расположены в порядке убывания их «строгости». Ограничение (4) является наиболее сильным в том смысле, что если расписание допустимо относительно некоторого возобновимого ресурса, то оно остается допустимым, если этот ресурс перевести в разряд складируемых или невозобновимых. Аналогичное утверждение верно для складируемых ресурсов по отношению к невозобновимым. Таким образом, складируемые ресурсы занимают промежуточное положение между возобновимыми и невозобновимыми.

Далее, кроме последнего раздела в статье будет рассматриваться одномодальный случай, когда $|\mathscr{M}_j| = 1$ для всех работ j, поэтому индекс $m(j)$ в соответствующих обозначения будет опускаться. В этом случае можно считать, что задача не содержит ограничений на невозобновимые ресурсы, поскольку проверка допустимости решения по ним (неравенство (6)) не зависит от расписания S. Задача записывается следующим образом: минимизировать время завершения проекта

$$C_{\max}(S) = \max_{j \in J}(s_j + p_j) \tag{9}$$

при условиях

$$s_j + p_j \leqslant d_j, \quad j \in J_{\text{dir}}; \tag{10}$$

$$s_i + p_i \leqslant s_j, \quad i \in \text{Pred}(j),\ j \in J; \tag{11}$$

$$\sum_{j \in J(t)} r_{jk}(t - s_j) \leqslant q_k(t), \quad k \in \mathscr{K}^{\rho},\ t \in \mathbb{Z}^+ \setminus \{0\}; \tag{12}$$

$$\sum_{t'=1}^{t} \sum_{j \in J(t')} r_{jk}(t' - s_j) \leqslant \sum_{t'=1}^{t} q_k(t'), \quad k \in \mathscr{K}^{\sigma},\ t \in \mathbb{Z}^+ \setminus \{0\}; \tag{13}$$

$$s_j \in \mathbb{Z}^+, \quad j \in J. \tag{14}$$

По классификации, предложенной в [10], сформулированная задача обозначается как $PS \mid prec \mid C_{\max}$. Поскольку эта задача является обобщением известной задачи job-shop scheduling, она принадлежит к классу NP-трудных задач [8]. Вместе с тем, как будет показано далее, при отсутствии ограничений на возобновимые ресурсы нахождение точного решения сформулированной задачи возможно за полиномиальное время.

2. Некоторые сведения из теории сетевого планирования и T-поздние расписания

Мы будем опираться на некоторые сведения, понятия и результаты из теории сетевого планирования (см., например, книгу [7] и приведенную в ней библиографию).

Прежде всего заметим, что по заданному набору $\{\mathrm{Pred}(j) \mid j \in J\}$ списков работ, непосредственно предшествующих работе j, можно вычислить набор $\{\mathrm{Succ}(j) \mid j \in J\}$ списков работ, непосредственно следующих за работой j, а также множества $J_{\min}$ и $J_{\max}$. Временна́я сложность соответствующих вычислений не превышает величины $O(u)$, где u — число дуг в графе G.

Важную роль в сетевом планировании играют так называемые наиболее ранние и наиболее поздние расписания.

Наиболее раннее расписание $S' = \{s'_j \mid j \in J\}$ учитывает только ограничения (11), (14) и определяется соотношениями

$$s'_j = \begin{cases} 0, & \text{если } j \in J_{\min}; \\ \max\limits_{i \in \mathrm{Pred}(j)} (s'_i + p_i), & \text{если } j \in J \setminus J_{\min}. \end{cases}$$

Длину наиболее раннего расписания $T_{\text{кр}} = C_{\max}(S')$ называют *критическим временем* проекта. Длину оптимального расписания S_{opt} (если оно существует) обозначим через $C_{\text{opt}} = C_{\max}(S_{\text{opt}})$. Для сформулированной выше задачи (9)–(14) критическое время $T_{\text{кр}}$ является тривиальной нижней оценкой длины C_{opt} оптимального расписания. Вычисление критического времени проекта $T_{\text{кр}}$ может быть проведено с использованием известных идей поиска в глубину или обхода лабиринта. Одна из реализаций таких алгоритмов приведена в [2].

Утверждение 1. *Построение наиболее раннего расписания S' и нахождение его длины $T_{\text{кр}}$ могут быть выполнены за время $O(u)$.*

Расписание назовем *полудопустимым*, если оно допустимо относительно ограничений (10), (11), (14) (т. е. без учета ресурсных ограничений), и *допустимым*, если оно удовлетворяет всем ограничениям (10)–(14).

Ясно, что полудопустимое расписание существует тогда и только тогда, когда наиболее раннее расписание S' удовлетворяет ограничениям (11) (т. е. допустимо по директивным срокам).

Ниже мы будем пользоваться следующим понятием T-позднего расписания. Пусть $T \geqslant T_{\text{кр}}$ — целое неотрицательное число. Полудопустимое расписание $S^T = \{s_j^T \mid j \in J\}$ назовем *T-поздним*, если

$$s_j^T + p_j \leqslant T, \quad j \in J, \tag{15}$$

и ни один из моментов s_j^T не может быть увеличен без нарушения условий полудопустимости или условия (15).

Для заданного T и каждой работы $j \in J$ определим *T-зависимый директивный срок* d_j^T:

$$d_j^T = \begin{cases} \min\{d_j, T\}, & \text{если } j \in J_{\text{dir}}; \\ T, & \text{если } j \in J \setminus J_{\text{dir}}. \end{cases} \tag{16}$$

Утверждение 2. *Если наиболее раннее расписание S' полудопустимо и $T_{\text{кр}}$ — его длина, то для любого $T \geqslant T_{\text{кр}}$ соответствующее T-позднее расписание $S^T = \{s_j^T\}$ существует, единственно и определяется соотношениями*

$$s_j^T = \begin{cases} d_j^T - p_j, & \text{если } j \in J_{\max}; \\ \min\left\{d_j^T, \min\limits_{i \in \text{Succ}(j)} s_i^T\right\} - p_j, & \text{если } j \in J \setminus J_{\max}. \end{cases} \tag{17}$$

И наоборот: если расписание S' не является полудопустимым, то ни для какого $T \geqslant T_{\text{кр}}$ не существует T-позднего расписания S^T.

Поскольку для всякого $T \geqslant T_{\text{кр}}$ существует не более одного T-позднего расписания, соотношения (17) можно принять за определение T-позднего расписания при условии, что все значения s_j^T неотрицательны. Из (16) и (17) вытекает

Утверждение 3. *Пусть $T_{\text{кр}} \leqslant T_1 \leqslant T_2$ и расписания $S^{T_1} = \{s_j^{T_1}\}$, $S^{T_2} = \{s_j^{T_2}\}$ существуют. Тогда $s_j^{T_1} \leqslant s_j^{T_2}$ для любого $j \in J$.*

Утверждение 4. *Для любого полудопустимого расписания $S = \{s_j\}$ длины $C_{\max}(S) = T$ выполняются соотношения $s_j \leqslant s_j^T$, $j \in J$.*

Утверждение 5. *Если для $T \geqslant T_{\text{кр}}$ существует T-позднее расписание $S^T = \{s_j^T\}$, то оно может быть найдено за время $O(n)$.*

Реализацию такого алгоритма можно найти, например, в [2].

3. Задача PS^σ и формулировка основного результата

Далее будем рассматривать задачу (9)–(14), не содержащую ограничений (12) на возобновимые ресурсы. (Напомним, что такую задачу мы обозначаем через PS^σ.) Поскольку в формулировке основного результата участвует оценка временной сложности алгоритма, необходимо уточнить, в каком виде предполагаются представленными исходные данные задачи PS^σ.

Директивные сроки работ предполагаем заданными в виде списка J_{dir} работ (в произвольном порядке), для которых назначены директивные сроки. Ограничения предшествования на множестве работ заданы

в виде семейства списков $\{\mathrm{Pred}(j) \mid j \in J\}$ работ, непосредственно предшествующих работе $j \in J$. (Что эквивалентно заданию графа $G = (V, U)$ редукции частичного порядка на множестве вершин-работ в виде списков входящих дуг, заданных для каждой вершины $v \in V$.) Наконец, функции $r_{jk}(t)$ (интенсивности потребления работой $j \in J$ складируемого ресурса $k \in \mathscr{K}^\sigma$) и функции $q_k(t)$ (интенсивности выделения ресурса $k \in \mathscr{K}^\sigma$) предполагаем неотрицательными кусочно-постоянными с **произвольными** целыми длинами интервалов постоянства. Будем предполагать, что для каждой работы $j \in J$ и каждого ресурса $k \in \mathscr{K}^\sigma$ задана последовательность пар (r^i_{jk}, p^i_{jk}), $i = 1, \ldots, N_{jk}$, где r^i_{jk} — значение функции $r_{jk}(t)$ в i-м интервале постоянства, а p^i_{jk} — длина этого интервала. Аналогично каждую функцию $q_k(t), k \in \mathscr{K}^\sigma$ предполагаем заданной последовательностью пар (q^i_k, ρ^i_k), $i = 1, \ldots, N'_k$, где q^i_k — значение функции в i-м интервале ее постоянства, а ρ^i_k — длина этого интервала. (Длина последнего интервала может быть, в принципе, неограниченной.) Такой способ представления информации о ресурсах является, с нашей точки зрения, достаточно универсальным (для дискретной постановки задачи) и в то же время позволяет наиболее гибко учитывать специфику той или иной индивидуальной задачи.

Из утверждения 3, вида ограничений (13) и неотрицательности функций $q_k(t)$ следует

Утверждение 6. *Пусть при некотором T_1 расписание S^{T_1} является допустимым решением задачи PS^σ. Тогда для любого $T_2 \geqslant T_1$ расписание S^{T_2} также допустимо.*

Из утверждения 4 и вида ограничений (13) следует

Утверждение 7. *Пусть при некотором T в задаче PS^σ существует допустимое расписание S длины T. Тогда T-позднее расписание является допустимым.*

Отсюда следует, что C_{opt}-позднее расписание также оптимально. Это позволяет искать оптимальное решение задачи PS^σ в классе T-поздних расписаний. Другими словами, для нахождения оптимального расписания (в случае, когда такое расписание существует) достаточно определить минимально возможное значение T, при котором соответствующее T-позднее расписание существует и допустимо по ресурсным ограничениям. Понятно, что в этом случае минимально возможное T совпадает с величиной C_{opt}. Теперь сформулируем основной результат статьи.

Теорема 1. *Решение задачи PS^σ может быть получено с временной сложностью $O\Big(|J_{\mathrm{dir}}| \log N_d + (N(\log f + \log K) + u + N')\widehat{D}\Big)$, где f — ширина частичного порядка, u — число дуг в графе редукции частичного*

порядка, K — число видов ресурсов, J_{dir} — множество работ с директивными сроками, N_d — число различных директивных сроков, N' — суммарное по всем ресурсам число интервалов постоянства функций интенсивности выделения ресурсов, N — суммарное по всем работам и всем ресурсам число интервалов постоянства функций интенсивности потребления ресурсов, $\widehat{D}$ — длина двоичной записи наибольшего директивного срока.

По поводу формулировки теоремы 1 необходимо сделать следующее

Замечание 1. Вместо величины $\widehat{D}$ в формулировках теоремы 1 мы могли бы использовать более привычную функцию $\log D$, где D — наибольший директивный срок. Однако в этом случае возникают формальные проблемы при $D = 1$ и $J_{\text{dir}} = \varnothing$. Теперь же, полагая $D = \max\{d_j \mid j \in J_{\text{dir}}\} = 0$ при $J_{\text{dir}} = \varnothing$, имеем $\widehat{D} = 1$ (как и в случае, когда $D = 1$).

4. Обоснование алгоритма решения задачи PS^σ на основе T-поздних расписаний

Далее K будет означать число видов ресурсов складируемого типа: $K = |\mathscr{K}^\sigma|$.

Определим функцию $Q_k(t)$ — суммарное количество ресурса $k \in \mathscr{K}^\sigma$, выделяемого в интервале $[0, t)$. После предварительно вычисления (за время $O(N'_k)$) значений функции $Q_k(t)$ в «узловых» точках $t^i_k = \rho^1_k + \ldots + \rho^i_k$, где i пробегает значения $1, \ldots, N'_k$, значения в промежуточных точках можно вычислять за константное время (в силу линейности этой функции на каждом интервале постоянства функции интенсивности выделения ресурса).

Условимся исключить из дальнейшего рассмотрения два тривиальных случая, когда задача PS^σ не имеет решения. Это, во-первых, случай, когда наиболее раннее расписание S' (длины $T_{\text{кр}}$) не является полудопустимым (т. е. не удовлетворяет директивным срокам) и согласно утверждению 2 ни для какого $T \geqslant T_{\text{кр}}$ не существует расписания длины T. Во-вторых, это случай, когда хотя бы для одного вида $k \in \mathscr{K}^\sigma$ суммарный объем ресурсов

$$R_{k,\Sigma} = \sum_{j \in J} \sum_{i=1}^{N_{jk}} r^i_{jk} p^i_{jk},$$

требуемых для выполнения работ проекта, строго превышает суммарный объем выделяемых ресурсов.

Исключение из рассмотрения первого случая означает (согласно утверждению 2), что для любого $T \geqslant T_{\text{кр}}$ существует T-позднее расписание длины S^T.

Исключение второго случая означает существование конечного момента времени

$$T_\Sigma = \max_{k \in \mathscr{K}^\sigma} \min\{T \in \mathbb{Z}^+ \mid R_{k,\Sigma} \leqslant Q_k(T)\},$$

который является еще одной тривиальной нижней оценкой длины C_{opt} оптимального расписания. Поэтому далее в качестве нижней оценки длины оптимального расписания будем использовать величину $T' = \max\{T_{\text{кр}}, T_\Sigma\}$.

Легко видеть, что проверка полудопустимости раннего расписания осуществляется за время $O(u)$, а вычисление параметра T_Σ — за время $O(N + N')$.

Введем еще несколько обозначений, используемых для обоснования алгоритма решения задачи PS^σ. Для $T \geqslant T'$ определим функцию $R_k^T(t)$ — суммарное количество ресурса k, потребляемое работами проекта в интервале $[0, t)$ при расписании S^T.

Определим следующие величины: $t_{\min}^T$ — минимальное целое значение t, для которого T-позднему расписанию не хватает ресурсов в интервале $[0, t)$:

$$t_{\min}^T = \min\{t \mid Q_k(t) < R_k^T(t);\ 1 \leqslant t \leqslant T;\ k \in \mathscr{K}^\sigma\},$$

при этом полагаем $t_{\min}^T = T + 1$, если множество таких t пусто; $\Delta(T)$ — минимальное целое значение, на которое нужно увеличить моменты начала всех работ в T-позднем расписании, чтобы выполнились все ограничения (кроме, быть может, ограничений на директивные сроки):

$$\Delta(T) = \min\{\Delta \in \mathbb{Z}^+ \mid R_k^T(t) \leqslant Q_k(t + \Delta);\ t = 1, \ldots, T;\ k \in \mathscr{K}^\sigma\}.$$

Из очевидных соотношений

$$\Delta(T) = \min\{\Delta \in \mathbb{Z}^+ \mid R_k^T(t) \leqslant Q_k(t + \Delta);\ t = 1, \ldots, T;\ t \leqslant T_\Sigma;\ k \in \mathscr{K}^\sigma\};$$
$$R_k^T(t) \leqslant R_k^T(T_\Sigma) \leqslant Q_k(T_\Sigma) \leqslant Q_k(t + T_\Sigma) \text{ для всяких } t = 1, \ldots, T_\Sigma;\ k \in \mathscr{K}^\sigma,$$

имеем неравенство $\Delta(T) \leqslant T_\Sigma$, влекущее конечность $\Delta(T)$ при конечном T_Σ.

Непосредственно из определения параметров $t_{\min}^T$ и $\Delta(T)$ следует, что $t_{\min}^T > T$ тогда и только тогда, когда $\Delta(T) = 0$, откуда вытекает используемый далее критерий проверки допустимости расписания S^T:

Утверждение 8. *T-позднее расписание S^T допустимо тогда и только тогда, когда $\Delta(T) = 0$.*

Все работы из множества J разделим на две категории. Первая категория — это множество J' «директивно-зависимых» работ, в которое

входят как работы из J_{dir}, так и все предшествующие им работы. Вторая категория состоит из множества J'' «свободных» работ с потенциально неограниченным сроком их начала в T-позднем расписании (при неограниченном увеличении параметра T).

Обозначим $T'' = D + T_{\text{кр}}$, где D — наибольший директивный срок; $t_{\min} = t^{T''}_{\min}$; $\Delta' = \min\{\Delta \in \mathbb{Z}^+ \mid R^{T''}_k(t) \leqslant Q_k(t+\Delta);\ t = 1, \ldots, D; k \in \mathscr{K}^\sigma\}$; $\Delta'' = \Delta(T'')$. Очевидно, что $\Delta' \leqslant \Delta''$.

Напомним, что в качестве нижней оценки для C_{opt} взята величина $T' = \{T_{\text{кр}}, T_\Sigma\}$.

Центральное место в обосновании предлагаемого ниже алгоритма решения задачи PS^σ занимает следующая

Теорема 2. *В зависимости от параметров Δ' и Δ'' решение задачи PS^σ разбивается на три качественно различных случая.*

1. *При $\Delta' > 0$ задача PS^σ не имеет решения.*

2. *При $\Delta' = 0$ и $\Delta'' > 0$ расписание $S_{\text{opt}} = \{s_j \mid j \in J\}$ может быть получено из T''-позднего расписания увеличением моментов начала всех работ из множества J'' на величину Δ'' :*

$$s_j = \begin{cases} s_j^{T''}, & \text{если } j \in J'; \\ s_j^{T''} + \Delta'', & \text{если } j \in J''. \end{cases} \tag{18}$$

При этом длина оптимального расписания равна $C_{\text{opt}} = T'' + \Delta''$.

3. *При $\Delta'' = 0$ оптимальное расписание S_{opt} имеет длину $C_{\text{opt}} \in \{T', \ldots, T''\}$ и может быть найдено за $O(\widehat{D})$ итераций, на каждой из которых осуществляются вычисление T-позднего расписания для некоторого $T \in \{T', \ldots, T''\}$ и проверка его допустимости.*

Доказательство. Проиллюстрируем три главных случая теоремы 2 (рис. 1–3). Рассмотрим каждый случай.

Случай 1 ($\Delta' > 0$).

Очевидно, что при любом значении T работы из J' заканчиваются в расписании S^T не позднее момента D. При выборе значения $T'' = D + T_{\text{кр}}$ работы из J' и J'' в расписании $S^{T''}$ четко разделяются моментом D: все работы из J' заканчиваются не позднее D, а все работы из J'' начинаются не раньше D. Следовательно, в интервале $[0, D)$ функция $R^{T''}_k(t)$ потребления ресурса k при расписании $S^{T''}$ целиком определяется работами из множества J', а в интервале $[D, T'')$ — работами из множества J''. Очевидно также, что на работах из J' расписание $S^{T''}$ совпадает с D-поздним расписанием для этих работ. А поскольку D-позднее расписание (для работ из J') недопустимо по ресурсам, то из утверждения 7 следует, что для работ из J' не существует допустимого по ресурсам расписания длины D. В то же время любое расписание большей длины для работ из J' будет недопустимо по директивным срокам,

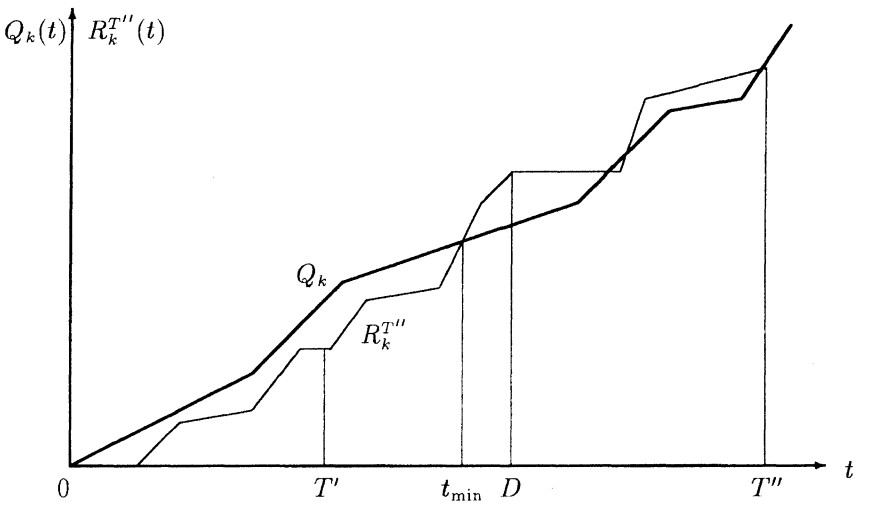

Рис. 1. Случай $\Delta' > 0$: допустимого расписания не существует

а значит, допустимого расписания для работ из J' (и, следовательно, для всего множества работ J) не существует.

Случай 2 ($\Delta' = 0$ и $\Delta'' > 0$).

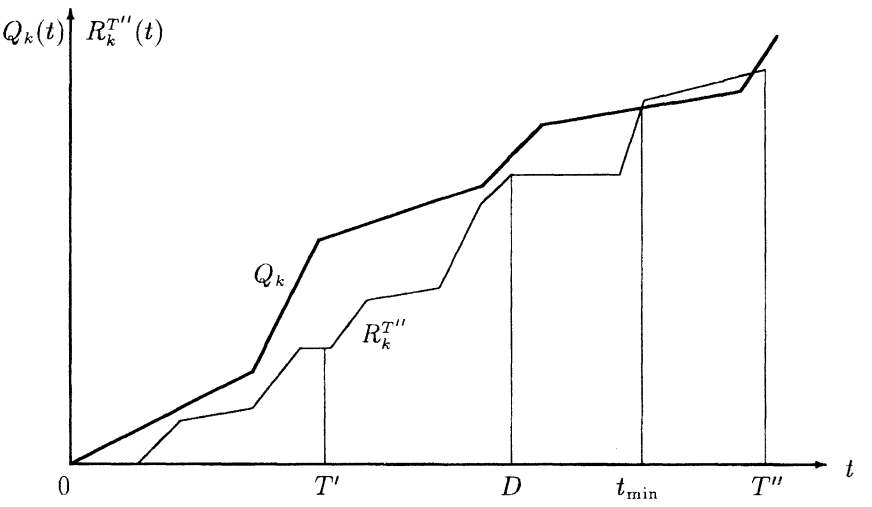

Рис. 2. Случай $C_{\text{opt}} > T''$

Обозначим $T^* = T'' + \Delta''$. Справедливость утверждения теоремы вытекает из двух фактов: из допустимости расписания S^*, определяемого соотношениями (18), и из несуществования допустимого расписания $S^{\widetilde{T}}$ ни при каком $\widetilde{T} < T^*$.

Докажем первое. Из $\Delta' = 0$ следует, что T''-позднее расписание $S^{T''} = \{s_j^{T''} \mid j \in J\}$ существует. Поскольку при этом величина параметра Δ'' конечна, то все величины в правой части (18) определены и расписание $S^* = \{s_j\}$ определено. Замечаем, что S^* есть не что иное, как T^*-позднее расписание (это следует из (16)–(18)). Допустимость S^* относительно директивных сроков (для всех работ $j \in J_{\text{dir}} \subseteq J'$) следует из полудопустимости расписания $S^{T''}$ (откуда S^* наследует значения s_j для работ $j \in J'$).

Для доказательства допустимости S^* по ресурсным ограничениям достаточно понять, как выглядит график функции $R_k^{T^*}(t)$ ($k \in \mathscr{K}^\sigma$). Из разделенности работ из J' и J'' барьером D в расписании $S^{T''}$ следует, что график функции $R_k^{T^*}(t)$ совпадает с графиком функции $R_k^{T''}(t)$ в интервале $[0, D)$ и, следовательно, допустим относительно ресурсных ограничений. В интервале $[D, D + \Delta'']$ функция $R_k^{T^*}(t)$ постоянна (и равна $R_k^{T''}(D)$) и также допустима по отношению к функции $Q_k(t)$. Наконец, график $R_k^{T^*}(t)$ в интервале $[D + \Delta'', T'' + \Delta'']$ есть график функции $R_k^{T''}(t)$ в интервале $[D, T'']$, сдвинутый вправо на величину параметра Δ''. По определению этого параметра такой сдвиг графика обеспечивает его допустимость.

Покажем теперь, что для любого (целого) $\widetilde{T} < T^*$ расписание $S^{\widetilde{T}}$ недопустимо по ресурсам (а следовательно, по утверждению 7, допустимого расписания длины $\widetilde{T}$ не существует ни при каком $\widetilde{T} < T^*$, откуда следует оптимальность расписания S^{T^*}).

Действительно, недопустимость $\widetilde{T}$-позднего расписания $S^{\widetilde{T}}$ при любом $\widetilde{T} \leqslant T''$ следует из недопустимости $S^{T''}$ и утверждения 6. При любом $\widetilde{T} > T''$, как отмечалось выше, график функции $R_k^{\widetilde{T}}(t)$ в интервале $[D + \widetilde{T} - T'', \widetilde{T}] = [\widetilde{T} - T', \widetilde{T}]$ совпадает с графиком функции $R_k^{T''}(t)$ в интервале $[D, T'']$, сдвинутом вправо на величину $\Delta' = \widetilde{T} - T''$. Если $\widetilde{T} < T^*$, то $\Delta' < T^* - T'' = \Delta''$. По определению параметра $\Delta'' = \Delta(T'')$ найдутся $k \in \mathscr{K}^\sigma$ и $t \leqslant T''$ такие, что $R_k^{T''}(t) > Q_k(t + \Delta')$, а следовательно, $\widetilde{T}$-позднее расписание $S^{\widetilde{T}}$ недопустимо по ресурсу k.

Случай 3. ($\Delta'' = 0$).

Согласно утверждению 8 при $\Delta(T'') = \Delta'' = 0$ расписание $S^{T''}$ допустимо; следовательно, длина C_{opt} оптимального расписания принадлежит интервалу $[T', T'']$. По утверждению 7 C_{opt} совпадает с минимальным значением T, при котором расписание S^T допустимо. Из утверждения 6 следует, что для нахождения такого минимального T можно использовать дихотомический поиск в интервале $[T', T'']$ с помощью следующей процедуры.

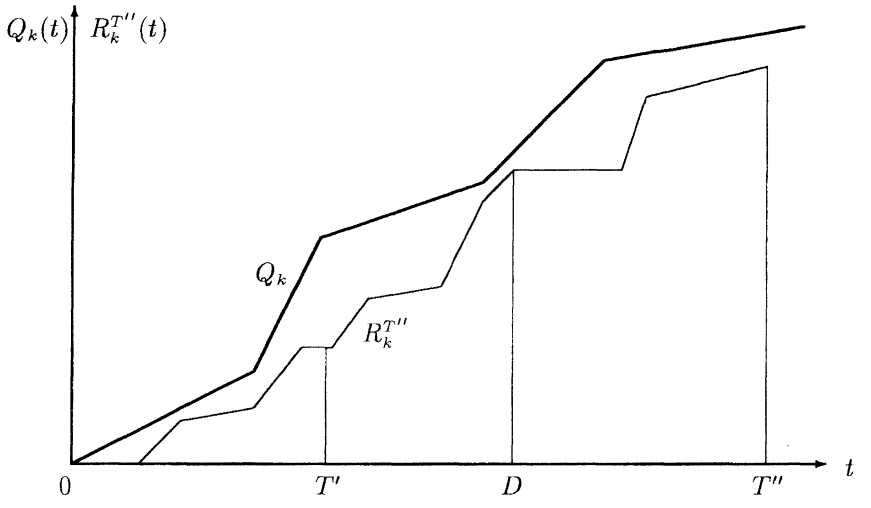

Рис. 3. Случай $T' \leqslant C_{\text{opt}} \leqslant T''$

Процедура P_0.

1. Считая известными параметры T', T'' и T''-позднее расписание $S^{T''}$, положить $T_{\text{ниж}} := T'$ и $T_{\text{верх}} := T''$ в качестве нижней и верхней границ интервала неопределенности для значения C_{opt}. При этом известно, что расписание $S^{T_{\text{верх}}}$ допустимо.

2. Положить $T := \lfloor (T_{\text{ниж}} + T_{\text{верх}})/2 \rfloor$.

3. Если $T = T_{\text{верх}}$, то получено оптимальное расписание $S_{\text{opt}} = S^T$ с длиной $C_{\text{opt}} = T_{\text{верх}}$. Стоп.

4. Если $T < T_{\text{верх}}$, то построить T-позднее расписание S^T.

5. Если S^T допустимо, то уменьшить верхнюю границу: $T_{\text{верх}} := T$, иначе увеличить нижнюю границу: $T_{\text{ниж}} := T + 1$. В обоих случаях вернуться на п. 2.

Процедура P_0 описана полностью.

Нетрудно заметить, что построение каждого T-позднего расписания для некоторого $T \in [T', T'']$ в п. 4 и проверка его допустимости в п. 5 сопровождаются уменьшением разности $T_{\text{верх}} - T_{\text{ниж}}$ как минимум вдвое. Таким образом, не более чем за $\log_2(T'' - T') \leqslant \log_2 D$ итераций цикла по п. 2 эта разность достигнет величины $T_{\text{верх}} - T_{\text{ниж}} = 1$, после чего на следующей итерации станет равной нулю. Теорема 2 доказана.

Из теоремы 2 вытекает следующий алгоритм решения задачи PS^σ.

Алгоритм $\mathscr{A}$.

Этап 1. Вычисляем критическое время $T_{\text{кр}}$, параметр T_Σ и нижнюю оценку длины оптимального расписания $T' = \max\{T_{\text{кр}}, T_\Sigma\}$. (Напомним,

что в начале разд. 4 исключены из рассмотрения два тривиальных случая отсутствия допустимых расписаний.)

Этап 2. Находим T''-позднее расписание $S^{T''}$, где $T'' = T_{\text{кр}} + D$. (Если $J_{\text{dir}} = \varnothing$, то $D = 0$.) Вычисляем значения Δ' и Δ''. Если $\Delta' > 0$, то Стоп (задача не имеет решения). В противном случае:

— если $\Delta'' > 0$, то переходим к этапу 3,

— иначе ($\Delta'' = 0$) переходим к этапу 4.

Этап 3. Оптимальное расписание S_{opt} определяется соотношениями (18). Стоп.

Этап 4. Оптимальное расписание S_{opt} находится с помощью процедуры P_0. Стоп.

Алгоритм $\mathscr{A}$ описан.

Далее для доказательства теоремы 1 необходимо вывести оценки временной сложности алгоритма проверки допустимости T-позднего расписания по ресурсным ограничениям и алгоритма вычисления $\Delta(T)$.

5. Алгоритм проверки допустимости T-позднего расписания в задаче PS^{σ}

В процедуре P_0 при различных значениях T требуется построить T-позднее расписание и проверить его допустимость относительно каждого ресурсного ограничения. При проверке допустимости расписания S^T воспользуемся критерием в форме утверждения 8, в котором фигурирует величина $\Delta(T)$, для вычисления которой необходимо знать функции объемов выделенного $Q_k(t)$ и потребляемого $R_k^T(t)$ (при расписании S^T) количества ресурса k. Поскольку обе функции кусочно-линейны, то они полностью определяются значениями в своих *узлах* (т. е. в точках, где происходит изменение линейного коэффициента функции). Таким образом, необходимо найти все узлы функций $Q_k(t)$ и $R_k^T(t)$, $k \in \mathscr{K}^{\sigma}$, после чего становится возможным вычисление параметра $\Delta(T)$ и установление допустимости расписания S^T.

Отыскание узлов функции $R_k^T(t)$

Опишем алгоритм $\mathscr{A}_R(T)$ отыскания убывающей по времени последовательности $y_k^0, y_k^1, \ldots$ узлов функции $R_k^T(t)$ для каждого $k \in \mathscr{K}^{\sigma}$. (При этом узлами функции $R_k^T(t)$ будем считать все точки, в которых происходит изменение интенсивности потребления ресурса k хотя бы одной из работ, а также точки, совпадающие с директивными сроками.) Попутно в алгоритме будет построено T-позднее расписание S^T, так что его предварительного построения не требуется.

Узел y_k^i будет отождествляться с парой $(t, R) = (t(y_k^i), R(y_k^i))$, где t и R — значения t и $R_k^T(t)$ в узле y_k^i. Информацию о директивных сроках

будем считать заданной в виде упорядоченного по возрастанию списка $\mathscr{D}$ **различных** директивных сроков. (Такой список может быть построен на этапе 2 алгоритма $\mathscr{A}$ по списку директивных сроков, заданных для всех работ $j \in J_{\mathrm{dir}}$.) При каждом директивном сроке $d \in \mathscr{D}$ имеется список работ $J(d)$, чей директивный срок d_j совпадает с d.

Алгоритм $\mathscr{A}_R(T)$.

Шаг 0. Для каждого ресурса $k \in \mathscr{K}^\sigma$ определить исходный узел $y_k^0 = (T, R(y_k^0))$, где $R(y_k^0) = R_{k,\Sigma}$ — суммарная по всем работам $j \in J$ потребность в ресурсе k; через $\mu(k)$ обозначаем номер последнего найденного узла функции $R_k^T(t)$, исходно полагаем $\mu(k) = 0$. В списке $\mathscr{D}$ найти максимальный директивный срок d^* такой, что $d^* < T$; переменной μ_d присвоить номер этого директивного срока в списке $\mathscr{D}$. Если множество таких директивных сроков пусто, то положить $d^* := 0$, $\mu_d := 0$.

Каждой работе $j \in J$ ставим в соответствие переменную c_j, отражающую состояние этой работы в алгоритме: $c_j \in \{1, \ldots, |\mathrm{Succ}(j)| + 1\}$ означает, что осталось $c_j - 1$ непросмотренных работ $j' \in \mathrm{Succ}(j)$; $c_j = 0$ означает, что работа включена во фронт, но ее просмотр еще не начался; $c_j \in \{-K, \ldots, -1\}$ означает, что по $-c_j$ ресурсам она уже просмотрена ($c_j = -K$ означает, что она просмотрена полностью). Исходно полагаем $c_j := |\mathrm{Succ}(j)| + 1$, $j \in J$. Определяем также глобальный счетчик C непросмотренных работ: исходно $C = n$.

Формируем фронт F текущих работ. На шаге 0 в него войдут все работы $j \in J_{\max}$ такие, что $d_j^T = T$ (при этом полагаем $c_j = 0$). Для каждой работы $j \in F$ и ресурса $k \in \mathscr{K}^\sigma$ формируем пятерку $h^{jk} = (j, k, i, r, \tau)$, где i — номер текущего интервала постоянства функции $r_{jk}(t)$, r — значение функции $r_{jk}(t)$ в i-м интервале, τ — момент начала i-го интервала в строящемся расписании S^T. Исходно для каждого $j \in F$ и $k \in \mathscr{K}^\sigma$ полагаем $i(h^{jk}) = N_{jk}$, $r(h^{jk}) = r_{jk}^i$, $\tau(h^{jk}) = T - p_{jk}^i$. Множество пятерок $H = \{h^{jk} \mid j \in F,\ k \in \mathscr{K}^\sigma\}$ организуем в виде «кучи» (т. е. двоичного дерева, полуупорядоченного по ключу τ: в корневой вершине этого дерева находится пятерка с максимальным значением τ^* ключа τ, а по любой цепочке, идущей из корня в один из «листьев», значение τ монотонно не возрастает; при этом длина такой цепочки не превышает $O(\log|H|)$; согласно Тарьяну [12] такая куча H может быть построена за время $O(|H|)$). Если $H = \varnothing$, положить $\tau^* = 0$. Для каждого $k \in \mathscr{K}^\sigma$ вычислить $\bar{r}_k = \sum\limits_{j \in F} r(h^{jk})$ — текущую суммарную интенсивность потребления ресурса k.

Шаг ν ($\nu = 1, 2, \ldots$). Находим $t^* = \max\{\tau^*, d^*\}$. Если $t^* = \tau^*$, то выполнить процедуру A. Если $t^* = d^*$, то выполнить процедуру B. (Возможно выполнение обеих процедур.)

Процедура А.

Пусть $\hat{h} = (j, k, i, r, \tau)$ — пятерка, соответствующая корневому элементу кучи H, $\tau = \tau^*$. Удалить $\hat{h}$ из H и перестроить кучу, с тем чтобы элемент с максимальным значением τ (определяющим новое значение τ^*) вновь оказался в вершине кучи. (Согласно Тарьяну [12] такая перестройка выполняется за время $O(\log_2 |H|)$.) Если $H = \varnothing$, то полагаем $\tau^* = 0$.

При $t^* < t(y_k^{\mu(k)})$, т. е. при обнаружении нового узла функции $R_k^T(t)$, вычислить значение функции в этом узле по формуле

$$R = R(y_k^{\mu(k)}) - (t(y_k^{\mu(k)}) - t^*)\bar{r}_k, \tag{19}$$

после чего положить $\mu(k) := \mu(k) + 1$; $y_k^{\mu(k)} := (t^*, R)$.

Пересчитать интенсивность потребления ресурса k: $\bar{r}_k := \bar{r}_k - r(\hat{h})$.

Если $i = i(\hat{h}) > 1$, то добавить в H новый элемент $h' = (j, k, i - 1, r_{jk}^{i-1}, t^* - p_{jk}^{i-1})$ (соответствующий $(i-1)$-му интервалу постоянства функции интенсивности потребления ресурса k работой j), перестроить кучу H, пересчитать τ^* и интенсивность потребления ресурса k: $\bar{r}_k := \bar{r}_k + r(h')$.

Если $i = i(\hat{h}) = 1$ (т. е. работа j просмотрена по ресурсу k), то

Начало

Положить $c_j := c_j - 1$.

Если $c_j = -K$ (работа j полностью просмотрена), то

Начало

Определить момент начала работы j в расписании S: $s_j := t^*$.

Изменить значение счетчика непросмотренных работ: $C := C - 1$.

Если $C = 0$, то Стоп (алгоритм $\mathscr{A}_R(T)$ завершил работу).

Иначе

Просмотреть все работы $j' \in \mathrm{Pred}(j)$ и для каждой положить $c_{j'} := c_{j'} - 1$. Если $c_{j'} = 1$ (это означает, что все работы $j'' \in \mathrm{Succ}(j)$ уже просмотрены) и при этом $d_{j'}^T \geqslant t^*$, то включить работу j' во фронт текущих работ, выполняя следующие действия:

— положить $c_{j'} := 0$;

— для каждого ресурса $k \in \mathscr{K}^\sigma$ определить пятерку $h^{j'k} = (j', k, i, r, \tau)$, где $i = N_{j'k}$, $r = r_{j'k}^i$, $\tau = t^* - p_{j'k}^i$;

— включить ее в дерево H и перестроить H до кучи;

— пересчитать $\bar{r}_k$: $\bar{r}_k := \bar{r}_k + r$;

— пересчитать τ^*.

Конец

Конец

Процедура В.

В момент $t^* = d^*$ во фронт F могут добавиться работы, чей директивный срок совпадает с d^*. Для выявления всех таких работ просмотреть список $J(d^*)$: если для $j' \in J(d^*)$ выполнено $c_{j'} = 1$, то включить работу j' в F, выполняя все предусмотренные действия (см. выше).

Директивный срок $d = d^*$ из списка $\mathscr{D}$ считаем «просмотренным», счетчик μ_d уменьшить на 1 и найти новое значение d^*, равное μ_d-му директивному сроку из $\mathscr{D}$; если $\mu_d = 0$, то $d^* := 0$.

Алгоритм $\mathscr{A}_R(T)$ описан.

Лемма 1. *Алгоритм $\mathscr{A}_R(T)$ строит T-позднее расписание S^T и находит все узлы функций $R_k^T(t)$ интенсивности потребления ресурсов $k \in \mathscr{K}^\sigma$ при расписании S^T. Временная сложность алгоритма не превышает $O(N(\log f + \log K) + u)$, где N — суммарное по всем работам и ресурсам число интервалов постоянства функций интенсивности потребления ресурсов, f — ширина частичного порядка графа G, u — число дуг в графе G, K — число ресурсов.*

Доказательство. Прежде всего необходимо убедиться, что алгоритм завершает работу за конечное время. (Иначе говоря, что в алгоритме через конечное время наступает ситуация, когда все работы просмотрены и счетчик C равен 0.) Для этого достаточно показать, что каждая работа $j \in J$ на каком-то шаге будет включена во фронт F (после чего она неизбежно будет просмотрена через конечное число шагов). Предположим противное, т. е. что множество J_x работ, не включаемых алгоритмом $\mathscr{A}_R(T)$ во фронт, не пусто. Тогда в алгоритме неизбежно наступает момент, когда все работы $j \in J \setminus J_x$ и все директивные сроки $d \in \mathscr{D}$ уже просмотрены. Поскольку в этот момент фронт пуст, то $\tau^* = 0$, а так как все директивные сроки просмотрены, то $d^* = 0$ и $t^* = 0$. Из неотрицательности директивных сроков следует, что для всех работ $j \in J_x$ выполнено $d_j^T \geqslant t^*$, и единственной причиной их невключения во фронт может быть неравенство $c_j > 1$. Пусть j^* — максимальная из работ J_x, т. е. в J_x не существует работы j, которой предшествует j^*. Отсюда следует, что $\mathrm{Succ}(j^*) \cap J_x = \varnothing$, т. е. все работы $j \in \mathrm{Succ}(j^*)$ просмотрены и $c_{j^*} = 1$. Противоречие доказывает, что в процессе работы алгоритма $\mathscr{A}_R(T)$ все работы $j \in J$ будут просмотрены (и им будет назначено время старта s_j).

Факт построения алгоритмом $\mathscr{A}_R(T)$ T-позднего расписания S^T вытекает из того, что любая работа $j \in J$ включается во фронт либо на шаге 0 (тогда $s_j = T - p_j = d_j^T - p_j$), либо на одном из последующих шагов в процедуре A $\left(\text{тогда } s_j = \min\left\{d_j^T, \min\limits_{i \in \mathrm{Succ}(j)} s_i\right\} - p_j\right)$, либо

в процедуре B $\left(\text{тогда } s_j = d_j - p_j = \min\left\{d_j^T, \min\limits_{i\in \mathrm{Succ}(j)} s_i\right\} - p_j\right)$. Во всех случаях значения s_j совпадают со значениями s_j^T, вычисляемыми по формуле (17) и определяющими T-позднее расписание.

Очевидно также, что величина R, вычисляемая по формуле (19), совпадает со значением функции $R_k^T(t^*)$ и что множество пар (t, R), вычисляемых алгоритмом $\mathscr{A}_R(T)$, содержит множество всех узлов функций $R_k^T(t)$, $k \in \mathscr{K}^\sigma$.

Для завершения доказательства леммы 1 остается найти оценку временной сложности алгоритма $\mathscr{A}_R(T)$. На шаге 0 потребуется $O(|H|) \leqslant O(|F| \cdot K)$ операций на создание кучи H, что не превосходит $O(N)$.

В процессе работы алгоритма каждый директивный срок $d \in D$ просматривается лишь один раз. Сложность такого просмотра не превосходит $O(n) \leqslant O(N)$.

Каждая дуга $u \in U$ также просматривается в алгоритме лишь однажды, что дает вклад $O(u)$.

Наконец, добавление или удаление одного элемента из кучи H требует $O(\log|H|)$ действий. Поскольку в каждый момент времени t одновременно может выполняться не более f работ, то на каждом шаге в куче содержится не более fK элементов, т. е. $|H| \leqslant fK$. Так как элементами кучи являются пятерки, соответствующие узлам функций $r_{jk}(t)$, и общее число этих узлов равно $O(N)$ и так как, очевидно, каждая пятерка включается в кучу и исключается из нее лишь однажды, то преобразование кучи H в процессе работы алгоритма требует не более $O(N\log(fK))$ операций. Сложив все приведенные выше оценки, получим требуемую в лемме 1 оценку времени работы алгоритма $\mathscr{A}_R(T)$. Лемма 1 доказана.

Замечание 2. Хотя в описанном выше алгоритме $\mathscr{A}_R(T)$ предполагается, что он нумерует узлы $y_k^i = (t, q)$, $i = 1, \ldots, N_k$, функции $R_k^T(t)$ в порядке **убывания** значения t, ниже для удобства изложения будем предполагать обратное. (Такая перенумерация узлов не требует значительных затрат времени.) При этом для каждой функции $R_k^T(t)$ узел y_k^0 будет отождествляться с парой $(0, 0)$, тогда как последний узел $y_k^{N_k}$ — с парой $(T, R_{k,\Sigma})$.

Отыскание узлов функции $Q_k(t)$

Нахождение узлов Y_k^i функций $Q_k(t)$ можно осуществить с помощью менее сложного алгоритма (по сравнению с описанным выше алгоритмом $\mathscr{A}_R(T)$), поскольку функции $Q_k(t)$ не зависят ни от ограничений предшествования на множестве работ, ни от директивных сроков, ни от какого-либо расписания.

Каждый узел Y_k^i будет отождествляться с парой $(t, Q) = (t(Y_k^i), Q(Y_k^i))$, где t — момент времени, а Q — значение функции $Q_k(t)$. Последовательность узлов $Y_k^0, Y_k^1, \ldots, Y_k^{N'_k}$ находится из рекуррентных соотношений: $Y_k^0 = (0, 0)$; $Y_k^i = Y_k^{i-1} + (\rho_k^i, q_k^i \rho_k^i)$, $i = 1, \ldots, N'_k$.

Вычисление величины $\Delta(T)$

Для вычисления величины $\Delta(T)$ применяется следующий алгоритм. Для каждого ресурса $k \in \mathscr{K}^\sigma$ вычисляется значение Δ_k величины минимального сдвига графика функции $R_k^T(t)$ вправо, при котором (сдвинутый) график целиком помещается под графиком функции $Q_k(t)$. Максимальное из чисел Δ_k, $k \in \mathscr{K}^\sigma$, выбирается в качестве значения величины $\Delta(T)$.

В силу сказанного (в начале разд. 4) о суммарных объемах выделяемых и потребляемых ресурсов для всякого $k \in \mathscr{K}^\sigma$ имеем $R_{k,\Sigma} = R(y_k^{N_k}) \leqslant Q(Y_k^{N'_k})$. Ясно, что в этом случае величина Δ_k равна максимальному горизонтальному расстоянию между графиками функций $Q_k(t)$ и $R_k^T(t)$. Так как эти функции кусочно-линейны, то значение Δ_k достигается в одном из узлов этих функций. Поэтому для вычисления Δ_k все узлы функций $Q_k(t)$ и $R_k^T(t)$ последовательно просматриваются по возрастанию номеров; при этом узлы сравниваются по объемам ресурсов $R(y_k^\mu)$ и $Q(Y_k^\nu)$. Приведем более детальное описание алгоритма $\mathscr{A}_{\Delta_k}$ вычисления величины Δ_k.

Алгоритм $\mathscr{A}_{\Delta_k}$.

Положить $\Delta_k := 0$. Найти минимальный номер $i = \mu$ такого узла y_k^i, что $R(y_k^i) > 0$, а также аналогичный номер ν для узла функции $Q_k(t)$. (Очевидно, что $\mu > 0$, $\nu > 0$.) Предполагаем, что интервалы $[R(y_k^{\mu-1}), R(y_k^\mu)]$ и $[Q(Y_k^{\nu-1}), Q(Y_k^\nu)]$ пересекаются. (Такую ситуацию мы имеем в начале работы алгоритма.) Далее выполнять общий шаг.

Общий шаг. Сравнить значения $R(y_k^\mu)$ и $Q(Y_k^\nu)$. Дальнейшая работа алгоритма разделяется на два симметричных случая в зависимости от того, на каком из двух узлов y_k^μ, Y_k^ν достигается $R' \doteq \min\{R(y_k^\mu), Q(Y_k^\nu)\}$.

Случай 1 ($R' = R(y_k^\mu)$).

Вычислить горизонтальное расстояние Δ между графиками функций $Q_k(t)$ и $R_k^T(t)$ в узле y_k^μ:

$$\Delta := t(Y_k^\nu) - t(y_k^\mu) + \frac{R(y_k^\mu) - Q(Y_k^\nu)}{\rho_k^\nu}.$$

Если $\Delta > \Delta_k$, то $\Delta_k := \lceil \Delta \rceil$.

Пока $R(y_k^\mu) = q'$, положить $\mu := \mu + 1$. Если $t(y_k^\mu) \geqslant T$, то Стоп. (В итоге в переменной μ будет храниться номер ближайшего узла y_k^μ такого, что $R(y_k^\mu) > R(y_k^{\mu-1}) = R'$.) Повторить общий шаг.

Случай 2 ($R' = Q(Y_k^\nu)$).

Расстояние Δ определяется из аналогичной формулы:

$$\Delta := t(Y_k^\nu) - t(y_k^\mu) + \frac{R(y_k^\mu) - Q(Y_k^\nu)}{\bar{r}_k},$$

где $\bar{r}_k = (R(y_k^\mu) - R(y_k^{\mu-1}))/(t(y_k^\mu) - t(y_k^{\mu-1}))$ — интенсивность потребления ресурса k в интервале $[t(y_k^{\mu-1}), t(y_k^\mu))$.

Если $\Delta > \Delta_k$, то $\Delta_k := \lceil \Delta \rceil$.

Пока $Q(Y_k^\nu) = R'$, присвоить $\nu := \nu + 1$. Если $\nu > N_k'$, то Стоп. Иначе повторить общий шаг.

Алгоритм $\mathscr{A}_{\Delta_k}$ описан.

Временная сложность алгоритма $\mathscr{A}_{\Delta_k}$ оценивается общим числом узлов функций $Q_k(t)$ и $R_k^T(t)$, что в сумме по всем $k \in \mathscr{K}^\sigma$ составляет сложность $O(N + N')$ вычисления величины $\Delta(T)$, используемой для проверки допустимости расписания S^T.

Замечание 3. Параметры Δ' и Δ'', фигурирующие на втором этапе алгоритма $\mathscr{A}$, также вычисляются с помощью алгоритмов $\mathscr{A}_{\Delta_k}$, $k \in \mathscr{K}^\sigma$, при $T = T''$. При этом для вычисления параметра Δ' применяется усеченный алгоритм $\mathscr{A}_{\Delta_k}$, где при повторении общего шага (в случае 1) команда Стоп выполняется раньше. А именно не при $t(y_k^\mu) \leqslant T''$, а при $t(y_k^\mu) \leqslant D$. Таким образом, параметры Δ' и Δ'' вычисляются с той же сложностью $O(N + N')$.

6. Завершение доказательства теоремы 1

Из теоремы 2 следует, что алгоритм $\mathscr{A}$ находит оптимальное решение задачи PS^σ. Для завершения доказательства теоремы 1 остается показать, что временная сложность алгоритма $\mathscr{A}$ оценивается величиной $O(|J_{\text{dir}}| \log N_d + (N(\log f + \log K) + u + N')\widehat{D})$, где f — ширина частичного порядка, u — число дуг в графе редукции частичного порядка, K — число видов ресурсов, J_{dir} — множество работ с директивными сроками, N_d — число различных директивных сроков, N' — суммарное по всем ресурсам число интервалов постоянства функций интенсивности выделения ресурсов, N — суммарное по всем работам и всем ресурсам число интервалов постоянства функций интенсивности потребления ресурсов, $\widehat{D}$ — длина двоичной записи наибольшего директивного срока.

Как следует из утверждения 1, построение раннего расписания S' на этапе 1 и проверка его допустимости по директивным срокам могут быть

выполнены за время $O(u)$. Вычисление параметра T_Σ осуществляется за время $O(N+N')$.

На этапе 2 нам прежде всего требуется организовать (в виде списка $\mathscr{D}$, упорядоченного по возрастанию) множество различных директивных сроков, с припиской каждому сроку $d \in \mathscr{D}$ множества $J(d)$ работ, чей директивный срок в точности равен d. Для создания такого списка сначала организуем множество различных директивных сроков в виде *сбалансированного бинарного дерева* $\widehat{\mathscr{D}}$ (см. [12, с. 45–57]). Такое дерево характеризуется тем, что для каждой его вершины $v \in \widehat{\mathscr{D}}$ с ключом $k(v)$ все вершины в левом поддереве имеют значения ключа, меньшие $k(v)$, а все вершины в правом поддереве — значения ключа, большие $k(v)$. При этом глубина дерева не превосходит $O(\log|\widehat{\mathscr{D}}|) = O(\log N_d)$.

Сбалансированное бинарное дерево $\widehat{\mathscr{D}}$ обладает тем свойством, что для любого значения x за $O(\log|\widehat{\mathscr{D}}|)$ шагов либо находится вершина v со значением ключа $k(v) = x$, либо (если такой вершины не существует в дереве $\widehat{\mathscr{D}}$) в дерево $\widehat{\mathscr{D}}$ добавляется новая вершина v со значением ключа $k(v) = x$; при этом дерево сохраняет свойство сбалансированности. Отсюда ясно, что нахождение множества всех **различных** директивных сроков работ из J_{dir} и организация его в виде сбалансированного бинарного дерева требуют $O(|J_{\text{dir}}|\log N_d)$ операций. В эту оценку входит трудоемкость формирования для каждого $d \in \widehat{\mathscr{D}}$ списка работ $J(d) = \{j \in J_{\text{dir}} \mid d_j = d\}$ (формирование таких списков происходит одновременно с построением дерева $\widehat{\mathscr{D}}$). Поскольку вершина с минимальным значением ключа является, очевидно, крайним левым «листом» дерева $\widehat{\mathscr{D}}$, она находится (при поиске, начинающемся в корне дерева $\widehat{\mathscr{D}}$) за время $O(\log N_d)$. Таким образом, перестройка дерева $\widehat{\mathscr{D}}$ во вполне упорядоченный список $\mathscr{D}$ может быть выполнена за $O(N_d \log N_d) \leqslant O(|J_{\text{dir}}|\log N_d)$ операций.

С учетом приведенных выше оценок временной сложности алгоритма $\mathscr{A}_R(T)$ построения T-позднего расписания S^T и нахождения узлов функций $R_k^T(t)$, $k \in \mathscr{K}^\sigma$ (см. лемму 1) и алгоритма проверки допустимости расписания S^T по ресурсным ограничениям (что эквивалентно вычислению величины $\Delta(T)$ и проверке равенства ее нулю), временная сложность каждой итерации процедуры P_0 (при фиксированном значении T) оценивается величиной $O(N(\log f + \log K) + u + N')$. Для получения итоговой оценки сложности процедуры P_0 требуется умножить эту величину на оценку числа итераций: $O(\widehat{D})$. Согласно замечанию 3 сложность вычисления параметров Δ' и Δ'' на этапе 2 составляет величину $O(N+N')$ и не влияет на итоговую оценку сложности алгоритма $\mathscr{A}$. Наконец, построение оптимального расписания на этапе 3 (с определением

множества J'' «директивно-независимых» работ и вычислением моментов s_j начала каждой работы $j \in J$ по формуле (18) выполняется за время $O(u)$. (Фактически множество J'' находится одновременно с построением расписания $S^{T''}$.) Нетрудно видеть, что из приведенных оценок сложности используемых алгоритмов вытекает требуемая оценка сложности алгоритма $\mathscr{A}$. Теорема 1 доказана.

Замечание 4. В качестве наибольшего директивного срока можно взять максимальный директивный срок D', для которого не существует D'-позднего расписания. Вычисление величины D', очевидно, не изменит общую оценку алгоритма $\mathscr{A}$.

7. Полиномиально разрешимый случай задачи MPS^{σ}

Напомним, что в задаче MPS^{σ} каждая работа $j \in J$ может быть выполнена в одной из мод $m \in \{1, \ldots, \mathscr{M}_j\}$. Каждой моде m работы j соответствуют продолжительность p_j^m выполнения работы j и интенсивность $r_{jk}^m(t) \geqslant 0$ потребления складируемого ресурса $k \in \mathscr{K}^{\sigma}$, а также общая потребность в невозобновимых ресурсах $r_{jk}^m \geqslant 0$, $k \in \mathscr{K}^{\nu}$. Прерывания работы или смена моды в процессе выполнения работы не допускаются.

Известно, что уже при двух невозобновимых ресурсах и $\mathscr{M}_j \geqslant 2$ существование допустимого решения задачи МЗКП является NP-полной задачей даже в отсутствие ограничений на ресурсы других типов и ограничений предшествования [11]. Заметим, что при допущении произвольной динамики выделения складируемых ресурсов невозобновимый ресурс можно представить как складируемый, весь объем которого выделяется к моменту начала выполнения проекта. (Функции интенсивностей потребления ресурсов доопределяются произвольно.) Отсюда, в частности, следует NP-трудность задачи MPS^{σ}. Тем не менее удается выделить частный случай задачи MPS^{σ}, для которого описанный выше алгоритм $\mathscr{A}$ гарантирует нахождение точного решения.

Теорема 3. *Пусть для каждой работы $j \in J$ первой моде соответствует наименьшая длительность работы (p_j^1) и выполнены неравенства*

$$\sum_{t'=1}^{t} r_{jk}^1(t') \leqslant \sum_{t'=1}^{p_j^m - p_j^1 + t} r_{jk}^m(t')$$

для всяких $k \in \mathscr{K}^{\sigma}$, $m \in \{2, \ldots, \mathscr{M}_j\}$ и $t = 1, \ldots, p_j^1$. Тогда оптимальное решение задачи MPS^{σ} может быть найдено с помощью предложенного выше алгоритма решения задачи PS^{σ}.

Для доказательства теоремы достаточно заметить, что если имеется допустимое расписание, в котором какая-то работа выполняется не в своей кратчайшей по длительности моде, то замена моды на первую с сохранением момента окончания работы оставит расписание допустимым с точки зрения ограничений предшествования и директивных сроков и при этом ни в один из моментов t функция $R_k^T(t)$ суммарного потребления ресурса k не возрастет. Таким образом, можно ограничиться рассмотрением только тех расписаний, в которых все работы выполняются в первой (кратчайшей) моде, задача сводится к одномодальной и справедливость теоремы 3 непосредственно следует из теоремы 1.

Следствие. *Пусть интенсивность $r_{jk}^m(t)$ потребления ресурса k работой j в моде m постоянна для любых k, j, m. Тогда условия теоремы 3 выполнены тогда и только тогда, когда для каждой работы j и каждого ресурса k первой моде соответствует наименьший объем потребляемых ресурсов $(r_{jk}^1 p_j^1)$.*

Следует отметить также, что ограничениям теоремы 3 удовлетворяет, в частности, такой практически важный класс задач MPS^σ, у которого для всякой работы j и ресурса k задан объем потребления ресурсов b_{jk} и длительности p_j^m выполнения работы по каждой моде, причем интенсивность r_{jk}^m потребления ресурсов для соответствующей моды обратно пропорциональна ее длительности.

ЛИТЕРАТУРА

1. **Гимади Э. Х.** О некоторых математических моделях и методах планирования крупномасштабных проектов // Модели и методы оптимизации. Новосибирск: Наука, 1988. С. 89–115. (Тр. / АН СССР. Сиб. отд-ние. Ин-т математики; Т. 10).

2. **Гимади Э. Х., Глебов Н. И.** Экстремальные задачи принятия решений. Новосибирск: Новосиб. ун-т, 1982.

3. **Гимади Э. Х., Пузынина Н. М.** Задача календарного планирования крупномасштабного проекта в условиях ограниченных ресурсов: опыт построения математического обеспечения // Управляемые системы: Сб. науч. тр. Новосибирск: Ин-т математики СО АН СССР, 1983. Вып. 23. С. 24–32.

4. **Гимади Э. Х., Пузынина Н. М., Севастьянов С. В.** О некоторых экстремальных задачах реализации крупных проектов типа БАМ // Экономика и мат. методы. 1979. Вып. 5. С. 1017–1020.

5. **Зуховицкий С. И., Радчик И. А.** Математические методы сетевого планирования. М.: Наука, 1965.

6. **Козлов М. К., Шафранский В. В.** Календарное планирование выполнения комплексов работ при заданной динамике поступления складируемых ресурсов // Изв. АН СССР. Техническая кибернетика. 1977. № 4. С. 75–81.
7. **Филлипс Д., Гарсиа-Диас А.** Методы анализа сетей. М.: Мир, 1984.
8. **Blaźewicz J., Lenstra J. K., Rinnoy Kan A. H. G.** Scheduling subject to resource constraints: Classification and complexity // Discrete Applied Math. 1983. V. 5, N 1. P. 11–24.
9. **Böttcher J., Drexl A., Salewski F.** Project scheduling under partially renewable resource constraints // Management Sci. 1999. V. 45, N 4. P. 543–559.
10. **Brucker P., Drexl A., Möhring R., Neumann K., Pesch E.** Resource-constrained project scheduling: Notation, classification, models, and methods // European J. Oper. Res. 1999. V. 112, N 1. P. 3–41.
11. **Kolish R.** Project scheduling under resource constraints // Efficient heuristics for several problem classes. Heidelberg: Physica, 1995.
12. **Tarjan R. E.** Data Structures and Network Algorithms. PA: SIAM, Philadelphia, 1983.

Адрес авторов:

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

Статья поступила
27 мая 1999 г.,
переработанный вариант —
25 мая 2000 г.