Оптимизация, системный анализ и исследование операций

**© 2023 г. Е.Г. МУСАТОВА, канд. физ.-мат. наук (nekolyap@mail.ru),
А.А. ЛАЗАРЕВ, д-р физ.-мат. наук (jobmath@mail.ru)
(Институт проблем управления им. В.А. Трапезникова РАН, Москва)**

ЗАДАЧА МИНИМИЗАЦИИ СУММАРНОЙ ВЗВЕШЕННОЙ ДЛИТЕЛЬНОСТИ КУРСОВ ДЛЯ ОДНОГО ПРИБОРА С ОГРАНИЧЕНИЯМИ ПРЕДШЕСТВОВАНИЯ

Рассматривается одноприборная задача теории расписаний с заданным частичным порядком выполнения работ. Имеются подмножества работ, именуемые курсами. Необходимо построить расписание работ, при котором суммарное взвешенное время обработки всех курсов минимально. Рассматривается случай, когда начальная и конечная работы каждого курса определены однозначно. Доказана NP-трудность рассматриваемой задачи. Предложен алгоритм решения задачи, трудоемкость которого полиномиально зависит от общего числа работ, но экспоненциально — от количества курсов, что позволяет эффективно его использовать при фиксированном небольшом количестве курсов и произвольном числе работ.

Ключевые слова: теория расписаний, задача одного прибора, NP-трудные задачи, минимизация простоя ресурсов.

DOI: 10.31857/S000523102309009X, **EDN:** JUNOUZ

1. Введение

Имеются множество работ, которые необходимо обслужить на одном приборе, и граф отношения предшествования между работами, задающий частичный порядок выполнения работ. Некоторые из работ объединены во множества, которые будем называть курсами. Необходимо построить расписание, при котором суммарное взвешенное время обработки всех курсов минимально. Под временем обработки курса понимается промежуток между началом обработки первой работы из курса и моментом окончания обработки последней работы из курса. В статье рассмотрен случай, когда начальная и конечная работы каждого курса определены однозначно.

Задачи построения расписаний множества работ на одном приборе всесторонне исследованы в теории расписаний [1, 2]. При этом одноприборная задача минимизации взвешенной суммарной продолжительности курсов ранее не рассматривалась.

Необходимость минимизировать суммарную продолжительность курсов возникает в разных областях производства, образования и сферы услуг. В [3] рассматривается задача управления проектом с ограничениями на ресурсы с данной целевой функцией применительно к построению расписания подготовки космонавтов к работе на Международной космической станции. Необходимо минимизировать растянутость каждого курса (или бортового комплекса в терминологии Центра подготовки космонавтов им. Ю.А. Гагарина), поскольку, если с начала изучения курса до экзамена проходит слишком много времени, навыки космонавтов считаются утраченными и приходится добавлять дополнительные часы в подготовительный процесс, что приводит к большим временным и финансовым потерям. В этой публикации предлагается эвристический алгоритм построения расписания для поставленной задачи.

Также можно интерпретировать задачу как задачу минимизации суммарного простоя ресурсов. Предположим, что работы каждого курса требуют свой специфический ресурс (например, обработка на дополнительном оборудовании). Данный ресурс берется во вре́менную аренду, которая начинает выплачиваться одновременно с выполнением первой работы из курса и заканчивается завершением последней работы из данного курса. Тогда длительность курса может ассоциироваться с суммарной выплатой за ресурс, а целевая функция характеризует общие выплаты за все арендуемые ресурсы. Помимо выплат за дополнительные ресурсы, данная целевая функция может рассматриваться как плата за хранение и аренду помещений.

Впервые вопрос длительности выполнения курсов был рассмотрен в [4]. Здесь наряду с обычным понятием «работа» вводится понятие «работа-гамак» (hammock activity). Продолжительность работы-гамака определяется началом и окончанием некоторых фиксированных работ. В данной статье рассматривается управление проектом без ограничения на ресурсы и приводятся методы подсчета продолжительности работы-гамака. Отметим, что в случае, когда первая и последняя работы курса определены однозначно, понятия работы-гамака и курса совпадают.

Свое развитие концепция работы-гамака получила в [5], где рассмотрена задача минимизации суммарной стоимости нескольких работ-гамаков в проекте как при наличии ресурсных ограничений (Resource-Constrained Hammock Cost Problem, RCHCP), так и без них. Под стоимостью работы-гамака понимается ее взвешенная продолжительность. При отсутствии ресурсных ограничений задача сводится к задаче линейного программирования. В случае наличия ресурсных ограничений предлагается формулировка задачи в виде задачи смешанного целочисленного линейного программирования. В диссертации [6] продолжено исследование задачи RCHCP и предложена метаэвристика для решения данной задачи, а также приведен обширный обзор публикаций по задачам типа RCHCP.

В некоторых исследованиях используются отличные от работы-гамака термины для описания подобной целевой функции. Так, в теории расписаний

с повторяющимися работами (см., например, [7]) подобные задачи известны как задачи управления проектом с ограничениями на непрерывность работы (work continuity constraints). Например, в [8] рассматривается следующая задача управления проектом с повторяющимися работами. Имеется некоторый базовый граф отношения предшествования работ, который дублируется k раз. Некоторые из повторяющихся работ требуют дополнительного ресурса (оборудования, бригады рабочих и т.д.), и необходимо выполнить проект к заданному директивному сроку с минимизацией длительности выполнения этих повторяющихся работ. В качестве практических приложений приводятся примеры строительства многоэтажных зданий, где на каждом этаже выполняются идентичные работы, или строительство мостов, дорог и т.д. В [9] используются термины минимизации простоя бригад (crew idle time) и минимизации простоя ресурсов (resource idle time), а также описывается практическое использование разработанных для такой задачи алгоритмов при строительстве туннеля Вестерсхелдетюннел в Нидерландах. В качестве ресурсов, суммарную длительность использования которых нужно было минимизировать, были выбраны бригады рабочих и морозильные машины.

Таким образом, если говорить о минимизации суммарной длительности курсов, основное внимание в литературе уделено задачам управления проектом с ресурсными ограничениями или без них. При этом в первом случае приходится иметь дело с NP-трудной задачей [5] и упор в таких исследованиях делается на разработку эвристических алгоритмов, а во втором случае строятся полиномиальные алгоритмы.

В данной статье рассматривается одноприборная задача, которая может быть проинтерпретирована как задача управления проектом с одним ресурсом, доступным в количестве одной единицы в каждый момент времени при условии, что на выполнение каждой работы также требуется одна единица ресурса. Показано, что данная задача является NP-трудной. Предлагается алгоритм, позволяющий найти точное решение в случае большого количества работ, но небольшого фиксированного количества курсов. В разделе 2 приводится формулировка задачи. Раздел 3 содержит доказательство NP-трудности рассматриваемой задачи, а также некоторые ее свойства. Раздел 4 посвящен решению вспомогательной задачи, а в разделе 5 описан алгоритм решения исходной задачи на базе решения вспомогательной задачи и приведены результаты численного эксперимента.

2. Постановка задачи

Имеется множество работ $I = \{1, \ldots, n\}$, которые должны быть обслужены на одном приборе. Для каждой работы $i \in I$ известна ее продолжительность $p_i > 0$. Все работы доступны в нулевой момент времени. Прерывания при обслуживании работ запрещены.

Задан ориентированный ациклический граф отношения предшествования работ $G(I, E)$, где I — множество вершин, а E — множество дуг. Будем гово-

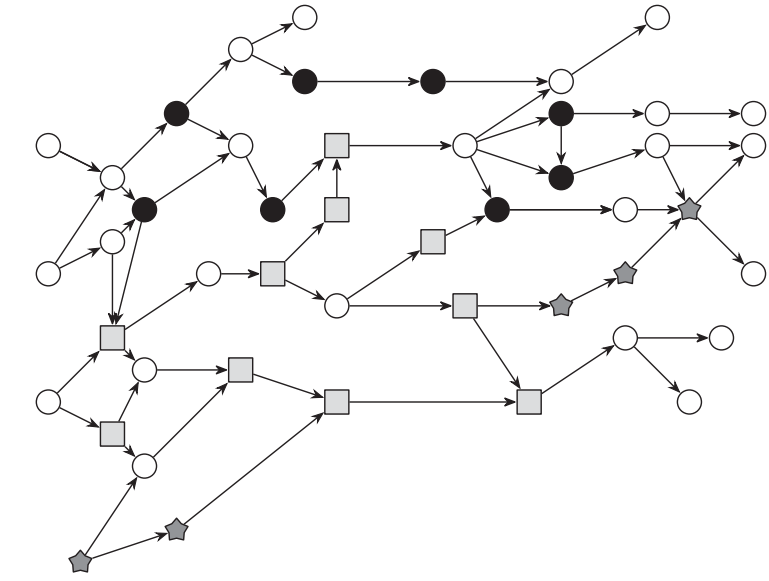

Рис. 1. Граф отношения предшествования работ и курсы.

рить, что для пары работ $i, j \in I$ работа i предшествует работе j, обозначая $i \to j$, если существует направленный путь из вершины i в вершину j в графе $G(I, E)$. Множество всех работ, предшествующих работе i, будем обозначать через $A(i)$, а множество работ, которым предшествует работа i, обозначим через $D(i)$. Каждая работа i должна быть выполнена после всех работ из множества $A(i)$ и до всех работ из $D(i)$.

Кроме того, заданы множества $I_k \subset I$, $|I_k| > 1$, $k \in \{1, \ldots, K\}$, именуемые *курсами*. На рис. 1 приведен пример графа отношения предшествования для задачи с тремя курсами. Каждый курс I_k, $k \in \{1, \ldots, K\}$, имеет свой вес $w_k > 0$. В зависимости от интерпретации вес — это цена арендованного ресурса в единицу времени или важность (значимость) курса. Для каждой работы $i \in I$ расписание π определяет ее порядковый номер обработки на приборе, который будем обозначать через $\pi(i)$, момент начала обработки $S_i(\pi)$ и момент завершения обработки $C_i(\pi) = S_i(\pi) + p_i$. Допустимым будем называть расписание, не противоречащее отношениям предшествования работ, в котором прибор в каждый момент времени обслуживает не больше одной работы. Задача минимизации суммарного взвешенного времени обработки всех курсов подразумевает минимизацию следующей целевой функции:

$$H(\pi) = \sum_{k=1}^{K} w_k \left(\max_{i \in I_k} C_i(\pi) - \min_{i \in I_k} S_i(\pi) \right). \tag{1}$$

В статье рассматривается случай, когда первая и последняя работы каждого курса определены однозначно, т.е. справедливо следующее

Предположение 1. Для каждого курса I_k, $k \in \{1, \ldots, K\}$, существуют работы i_k^a и i_k^d, такие что для любых $j \in I_k \setminus \{i_k^a\}$ выполнено $i_k^a \in A(j)$ и для любых $j \in I_k \setminus \{i_k^d\}$ выполнено $i_k^d \in D(j)$.

Такое условие часто выполняется на практике. Например, в учебном процессе первое занятие, как правило, вводное, а последнее подразумевает общую проверку знаний, в то время как последовательность других занятий в курсе может меняться. В этом случае целевая функция может быть записана как

$$H(\pi) = \sum_{k=1}^{K} w_k \Big(C_{i_k^d}(\pi) - S_{i_k^a}(\pi) \Big) = \sum_{k=1}^{K} w_k \Big(C_{i_k^d}(\pi) - C_{i_k^a}(\pi) + p_{i_k^a} \Big). \tag{2}$$

Будем называть i_k^a и i_k^d *крайними* работами (вершинами) курса k, $k \in \{1, \ldots, K\}$. Множество всех крайних работ курсов обозначим через I_{ad}, а их количество — через e. Поскольку некоторые работы из I_{ad} могут быть одновременно крайними в нескольких курсах, $e \leqslant 2K$.

Минимизация функции (2) в точности совпадает с минимизацией взвешенной суммарной продолжительности работ-гамаков, описанных во введении. С точки зрения принятой в теории расписаний системы обозначений [10] данная задача может быть классифицирована как $1|prec|H$, где 1 означает один прибор, $prec$ — наличие ограничений предшествования, а H — целевую функцию (2).

3. Свойства задачи

Замечание 1. Поскольку все работы в задаче $1|prec|H$ доступны одновременно и простои в работе не улучшают значение целевой функции, можно рассматривать только расписания без перерывов между работами, с началом выполнения первой работы в нулевой момент времени. Действительно, если существует оптимальное расписание задачи π_1, в котором имеются простои прибора или первая работа начинается не в нулевой момент времени, то расписание π_2, в котором все работы выполняются в том же порядке, что и в π_1, но начиная с нулевого момента времени и без перерывов между работами, также является оптимальным.

Покажем, что даже при одинаковой продолжительности всех работ рассматриваемая задача NP-трудна.

Теорема 1. Задача $1|prec, p_i = 1|H$ является NP-трудной в сильном смысле.

Доказательство. Рассмотрим классическую одноприборную задачу минимизации взвешенной суммы моментов окончания всех работ $1|prec, p_i = 1| \sum w_i C_i$. Задача формулируется следующим образом. Даны один прибор и множество $I' = \{1, \ldots, n'\}$ работ, каждая i-я работа имеет вес w'_i и продолжительность обслуживания $p'_i = 1$, $i \in \{1, \ldots, n'\}$. Пусть также задан ориен-

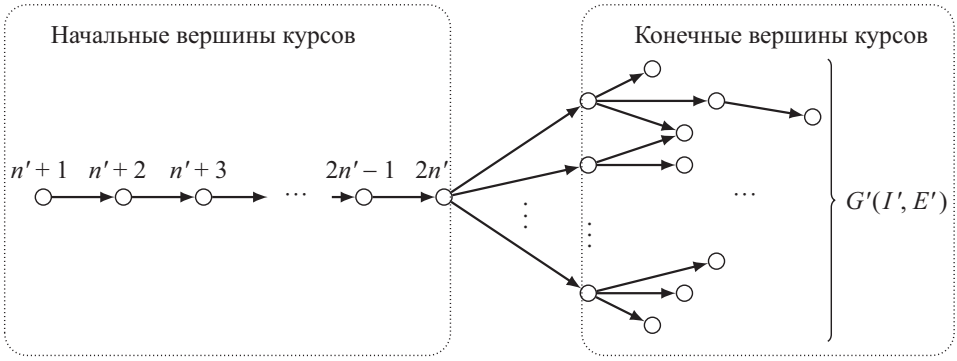

Рис. 2. Структура графа $G(I, E)$ из доказательства теоремы 1.

тированный граф отношения предшествования работ $G'(I', E')$. Необходимо найти расписание π', минимизирующее целевую функцию $\sum_{i=1}^{n'} w'_c C'_i(\pi')$, где $C'_i(\pi')$ — момент окончания обслуживания i-й работы при расписании π'.

Данная задача является NP-трудной в сильном смысле [11]. Сведем ее к следующей задаче $1|prec, p_i = 1|H$. Дано множество работ $I = \{1, \ldots, n\}$, $n = 2n'$. Каждая работа i имеет продолжительность $p_i = 1$. Граф $G(I, E)$ имеет следующую структуру. Имеется $|E'|$ дуг, заданных следующим правилом: если в графе $G'(I', E')$ есть дуга (j, k), то в графе $G(I, E)$ также есть дуга (j, k). Кроме того, имеется $n'-1$ дуг вида $(i, i+1)$ для $i \in \{n'+1, \ldots, 2n'-1\}$ и $|L|$ дуг вида $(2n', l)$, где L — множество корневых вершин (источников) в графе G', $l \in L$. Структура графа $G(I, E)$ отражена на рис. 2.

Зададим курсы следующим образом: работы $n'+1$ и 1 относятся к первому курсу, имеющему вес w'_1, работы $n'+2$ и 2 относятся ко второму курсу, имеющему вес $w'_2, \ldots$, работы $2n'$ и n' относятся к n'-му курсу, имеющему вес $w'_{n'}$, т.е. каждый курс состоит из двух работ, первая из которых лежит во множестве $\{n'+1, \ldots, 2n'\}$, а конечная — во множестве I'. Пусть π — произвольное оптимальное расписание данной задачи. Значение целевой функции равно

$$\begin{aligned} H(\pi) &= \sum_{i=1}^{n'} w'_i \left(C_i(\pi) - C_{i+n'}(\pi) + 1\right) = \\ &= \sum_{i=1}^{n'} w'_i C_i(\pi) - \sum_{i=1}^{n'} w'_i C_{i+n'}(\pi) + \sum_{i=1}^{n'} w'_i. \end{aligned} \tag{3}$$

В силу структуры графа отношения предшествования первой будет выполнена работа $n'+1$. Тогда с учетом замечания 1 имеем $C_{n'+1}(\pi) = 1$. Поскольку в графе G имеются дуги $(i, i+1)$ для $i \in \{n'+1, \ldots, 2n'-1\}$, а все работы из $\{1, \ldots, n'\}$ выполняются после работы $2n'$, порядок работ $n'+2, \ldots, 2n'$ известен, более того, выполняется

$$C_i(\pi) = i - n', \quad i \in \{n'+2, \ldots, 2n'\}, \tag{4}$$

т.е. каждая работа выполняется друг за другом без перерывов в работе прибора. Таким образом, в расписании π выполнение работ $n'+1,\dots,2n'$ предопределено и закончится в момент времени n'. Но тогда с учетом (3) расписание π является оптимальным тогда и только тогда, когда выполняется (4) и при расписании π достигается минимум функции $\sum_{i=1}^{n'} w'_i C_i(\pi)$, т.е. когда в расписании π работы $1,\dots,n'$ обслуживаются начиная с момента n' таким образом, чтобы минимизировать их взвешенную сумму моментов окончания. В результате оптимальное расписание π' в задаче $1|prec, p_i = 1|\sum w_i C_i$ может быть получено из оптимального расписания π описанной задачи $1|prec, p_i = 1|H$. Моменты окончания обслуживания работ в задаче $1|prec, p_i = 1|\sum w_i C_i$ находятся следующим образом:

$$C'_i(\pi') = C_i(\pi) - n', \quad i \in \{1,\dots,n'\}.$$

Теорема доказана.

Замечание 2. Аналогичным образом можно доказать NP-трудность в сильном смысле одноприборной задачи минимизации суммарной (невзвешенной) продолжительности курсов с различными длительностями работ, используя сведение к ней NP-трудной задачи $1|prec|\sum C_i$.

Как было отмечено ранее, в силу замечания 1 далее будем рассматривать только расписания без перерывов между работами, с началом выполнения первой работы в нулевой момент времени. В этом случае для каждой работы j, $j \in I$, ее порядковый номер $\pi(j)$ в расписании π однозначно задает начало и завершение работы. Заметим также, что только крайние работы курса входят в определение целевой функции (2), более того, определяющей является только их разность, а не абсолютные значения. Поскольку при отсутствии перерывов в работе прибора длительность курса определяется работами, начатыми после первой работы курса и законченными до завершения последней работы курса, перепишем целевую функцию (2) в другом виде, без использования моментов окончания работ:

$$H(\pi) = \sum_{k=1}^{K} w_k \left(\sum_{j:\pi(i_k^a)\leqslant\pi(j)\leqslant\pi(i_k^d)} p_j \right). \tag{5}$$

Тогда можно записать

$$H(\pi) = \sum_{j=1}^{n} W_j(\pi) p_j, \tag{6}$$

где

$$W_j(\pi) = \sum_{\substack{k\in K:\\ \pi(j)\geqslant\pi(i_k^a)}} w_k - \sum_{\substack{k\in K:\\ \pi(j)>\pi(i_k^d)}} w_k. \tag{7}$$

Таким образом, вклад $W_j(\pi)$ каждой работы $j \in I$ в целевую функцию зависит от взаимного порядка крайних работ курсов и ее места среди крайних работ. В связи с этим возникает следующая идея решения задачи: для каждой допустимой перестановки крайних работ курсов необходимо найти оптимальный порядок выполнения работ относительно крайних работ. В следующем разделе будет представлен полиномиальный алгоритм построения оптимального расписания при заданном порядке крайних работ курсов.

4. Решение вспомогательной задачи

Как было показано в предыдущем разделе, значение целевой функции исходной задачи зависит от взаимного порядка крайних работ курсов. Обозначим через $\Lambda = (\lambda_1, \ldots, \lambda_e)$ произвольную перестановку крайних работ, не противоречащую отношениям предшествования, задаваемым графом $G(I, E)$, и введем ориентированный ациклический граф $G'(I, E')$ такой, что $E \subset E'$ и для множества крайних работ выполняется условие

$$\lambda_1 \to \lambda_2 \to \ldots \lambda_{e-1} \to \lambda_e. \tag{8}$$

Граф $G'(I, E')$ получен из $G(I, E)$ последовательным соединением крайних вершин $\lambda_1, \ldots, \lambda_e$ дугами в соответствии с порядком, задаваемым Λ. Если такой порядок не противоречит ограничениям предшествования задачи, полученный граф $G'(I, E')$ будет ациклическим. В силу ацикличности исходного графа $G(I, E)$ всегда существует хотя бы одна последовательность крайних работ Λ, не противоречащая ограничениям предшествования. Тогда в любом расписании π для графа $G'(I, E')$ будет выполнено

$$\pi(\lambda_1) < \pi(\lambda_2) < \ldots < \pi(\lambda_{e-1}) < \pi(\lambda_e).$$

Ставится задача минимизации функции (6)–(7) относительно нового графа $G'(I, E')$. Будем обозначать данную вспомогательную задачу через P_Λ.

Для всех работ, не являющихся крайними, необходимо определить место в последовательности $\lambda_1, \lambda_2, \ldots, \lambda_e$. Для каждой работы имеется не более $e+1$ вариантов (работа выполнятеся до λ_1, между λ_1 и λ_2 и т.д.). Будем говорить, что работа j помещена в *ячейку* q, $q \in \{1, \ldots, e-1\}$, если она обслуживается после крайней работы λ_q и до крайней работы λ_{q+1}. Будем считать, что $q = 0$, если работа j обслуживается до λ_1, и $q = e$, если j обслуживается после λ_e.

Определим для каждой крайней работы $\lambda_i \in I_{ad}$ множества $A(\lambda_i)$ и $D(\lambda_i)$ в графе $G'(I, E')$. Отметим несколько очевидных утверждений, которые будут использоваться далее и доказательство которых вытекает непосредственно из (8).

Лемма 1. a) Если $j \in A(\lambda_i)$, *то* $j \in A(\lambda_k)$ *для всех* $k \geqslant i$.

b) Если $j \in D(\lambda_i)$, *то* $j \in D(\lambda_k)$ *для всех* $k \leqslant i$.

c) Если $j \notin D(\lambda_i)$, *то* $j \notin D(\lambda_k)$ *для всех* $k \geqslant i$.

d) Если $j \notin A(\lambda_i)$, *то* $j \notin A(\lambda_k)$ *для всех* $k \leqslant i$.

Для определения границ возможного расположения некрайних работ по ячейкам в ряду крайних работ введем следующие обозначения:

$$q_1(j) = \begin{cases} 0, & \text{если } j \notin D(\lambda_1), \\ \max\{g \in \{1, \dots, e\} \mid j \in D(\lambda_g)\} & \text{иначе;} \end{cases}$$

$$q_2(j) = \begin{cases} e, & \text{если } j \notin A(\lambda_e), \\ \min\{g \in \{1, \dots, e\} \mid j \in A(\lambda_g)\} - 1 & \text{иначе.} \end{cases}$$

Лемма 2. Для любой работы $j \in I \setminus I_{ad}$ выполняется $q_1(j) \leqslant q_2(j)$.

Доказательство. Если $j \notin D(\lambda_1)$ или $j \notin A(\lambda_e)$, то утверждение очевидно. Для доказательства в остальных случаях предположим противное. Пусть $q_1(j) > q_2(j)$. По определению $q_1(j)$ имеем $j \in D(\lambda_{q_1(j)})$. С другой стороны, по определению $q_2(j)$ имеем $j \in A(\lambda_{q_2(j)+1})$. Но тогда по лемме 1 получаем $j \in A(\lambda_k)$ для всех $k \geqslant q_2(j) + 1$, а значит, $j \in A(\lambda_{q_1(j)})$. Полученное противоречие доказывает лемму.

Лемма 3. Если в допустимом решении задачи P_λ работа j помещена в ячейку q, то справедливо $q_1(j) \leqslant q \leqslant q_2(j)$.

Доказательство. Предположим противное. Пусть в некотором допустимом расписании работа j помещена в ячейку q, для которой справедливо либо $q < q_1(j)$, либо $q > q_2(j)$. Пусть $q < q_1(j)$. Тогда $q_1(j) > 0$ и по определению $j \in D(\lambda_{q_1(j)})$, а значит, работа j не может выполняться до $\lambda_{q_1(j)}$, что противоречит выбору ячейки q. Пусть $q > q_2(j)$. Тогда $q_2(j) < e$ и по определению $j \in A(\lambda_{q_2(j)+1})$, а значит, работа не может выполняться после $\lambda_{q_2(j)+1}$, что противоречит выбору ячейки q. Лемма доказана.

Для каждой ячейки $q \in \{0, \dots, e\}$ введем ее стоимость $f(q)$ по следующему правилу:

$$f(q) = \sum_{\substack{k \in K: \\ x(i_k^a) \leqslant q}} w_k - \sum_{\substack{k \in K: \\ x(i_k^d) \leqslant q}} w_k,$$

где $x(i_k^a)$ и $x(i_k^d)$ — номера крайних работ курса k в перестановке $\Lambda = (\lambda_1, \dots, \lambda_e)$. Данная величина определяет «вклад» работы в исходной целевой функции (6) в случае ее помещения в ячейку q. Обозначим через $q^*(j)$ первое число от $q_1(j)$ до $q_2(j)$, для которого достигается минимум f:

$$q^*(j) = \min\left\{ t \mid f(t) = \min_{q_1(j) \leqslant q \leqslant q_2(j)} f(q) \right\}. \tag{9}$$

Будем называть $q^*(j)$ оптимальной ячейкой для работы j, $j \in I$. Следующая лемма показывает, что, если одна работа должна предшествовать в расписании другой, то ее оптимальная ячейка будет иметь не больший номер, чем оптимальная ячейка для другой работы.

Лемма 4. Если в графе $G'(I, E')$ для двух некрайних работ j и g выполняется $j \to g$, то

a) $q_1(j) \leqslant q_1(g)$;

b) $q_2(j) \leqslant q_2(g)$;

c) $q^*(j) \leqslant q^*(g)$.

Доказательство. a) Поскольку $j \to g$, то $g \in D(j)$. Если $q_1(j) = 0$, то утверждение очевидно. Если $q_1(j) > 0$, то $j \in D(\lambda_{q_1(j)})$. Это означает, что $g \in D(\lambda_{q_1(j)})$. Тогда по определению $q_1(g)$ получаем $q_1(g) \geqslant q_1(j)$.

b) Поскольку $j \to g$, то $j \in A(g)$. Если $q_2(g) = e$, то утверждение очевидно. Если $q_2(g) < e$, то $g \in A(\lambda_{q_2(g)})$. Это означает, что $j \in A(\lambda_{q_2(g)})$. Тогда по определению $q_2(j)$ получаем $q_2(j) \leqslant q_2(g)$.

c) Пусть $q^*(j) > q^*(g)$. Принимая во внимание a) и b), получаем

$$q_1(j) \leqslant q_1(g) \leqslant q^*(g) < q^*(j) \leqslant q_2(j) \leqslant q_2(g).$$

Это означает, что обе ячейки $q^*(j)$ и $q^*(g)$ доступны для работ j и g. Это противоречит правилу выбора ячейки (9). Действительно, если $f(q^*(j)) = f(q^*(g))$, то ячейка $q^*(g)$ должна быть выбрана для обеих работ как более ранняя. Если же $f(q^*(j)) \neq f(q^*(g))$, то ячейка с минимальным значением стоимости f должна быть выбрана для обеих работ. Лемма доказана.

Для каждой ячейки $q \in \{0, \ldots, e\}$ введем множество работ I_q, для которых данная ячейка является оптимальной:

$$I_q = \{j \in I \setminus I_{ad} \ : \ q^*(j) = q\}, \ q \in \{0, \ldots, e\}.$$

Пусть $E_q \subset E$ — множество дуг, соединяющих вершины из I_q, $q \in \{0, \ldots, e\}$. Обозначим через $\bar{\pi}(I_q, E_q)$ произвольную топологическую сортировку графа $G_q(I_q, E_q)$, т.е. некоторую перестановку работ из I_q, удовлетворяющую частичному порядку, задаваемому множеством дуг E_q. В силу ацикличности исходного графа $G(I, E)$ топологическая сортировка любого его подграфа $G_q(I_q, E_q)$ существует. Следующая теорема показывает, что, упорядочив работы в каждой ячейке по отдельности, можно получить оптимальное расписание для задачи P_Λ по следующему правилу: сначала выполняются все работы из множества I_0, затем крайняя работа λ_1, затем все работы из множества I_1, затем крайняя работа λ_2 и т.д.

Теорема 2. Расписание $\pi^\Lambda = (\bar{\pi}(I_0, E_0), \lambda_1, \bar{\pi}(I_1, E_1), \lambda_2, \ldots, \lambda_e, \bar{\pi}(I_e, E_e))$ является оптимальным решением задачи P_Λ.

Доказательство. Расписание π^Λ является допустимым в задаче P_Λ. Действительно, рассмотрим любые две работы $i, j \in I \setminus I_{ad}$ такие, что $i \to j$. Если эти работы оказываются в одной ячейке, то ограничение предшествования выполняется в силу построения топологической сортировки всех работ из данной ячейки. Если же i и j оказываются в разных ячейках, то в силу леммы 4 имеем $q^*(i) < q^*(j)$, а значит, в расписании π^Λ работа i будет выполнена раньше работы j. Отношения предшествования между крайними работами

и всеми остальными работами выполнены в силу правила построения оптимальных ячеек.

Оптимальность решения следует из определения оптимальной ячейки. Действительно,

$$\sum_{j \in I \setminus I_{ad}} f(q^*(j))p_j = \sum_{j \in I \setminus I_{ad}} \min_{q_1(j) \leqslant q \leqslant q_2(j)} \left(\sum_{\substack{k \in K: \\ x(i_k^a) \leqslant q}} w_k - \sum_{\substack{k \in K: \\ x(i_k^d) \leqslant q}} w_k \right) p_j =$$

$$= \min_{\pi} \sum_{j \in I \setminus I_{ad}} W_j(\pi) p_j.$$

Поскольку при заданном порядке Λ вклад в целевую функцию крайних работ фиксирован и равен

$$\sum_{j \in I_{ad}} \left(\sum_{\substack{k \in K: \\ x(j) \geqslant x(i_k^a)}} w_k - \sum_{\substack{k \in K: \\ x(j) > x(i_k^d)}} w_k \right) p_j,$$

это означает, что расписание π^Λ, которое соответствует распределению работ по ячейкам, доставляет минимум целевой функции (6)–(7). Теорема доказана.

Таким образом, решение задачи P_Λ может быть сведено к вычислению оптимальной ячейки для каждой работы и упорядочению работ в каждой ячейке по отдельности. Общую схему поиска решения вспомогательной задачи описывает алгоритм 1. Оценим трудоемкость данного подхода. Необхо-

Algorithm 1 Процедура $Solv(\Lambda)$

$\pi^\Lambda := ()$
for all $q \in \{0, 1, \ldots, e\}$ **do**
    $I_q := \emptyset$
end for
Сгенерировать граф $G'(I, E')$ по перестановке $\Lambda = (\lambda_1, \ldots, \lambda_e)$
for all $j \in I \setminus I_{ad}$ **do**
    Вычислить $q^*(j)$
    $I_{q^*(j)} := I_{q^*(j)} \cup \{j\}$
end for
Построить $\bar{\pi}(I_0, E_0)$
$\pi^\Lambda := \bar{\pi}(I_0, E_0)$
for all $q \in \{1, \ldots, e\}$ **do**
    Построить $\bar{\pi}(I_q, E_q)$
    $\pi^\Lambda := \pi^\Lambda \cup (\lambda_q, \bar{\pi}(I_q, E_q))$
end for
Вернуть π^Λ

димо построить множества $A(\lambda)$, $D(\lambda)$ для каждой крайней работы λ, что суммарно потребует $O(nK)$ операций. Далее для каждой работы j необходимо определить границы $q_1(j)$, $q_2(j)$ и $q^*(j)$ — еще $O(nK)$ операций. Построение частичных расписаний работ в каждой ячейке — не более $O(n + |E|)$ операций [12].

5. Алгоритм решения задачи $1|prec|H$

Обозначим через B множество всех возможных перестановок крайних работ Λ, не противоречащих ограничениям предшествования в исходной задаче. Если количество курсов в задаче мало или в силу структуры графа $G(I, E)$ взаимный порядок крайних работ не допускает большого числа вариантов, возможен эффективный поиск решения задачи путем перебора перестановок крайних работ и решения вспомогательной задачи для каждой перестановки. Таким образом, схема решения задачи может быть представлена в виде алгоритма 2, где H^*_Λ — оптимальное значение целевой функции во вспомогательной задаче P_Λ, а H^* и π^* — оптимальное значение и оптимальное расписание исходной задачи $1|prec|H$ соответственно. Отметим, что в алгоритме 2 нет необходимости находить расписание для каждой рассматриваемой перестановки Λ, поскольку для вычисления значения H^*_Λ достаточно знать номера оптимальных ячеек для каждой работы.

Алгоритм решения задачи имеет трудоемкость $O(|B|(nK + |E|))$, где n — общее число работ, K — количество курсов, $|E|$ — количество ребер в графе отношения предшествования, $|B|$ — количество допустимых перестановок крайних работ курсов. Наибольший вклад в трудоемкость дает величина $|B|$. Максимально возможное значение $|B|$ равно $\frac{(2K)!}{2^K}$, когда рассматриваются все возможные перестановки крайних работ курсов без учета их взаимосвязей в графе отношения предшествования работ, однако в случае, например, плотных графов отношения предшествования величина $|B|$ может быть приемлемой для использования алгоритма 2 даже при большом числе курсов.

При проведении вычислительного эксперимента было проведено сравнение работы предложенного алгоритма с математическим пакетом ILOG CPLEX

Algorithm 2 Решение задачи $1|prec|H$

```
1: H* := +∞
2: for all Λ ∈ B do
3:     вычислить H*_Λ
4:     if H*_Λ < H* then
5:         H* := H*_Λ
6:         Λ* := Λ
7:     end if
8: end for
9: π* := Solv(Λ*)
10: Вернуть π*
```

22.1.0.0 [13]. Для применения данного пакета было использовано следующее представление задачи в виде задачи целочисленного линейного программирования:

$$\sum_{k\in K}\sum_{j\in I, j\neq i_k^a} w_k p_j x_{i_k^a,j} + \sum_{k\in K}\sum_{j\in I, j\neq i_k^d} w_k p_j x_{j,i_k^d} +$$

$$+ \sum_{k\in K}\left(p_{i_k^a} + p_{i_k^d} - \sum_{i\in I} p_i\right) \to \min,$$

$$x_{i,j} + x_{j,i} = 1 \ \forall i, j \in I;$$

$$x_{i,j} + x_{j,k} + x_{k,i} \geqslant 1 \ \forall i, j, k \in I;$$

$$x_{i,j} = 1 \ \forall (i,j) \in E;$$

$$x_{i,j} \in \{0,1\} \ \forall i, j \in I,$$

где переменная $x_{i,j}$, $i \neq j \in I$, принимает значение 1, если работа i выполняется раньше работы j, и значение 0 в противном случае. Такие переменные и ограничения задачи — стандартные для целочисленных постановок одноприборных задач с ограничениями предшествования (см., например, [14]).

Вычисления проводились на персональном компьютере (Intel Core i7-7700K, 4.2 GHz, 32,0 ГБ), алгоритм реализован на Python с использованием библиотеки NetworkX для работы с графами. В табл. 1 и 2 приведены результаты расчетов для случайно сгенерированных задач. В качестве весов курсов и продолжительностей работ выбирались произвольные целые числа из интервала $[1; 10]$. В качестве крайних работ курсов выбирались случайные вершины графа таким образом, чтобы не нарушались отношения предше-

Таблица 1. Результаты тестирования для разреженных графов

n	K	$\lvert E\rvert$	$\lvert B\rvert$	Algorithm 2	CPLEX
100	3	481	6	0,007	13,218
	5	467	4299	5,124	13,436
	7	525	26244	35,191	12,774
200	3	1864	6	0,034	106,282
	5	1942	150	0,516	105,152
	7	1990	870	3,169	102,967
300	3	4550	5	0,050	426,563
	5	4423	10	0,069	404,386
	7	4410	950	6,834	396,179
400	3	7765	1	0,022	>10 мин
	5	7662	4	0,051	>10 мин
	7	7746	280	3,362	>10 мин
500	3	12241	2	0,037	>10 мин
	5	12118	2	0,065	>10 мин
	7	12194	96	1,854	>10 мин

Таблица 2. Результаты тестирования для плотных графов

n	K	$\|E\|$	$\|B\|$	Algorithm 2	CPLEX
100	3	2514	2	0,009	11,144
	5	2541	4	0,054	10,717
	7	2515	3	0,025	10,777
	9	2487	8	0,048	10,668
200	3	10103	1	0,028	94,987
	5	10194	2	0,071	95,790
	7	10199	2	0,043	99,734
	9	10123	4	0,165	94,775
300	3	22846	1	0,034	386,899
	5	22943	1	0,039	398,339
	7	22933	6	0,225	378,841
	9	22845	4	0,156	400,153
400	3	40862	1	0,062	>10 мин
	5	40692	2	0,135	>10 мин
	7	40779	1	0,094	>10 мин
	9	40806	2	0,147	>10 мин
500	3	63657	1	0,090	>10 мин
	5	63424	1	0,098	>10 мин
	7	63676	1	0,136	>10 мин
	9	63802	3	0,316	>10 мин

ствования между первыми и последними вершинами каждого курса. Время работы алгоритма и решателя CPLEX ограничивалось десятью минутами.

Обозначения n, K, $|E|$, $|B|$, используемые в таблицах, совпадают с обозначениями, принятыми ранее в статье, а в колонках «Algorithm 2» и «CPLEX» указано время решения задач в секундах предложенным в статье алгоритмом и решателем CPLEX соответственно.

Как можно видеть из табл. 1, время работы алгоритма зависит от мощности множества $|B|$ сильнее, чем от общего количества работ. Так, даже при высокой размерности задачи, но небольшом количестве допустимых перестановок крайних работ алгоритм дает точное решение задачи за доли секунды. В случае же большого значения $|B|$ (см., например, задачу с $n = 100$, $K = 7$) время работы алгоритма значительно увеличивается. Решатель CPLEX, напротив, нечувствителен к изменению $|B|$ и K, но при повышении n время его работы сильно растет.

В табл. 2 приведены результаты работы алгоритма на задачах с более плотными графами. Здесь, как и ожидалось, количество допустимых перестановок крайних работ меньше, поэтому алгоритм нашел решения во всех задачах менее чем за секунду.

Таким образом, результаты вычислительного эксперимента подтверждают теоретическую оценку сложности разработанного алгоритма и показывают, что алгоритм может эффективно применяться в задачах с небольшой мощностью множества $|B|$, что соответствует случаю задач с плотным графом

отношения предшествования, или задач с небольшим количеством курсов или задач, в которых положения крайних работ зафиксировано друг относительно друга. В перечисленных случаях алгоритм позволяет быстро решать задачи высокой размерности.

6. Заключение

В статье рассмотрена одноприборная задача теории расписаний с ограничениями предшествования на работы, в которой необходимо минимизировать суммарное взвешенное время выполнения курсов (некоторых подмножеств работ). Доказана NP-трудность рассматриваемой задачи, а также предложен точный алгоритм ее решения, полиномиально зависящий от общего количества работ и позволяющий эффективно решать задачи при условии небольшого количества вариантов взаимного расположения крайних работ курсов. Дальнейшее направление исследований может касаться общей постановки задачи, когда крайние работы курсов не заданы однозначно, или задач управления проектом с несколькими ресурсами и данной целевой функцией.

СПИСОК ЛИТЕРАТУРЫ

1. *Brucker P.* Scheduling algorithms. Springer: Heidelberg, 2007.
2. *Лазарев А.А.* Теория расписаний. Методы и алгоритмы. М.: ИПУ РАН, 2019.
3. *Lazarev A., Khusnullin N., Musatova E., Yadrentsev D., Kharlamov M., Ponomarev K.* Minimization of the weighted total sparsity of cosmonaut training courses // Optimization and Applications. OPTIMA 2018. Communications in Computer and Information Science. 2019. P. 202–215.
4. *Harhalakis G.* Special features of precedence network charts // Eur. J. Oper. Res., Elsevier Publ. 1990. V. 49. No. 1. P. 50–59.
5. *Csébfalvi A.B., Csébfalvi G.* Hammock activities in project scheduling // Proceedings of the Sixteenth Annual Conference of POMS. 2005.
6. *Eliezer O.* A new bi-objective hybrid metaheuristic algorithm for the resource-constrained hammock cost problem (RCHCP) / Doctoral Dissertation. Pécs, 2011.
7. *El-Rayes K., Moselhi O.* Resource-driven scheduling of repetitive activities // Construction Management and Economics. 1998. V. 16. No. 4. P. 433–446.
8. *Vanhoucke M.* Work continuity constraints in project scheduling / Working Paper 04/265, Ghent University, Faculty of Economics and Business Administration, Belgium. 2004.
9. *Vanhoucke M., Van Osselaer K.* Work continuity in a real-life schedule: the Westerschelde Tunne / Working Paper 04/271, Ghent University, Faculty of Economics and Business Administration, Belgium. 2005.
10. *Graham R.L., Lawler E.L., Lenstra J.K., Rinnooy Kan A.H.G.* Optimization and approximation in deterministic sequencing and scheduling: a survey // Annals of Discrete Mathematics, Elsevier Publ. 1979. V. 5. P. 287–326.
11. *Lenstra J.K., Rinnooy Kan A.H.G.* Complexity of scheduling under precedence constraints // Oper. Res. 1978. V. 26. No. 1. P. 22–35.

12. *Cormen T.H., Leiserson C.E., Rivest R.L., Stein C.* Introduction to algorithms. MIT press, 2022.
13. IBM ILOG CPLEX Optimization Studio // URL: https://www.ibm.com/products/ilog-cplex-optimization-studio.
14. *Potts C.N.* An algorithm for the single machine sequencing problem with precedence constraint / Combinatorial Optimization II. Mathematical Programming Studies, Springer: Berlin, Heidelberg, 1980. V. 13. P. 78–87.

Статья представлена к публикации членом редколлегии П.Ю. Чеботаревым.

Поступила в редакцию 08.02.2023

После доработки 20.06.2023

Принята к публикации 20.07.2023