МЕТОД ВЕТВЕЙ И ГРАНИЦ ДЛЯ РЕШЕНИЯ ЗАДАЧИ МИНИМИЗАЦИИ ПЛАТЫ ЗА ВНЕШНИЕ РЕСУРСЫ

Мусатова Е. Г.[1], **Лазарев А. А.**[2]
(*ФГБУН Институт проблем управления им. В.А. Трапезникова РАН, Москва*)

Рассматривается задача построения расписания работ, выполняемых на одном приборе. Заданы отношения предшествования между работами, а также подмножества работ, требующих дополнительные внешние ресурсы, за аренду которых взимается плата. Для каждого внешнего ресурса однозначно определены крайние (первая и последняя) работы, выполняемые с использованием этого ресурса. Необходимо упорядочить работы, не нарушив отношения предшествования и минимизируя суммарные арендные выплаты. Доказана теорема об NP-трудности данной задачи в сильном смысле даже при условии одинаковой продолжительности всех работ и одинаковых цен на все внешние ресурсы. На основе свойств целевой функции для решения поставленной задачи предлагаются нижние оценки и метод ветвей и границ, в котором перебор ведется по допустимым перестановкам крайних работ, использующих внешние ресурсы. Проведенный вычислительный эксперимент показал эффективность предлагаемых нижних оценок целевой функции, позволяющих отсекать бесперспективные ветви в дереве поиска. Определен тип входных данных задачи, для которого разработанный метод работает лучше других известных вариантов точных методов, в которых перебор происходит на множестве всех работ. В частности, на задачах большой размерности при количестве внешних ресурсов меньше 20 данный метод оказывается эффективнее решателя CP Optimizer на базе программирования в ограничениях.

Ключевые слова: дискретная оптимизация, теория расписаний, метод ветвей и границ, алгоритмы, минимизация использования внешних ресурсов.

1. Введение

Рассматривается задача построения расписания обслуживания работ на одном приборе. Все работы доступны в начальный момент времени и должны быть выполнены без прерывания, при этом прибор может обслуживать не более одной работы одновре-

[1] *Елена Геннадьевна Мусатова, к.ф.-м.н.* (*nekolyap@mail.ru*).
[2] *Александр Алексеевич Лазарев, д.ф.-м.н., профессор* (*jobmath@mail.com*).

менно. Установлен частичный порядок выполнения работ. Некоторые работы требуют для своего выполнения внешние ресурсы. В качестве таких ресурсов могут выступать услуги сторонних специалистов или дополнительное оборудование. Предполагается, что каждый требуемый ресурс берется в аренду один раз в течение планируемого периода. Выплата за аренду каждого ресурса начинается в момент первого его использования и заканчивается с окончанием его последнего использования. Необходимо минимизировать суммарные выплаты за все арендованные ресурсы.

На практике минимизация выплат за использование дополнительных ресурсов чаще возникает в задачах построения расписания работ на нескольких параллельных приборах или в задачах управления проектом с ресурсными ограничениями [7]. В таких системах, как правило, появляется дополнительное ограничение на длительность всего проекта, которое не требуется в одноприборной задаче, рассматриваемой в статье. Примеры таких задач могут быть найдены в строительной сфере при возведении многоэтажных зданий, прокладке трубопроводов, строительстве туннелей [19, 20]. Часто в этих задачах весь проект может быть разбит на этапы, на каждом из которых требуется использование дополнительного арендуемого оборудования [18, 22].

Кроме того, существует ряд задач, которые формально не имеют отношения к минимизации стоимости арендуемых ресурсов, хотя целевая функция в них имеет тот же вид. В таких задачах необходимо минимизировать длительность выполнения некоторых групп работ из-за технологических ограничений в производственной сфере (например, в сталелитейной промышленности [1, 16]) или для повышения качества услуг в сфере обслуживания или образования (например, для уменьшение «растянутости» учебных курсов [12]).

В связи с разнообразием приложений имеется большая вариативность в используемой терминологии. Так, помимо задачи минимизации использования внешних ресурсов можно встретить такие названия, как задача «минимизация простоя ресурсов (бригад)» [19], задача с ограничениями на непрерывность работы (work continuity constraints) [22], где под ограничениями чаще всего понимаются мягкие ограничения, добавляемые в целевую функцию. Встречается также термин «hammock cost problem» [7], где под работой-гамаком (hammock activity) понимается искусственно введенная работа, начало и конец которой ассоциируются с началом и концом некоторых двух работ проекта [6, 9]. В этом случае длительность выполнения работы-гамака в точности совпадает с длительностью выполнения подмножества работ, требующих дополнительного ресурса, при условии, что первая и последняя работы данного подмножества определены однозначно. Близкой также является задача минимизации стоимости доступности ресурсов (resource availability cost problem, RACP) [14, 17], однако в такой постановке отсутствует предположение, что ресурс арендуется один раз за планируемый период до момента окончания его последнего использования.

Несмотря на важность данной целевой функции для практики, мы не обнаружили системных теоретических исследований подобного рода задач. Как правило, для каждой задачи, возникающей на практике, разрабатывается свой, чаще всего эвристический подход к решению.

Наиболее полное теоретическое исследование данной целевой функции проведено для одноприборной задачи [3], где рассмотрена задача с одним внешним ресурсом, работы поступают в различные моменты времени, а отношение предшествования отсутствует. Авторами рассмотрены несколько постановок, в которых или присутствует ограничение на длительность использования внешнего ресурса и минимизируется одна из четырех классических целевых функций теории расписаний (максимальное временнóе смещение, взвешенное количество опозданий, сумма моментов окончаний и взвешенная сумма моментов окончаний работ), или ограничивается значение одной из вышеуказанных целевых функций и минимизируется длительность использования ресурса. Для каждой из постановок либо предложен полиномиальный алгоритм решения, либо доказана NP-трудность. В [21] рассмотрена задача построения расписания для параллельных машин также с одним внешним ресурсом, но в качестве целевой функции выступает время окончания всех работ.

Стоит отметить, что задачи планирования работы одного прибора — наиболее изученная область в теории расписаний [5]. При этом сложность такой задачи напрямую зависит от вида целевой функции. Так, например, однориборная задача минимизации максимального временнóго смещения при наличии ограничений предшествования и одновременном поступлении работ является полиномиально разрешимой [10], а задача минимизации суммы взвешенных моментов окончаний при тех же ограничениях — NP-трудная в сильном смысле [11, 13]. Целевая функция, рассматриваемая в статье, близка по своей структуре к взвешенной сумме моментов окончаний работ. Однако, как правило, такая целевая функция рассматривается при условии неотрицательности весов, что влечет ее регулярность [4]. В нашем случае условие регулярности, т.е. неубывания по времени, не выполняется.

Ранее [2] нами уже была рассмотрена одноприборная задача с ограничениями предшествования и целевой функцией в виде суммарной платы за использование внешних ресурсов, доказана ее NP-трудность и предложен алгоритм, полиномиально зависящий от количества работ. Однако данный алгоритм включает полный перебор перестановок первых и последних работ каждого подмножества, требующего своего дополнительного ресурса. Поэтому алгоритм успешно работает для большого количества работ и малого количества внешних ресурсов. Данная работа является продолжением указанной статьи и посвящена разработке метода ветвей и границ, позволяющего значительно сократить перебор вариантов расположения первых и последних работ и решить задачи с большим количеством внешних ресурсов.

Статья организована следующим образом. В разделе 2 представлена математическая постановка задачи, а в разделе 3 доказана ее NP-трудность даже при одинаковых продолжительностях всех работ и одинаковых ценах на все ресурсы. В разделе 4 обсуждаются некоторые свойства задачи и кратко описаны ранее полученные результаты. Раздел 5 посвящен построению допустимых перестановок первых и последних работ множеств, требующих дополнительные ресурсы, и дерева поиска на их основе. Далее в разделе 6 представлен метод построения нижних оценок, позволяющих уменьшить перебор. Раздел 7 включает результаты вычислительного эксперимента и сравнение предложенного подхода с популярным решателем на базе программирования в ограничениях.

2. Постановка задачи

Пусть задано множество работ $J = \{1, \dots, n\}$, где для каждой работы $j \in J$ известна ее продолжительность обслуживания p_j. Все работы доступны в нулевой момент времени и должны быть обслужены на приборе без прерывания, при этом в каждый момент времени может обслуживаться только одна работа. Имеется частичный порядок выполнения работ, задаваемый ориентированным ациклическим графом $G(J, E)$, т.е. работа j не может быть выполнена раньше работы i, если $(i, j) \in E$. Будем писать $i \to j$, если в графе $G(J, E)$ есть путь из вершины i в вершину j, а через $A(j)$ и $D(j)$ обозначим соответственно множество всех работ, предшествующих работе j, и множество всех работ, которые должны быть выполнены только после выполнения работы j.

Пусть J_k, $k \in K$, – подмножество работ, требующих для своего выполнения дополнительного внешнего ресурса k, который берется в аренду по цене $w_k > 0$ за единицу времени, где K – множество всех видов внешних ресурсов, используемых в проекте. Каждый ресурс может быть арендован только один раз. Пусть выполнено следующее условие:

<u>Предположение 1.</u> *Каждое подмножество J_k, $k \in K$, содержит такие работы j_k^a, j_k^d, что $j_k^a \to j$ для всех $j \in J_k \setminus \{j_k^a\}$ и $j \to j_k^d$ для всех $j \in J_k \setminus \{j_k^d\}$.*

Работы j_k^a, j_k^d будем называть крайними работами ресурса k. Через J^{ad} обозначим множество всех крайних работ, а через J^a и J^d – множества всех крайних работ, являющихся начальными и конечными соответственно. При выполнении данного предположения продолжительность аренды ресурса полностью определяется его крайними работами, а задача может быть идентифицирована как hammock cost problem (чем больше работ будет обслужено между крайними работами ресурса, тем дольше будет продолжительность аренды ресурса, т.е. сильнее растянется «гамак», ассоциированный с выполнением крайних работ).

Очевидно, что независимо от порядка выполнения работ минимальная продолжительность проекта определяется как $\sum_{j\in J} p_j$. Перерывы между работами не улучшают значение целевой функции, поэтому будем полагать, что перерывов нет и первая работа начинает обслуживаться в нулевой момент времени. Произвольное расписание π для каждой работы $j \in J$ задает порядковый номер $\pi(j)$ ее обслуживания. Обозначим через $C_j(\pi)$ момент окончания работы j, равный $\sum_{i\in J:\pi(i)\leqslant\pi(j)} p_i$. Наша цель – найти допустимое расписание, удовлетворяющее ограничениям предшествования, с минимальными суммарными выплатами за аренду внешних ресурсов, т.е. минимизировать функцию

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

Обозначим данную задачу через $1|prec|H$ в соответствии с обозначениями, принятыми в теории расписаний [8], где 1 означает один прибор, «prec» – наличие частичного порядка между работами, а H – целевую функцию (1).

3. Сложность задачи

В [2] доказано, что данная задача является NP-трудной в сильном смысле даже в случае одинаковых продолжительностей работ. Однако, используя результаты полученные Лоулером в [11], можно доказать более сильное утверждение.

Теорема 1. *Задача* $1|prec|H$ *является NP-трудной, даже если* $p_j = 1$, $j \in J$, *и* $w_k = 1$, $k \in K$.

Доказательство. Рассмотрим специальную задачу минимизации суммарного взвешенного окончания работ на одном приборе, которая описывается следующим образом. Имеется множество n' работ J', которые должны быть обслужены без прерывания на одном приборе с соблюдением отношений предшествования, задаваемых графом $G'(J', E')$. Работы доступны с нулевого момента времени, продолжительность каждой работы равна 1, а вес w'_j каждой работы $j \in J'$ принадлежит множеству $\{a, a+1, a+2\}$. Необходимо построить расписание, минимизирующее функцию $F(\pi') = \sum\limits_{j \in J'} w'_j C'_j(\pi')$, где $C'_j(\pi')$ – момент окончания работы j в расписании π'. Доказано [11], что данная задача (обозначим ее через P_1) является NP-трудной в сильном смысле для любого целого a.

Пусть $a = -1$, т.е. веса могут принимать значения -1, 0, 1, тогда

$$F(\pi') = \sum_{j \in J' : w'_j = 1} C'_j(\pi') - \sum_{j \in J' : w'_j = -1} C'_j(\pi').$$

Обозначим через n_{-1} количество работ, имеющих вес -1, а через n_1 – количество работ с весом 1.

Сведем данную NP-трудную задачу к следующему частному случаю задачи $1|prec, p_i = 1, w_i = 1|H$. Множество работ задается как $J = J' \cup J^1 \cup J^{-1}$, где J^1 содержит n_1 работ, а множество J^{-1} включает n_{-1} работ, т.е. $n = n' + n_1 + n_{-1}$. Все работы имеют единичную продолжительность и единичные веса. Множество дуг в графе отношения предшествования $G(J, E)$ строится как $E = E' \cup E^1 \cup E^{-1}$, где новые подмножества дуг получены следующим образом. Работы множества J^1 образуют цепь, причем последняя работа цепи должна предшествовать всем корневым вершинам графа G' (см. рис. 1).

Множество E^1 есть множество дуг цепи плюс дуги, идущие к корневым вершинам. Аналогично строится множество дуг E^{-1}: работы множества J^{-1} соединяются последовательно в цепь, а все конечные вершины графа G' предшествуют первой вершине в этой цепи. Пусть в задаче имеется $n_1 + n_{-1}$ внешних ресурсов. Каждый ресурс характеризуется парой крайних работ: первой и последней работами, использующими этот ресурс. Крайние работы ресурсов выберем следующим образом. Для первых

n_1 ресурсов первые работы – это работы множества J^1, последние работы – работы j из множества J', веса w'_j которых в задаче P_1 были равны 1. Распределение, каким ресурсам какая пара крайних работ соответствует, проведем произвольным образом. Для следующих n_{-1} ресурсов в качестве первых работ выберем работы j из множества J' с весами $w'_j = -1$, а в качестве последних – работы из множества J^{-1}, сочетая их произвольным образом. Обозначим данную задачу через P_2.

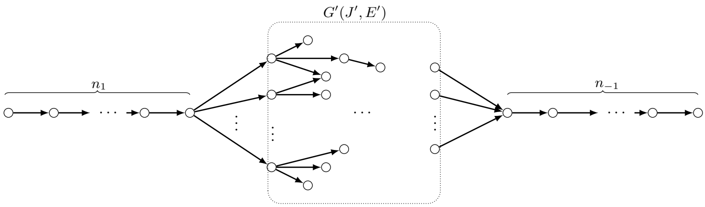

Рис. 1. Структура графа $G(J, E)$ из доказательства теоремы 1

В силу структуры графа $G(J, E)$ в любом допустимом расписании первыми будут выполнены n_1 работ из множества J^1, а последними – n_{-1} работ из J^{-1} в порядке, задаваемом ограничениями цепей. Поскольку перерывы в работе прибора не улучшают значение целевой функции, будем рассматривать только расписания без таких перерывов. Тогда расписание π задачи P_2 может быть представлено как последовательность трех перестановок работ из непересекающихся множеств $\pi = (\pi_1;\ \pi'';\ \pi_{-1})$, где π_1, π_{-1} – фиксированные последовательности работ из множеств J^1 и J^{-1} соответственно, а π'' – перестановка работ из множества J'.

Покажем, что если расписание π является оптимальным в задаче P_2, то его компонента π'' является оптимальным расписанием в задаче P_1. Несложно видеть, что π'' является допустимым расписанием в P_1, поскольку удовлетворяет всем ограничениям предшествования, задаваемым графом G'. Используя (1), получаем

$$H(\pi) = \Big(\sum_{j \in J' : w'_j = 1} C_j(\pi) - \sum_{j \in J^1} C_j(\pi) + n_1 \Big) +$$

$$+ \Big(\sum_{j \in J^{-1}} C_j(\pi) - \sum_{j \in J' : w'_j = -1} C_j(\pi) + n_{-1} \Big).$$

Учитывая, что $\sum\limits_{j \in J^1} C_j(\pi) = \frac{n_1(n_1+1)}{2}$ и

$$\sum_{j \in J^{-1}} C_j(\pi) = n_1 n_{-1} + n' n_{-1} + \frac{n_{-1}(n_{-1}+1)}{2},$$

а также тот факт, что выполнение работ из множества J' в задаче P_2 начинается на n_1 единиц времени позже, чем в задаче P_1, получаем

$$H(\pi) = \sum_{j \in J' : w'_j = 1} C'_j(\pi'') - \sum_{j \in J' : w'_j = -1} C'_j(\pi'') +$$

$$+ \frac{n_1^2 + n_1 + 3n_{-1} + n_{-1}^2}{2} + n' n_{-1} = F(\pi'') + const.$$

Таким образом, найдя оптимальное расписание в задаче P_2, мы получаем оптимальное расписание в P_1, что доказывает теорему.

4. Предварительные сведения

В [2] для решения задачи $1|prec|H$ предложен алгоритм сложности $O(|B|(n|K| + |E|))$, где B – множество допустимых взаимных перестановок крайних работ, а через $|\cdot|$ обозначена мощность множества. Идея метода основана на представлении целевой функции в следующем виде:

$$H(\pi) = \sum_{j \in J} W_j(\pi) p_j, \tag{2}$$

где

$$W_j(\pi) = \sum_{\substack{k \in K: \\ \pi(j) \geqslant \pi(j_k^a)}} w_k - \sum_{\substack{k \in K: \\ \pi(j) > \pi(j_k^d)}} w_k. \tag{3}$$

Иными словами, значение целевой функции зависит от взаимного порядка крайних работ, а каждая работа j дает свой вклад в целевую функцию $W_j(\pi)$ в зависимости от ее расположения относительно крайних работ. Идея метода заключается в переборе всех возможных взаимных перестановок крайних работ, и определении для каждой фиксированной перестановки оптимального расположения всех работ относительно крайних работ.

Пусть $\Lambda = (\lambda_1, \ldots, \lambda_e)$ – некоторая допустимая перестановка крайних работ. Будем рассматривать только расписания, в которых крайние работы выполняются в данном порядке. Заметим, что количество крайних работ e не обязательно равно $2|K|$, поскольку некоторые работы могут быть крайними для нескольких ресурсов. Обозначим данную задачу с дополнительным ограничением на порядок крайних работ через P_Λ.

Вспомогательная задача формируется следующим образом. Строится новый граф $G'(J, E')$, в котором добавляются дуги (λ_1, λ_2), ... ,$(\lambda_{e-1}, \lambda_e)$. Для каждой некрайней работы имеется (без учета ограничений предшествования) $e + 1$ вариант расположения между крайними работами. Будем говорить, что работа помещена в ячейку q, если она выполняется после крайней работы λ_q, но до крайней работы λ_{q+1}, $q \in \{1, \ldots, e-1\}$. Определим также ячейки 0 и e как варианты выполнения работы до всех крайних работ и после всех крайних работ соответственно. В [2] показано, что для каждой некрайней работы j может быть независимо выбрана оптимальная с точки зрения целевой функции ячейка. Для этого необходимо определить $q_1(j)$, $q_2(j)$ – соответственно минимальный и максимальный номера допустимых ячеек для работы j относительно ограничений предшествования, которые могут быть найдены по следующим формулам:

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

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

Тогда оптимальная ячейка для работы j определяется как

$$q^*(j) = \min\{t \mid f(t) = \min_{q_1(j) \leqslant q \leqslant q_2(j)} f(q)\}, \tag{4}$$

где $f(q)$ – стоимость ячейки q:

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

Тогда для каждого множества J_q некрайних работ, оптимальной для которых является ячейка q, может быть найден порядок выполнения этих работ $\bar{\pi}_q(I_q, E_q)$ путем построения топологической сортировки подграфа $\bar{G}(J_q, E_q)$, где $E_q \subset E$ – множество дуг, соединяющих работы из J_q. Доказана следующая

Теорема 2 [2, с. 162]. *Расписание*
$\pi^\Lambda = (\bar{\pi_0}(I_0, E_0), \lambda_1, \bar{\pi_1}(I_1, E_1), \lambda_2, \bar{\pi_2}(I_2, E_2), \ldots, \lambda_e, \bar{\pi}_e(I_e, E_e))$ *является оптимальным в задаче* P_Λ.

Таким образом, перебрав все возможные варианты перестановок крайних работ, можно найти оптимальное решение задачи. В [2] была показана эффективность данного подхода для плотных графов или небольшого количества ресурсов. Далее в статье предлагается использовать метод ветвей и границ для уменьшения перебора и возможности решать задачи с большим количеством внешних ресурсов.

5. Построение допустимых последовательностей крайних работ

Будем говорить, что расписание π соответствует частичной перестановке t крайних работ $\Lambda_t = (\lambda_1, \ldots, \lambda_t)$, $1 \leqslant t \leqslant e$, если выполняется

$$\pi(\lambda_1) < \ldots < \pi(\lambda_t) \text{ при } t > 1, \tag{5}$$

$$\pi(\lambda_t) < \pi(\lambda) \ \forall \lambda \notin \{\lambda_1, \ldots, \lambda_t\}. \tag{6}$$

Назовем частичную перестановку крайних работ допустимой, если существует соответствующее ей допустимое расписание. Пустая перестановка Λ_0 является допустимой, так как не добавляет никаких дополнительных ограничений в задачу.

Лемма 1. *Пусть частичная перестановка крайних работ* Λ_t*,* $0 \leqslant t \leqslant e - 1$*, является допустимой. Тогда перестановка*

$\Lambda_{t+1} = (\Lambda_t, \lambda_{t+1})$, *где* $\lambda_{t+1} \notin \Lambda_t$, *является допустимой тогда и только тогда, когда*

$$\forall \lambda' \in J^{ad} \setminus \{\lambda : \lambda \in \Lambda_t\} \ \ \lambda_{t+1} \notin P(\lambda'). \tag{7}$$

Доказательство. Необходимость выполнения условия (7) очевидна, так как при его нарушении условие (6) не совместимо с выполнением ограничений предшествования. Докажем достаточность от противного. Пусть выполнено условие (7), но не существует допустимого расписания, соответствующего частичной перестановке Λ_{t+1}. Для $t > 0$ это означает, что при добавлении в ациклический граф дуг $(\lambda_t, \lambda_{t+1})$ и (λ_{t+1}, λ) для всех $\lambda \in J^{ad} \setminus \{\lambda_1, \ldots, \lambda_{t+1}\}$ в нем появляются циклы. Любой из таких циклов обязательно содержит λ_{t+1}, а также либо λ_t, либо некоторую работу $\lambda' \in J^{ad} \setminus \{\lambda_1, \ldots, \lambda_{t+1}\}$ (либо обе эти работы). Если цикл содержит только λ_t, то в исходном графе существовал путь из λ_{t+1} в λ_t, что противоречит допустимости частичной перестановки Λ_t (условию (6) в частности). Если же цикл содержит λ', то это противоречит выполнению условия (7). Для $t = 0$ доказательство проводится по той же схеме, но в упрощенной форме. Лемма доказана.

Множество допустимых перестановок крайних работ будем представлять с помощью деревьев, каждое из которых имеет в качестве корневой вершины некоторую начальную работу ресурса, не являющуюся потомком других крайних работ. Пусть количество таких работ равно v. Каждая корневая вершина соединяется дугами с другими крайними работами, удовлетворяющими лемме 1 при $t = 1$, и т.д. В результате получается лес из v деревьев. Каждая цепь, соединяющая лист одного из деревьев с его корневой вершиной (вершиной первого уровня) задает допустимую перестановку e крайних работ. При этом цепь, соединяющая вершину t-го уровня, $1 \leqslant t \leqslant e - 1$, с корневой вершиной, задает допустимую частичную перестановку.

6. Построение нижних оценок и отсечение бесперспективных ветвей

Описанный выше лес допустимых перестановок определяет схему перебора, однако при большом количество вершин осуществление данного перебора занимает большое время. Для сокращения вычислений будем строить нижние оценки как для полных, так и для частичных перестановок.

Обозначим через Ω_k, $k \in K$, множество работ, лежащих на пути из j_k^a в j_k^d в графе $G(J, E)$, включая крайние работы j_k^a и j_k^d. Данное множество содержит работы, которые будут выполняться во время аренды k-го ресурса в любом расписании. Каждая работа имеет некоторый гарантированный вклад в целевую функцию в зависимости от того, в какие множества Ω_k она входит, т.е. во время аренды каких ресурсов она будет выполняться в любом расписании. Данный вклад равен

$$\underline{W}_j = \sum_{k \in K : j \in \Omega_k} w_k.$$

Тогда в качестве нижней оценки целевой функции выступает следующая величина:

$$\underline{H} = \sum_{j \in J} \underline{W}_j p_j.$$

Определим вначале для каждой работы, в какие множества Ω_k, $k \in K$, она включена. Пусть функция $out(j, k)$ принимает значение 1 для работы j, равной j_k^a, или если в графе G существует путь из работы j_k^a к работе j, и значение 0 в противном случае. Аналогично введем функцию $in(j, k)$, принимающую значение 1, если работа j равна j_k^d или существует путь из вершины j к j_k^d, и значение 0 в противном случае. Тогда $j \in \Omega_k$, если одновременно $out(j, k) = 1$ и $in(j, k) = 1$. Пусть J_{sort} – произвольная топологическая сортировка графа G, $\bar{J}_{sort}$ – сортировка, обратная J_{sort}. Введем обозначения для множества родительских вершин $Pr(i) = \{j \in J \mid (j, i) \in E\}$ и множества дочерних вершин $Ch(i) = \{j \in J \mid (i, j) \in E\}$. Алгоритм 1 описывает способ вычисления значений функций $out(\cdot, \cdot)$ и $in(\cdot, \cdot)$ для всех работ и всех ресурсов. Сложность алгоритма составляет $O(k(n + |E|))$ операций.

Алгоритм 1. (Вычисление значений $out(\cdot, \cdot)$ и $in(\cdot, \cdot)$))

```
for all k ∈ K do
    out(j_k^a, k) = 1
    in(j_k^d, k) = 1
end for
for all i ∈ J_sort do
    for all k ∈ K do
        for all j ∈ Pr(i) do
            if out(j, k) == 1 then
                out(i, k) = 1
                break
            end if
        end for
    end for
end for
for all i ∈ J̄_sort do
    for all k ∈ K do
        for all j ∈ Ch(i) do
            if in(j, k) == 1 then
                in(i, k) = 1
                break
            end if
        end for
    end for
end for
```

Рассмотрим частичную перестановку крайних работ $\Lambda_t = (\lambda_1, \ldots, \lambda_t)$. Если для некоторой работы $j \in J$ выполняется $j \in A(\lambda_i)$ хотя бы для одного $i \in \{1, \ldots, t\}$, то $q_2(j) \leqslant t$ и для данной работы уже может быть вычислена оптимальная ячейка по формуле (4) и соответствующая стоимость $W_j^*(\Lambda_t) = f(q^*(j))$, которая будет оптимальной для всех расписаний, соответствующих этой частичной перестановке. Обозначим множество всех таких работ через J_{Λ_t}. Тогда нижняя оценка для расписаний, соответствующих частичной перестановке Λ_t, может быть уточнена как

$$\underline{H}_{\Lambda_t} = \sum_{j \in J_{\Lambda_t}} W_j^*(\Lambda_t) p_j + \sum_{j \in J \setminus J_{\Lambda_t}} \underline{W}_j p_j. \tag{8}$$

По мере роста t нижняя оценка частичной перестановки будет уточняться:

$$\underline{H}_{\Lambda_{t+1}} = \sum_{j \in J_{\Lambda_t}} W_j^*(\Lambda_t) p_j + \sum_{j \in J_{\Lambda_{t+1}} \setminus J_{\Lambda_t}} W_j^*(\Lambda_{t+1}) p_j + \sum_{j \in J \setminus J_{\Lambda_{t+1}}} \underline{W}_j p_j,$$

а для $t = e$, т.е. для концевой вершины дерева, данная нижняя оценка $\underline{H}_{\Lambda_e}$ в точности совпадет с оптимальным значением задачи P_Λ для $\Lambda = (\lambda_1, \ldots, \lambda_e)$.

Таким образом, для отсечения бесперспективных ветвей можно действовать следующим образом. Пусть известно рекордное (минимальное) значение целевой функции H_{min} для некоторой полной перестановки крайних работ (т.е. значение нижней оценки для концевой вершины дерева). Данное значение H_{min} будем уточнять при достижении каждого очередного листа дерева. Если для некоторой промежуточной вершины на t-м уровне выполняется $\underline{H}_{\Lambda_t} > H_{min}$, то данная вершина может быть исключена из дальнейшего рассмотрения вместе со всеми исходящими из нее ветвями.

7. Вычислительный эксперимент

Метод ветвей и границ на основе разделов 5–6 был реализован на языке Python. Для получения начального значения величины H_{min} предварительно строилась произвольная топологическая сортировка графа. При выборе очередной крайней работы (при построении очередной ветви дерева поиска) использовалась следующая стратегия. Сначала выбирались работы из множества J^d, удовлетворяющие лемме 1, в порядке невозрастания весов ресурсов, а затем – допустимые работы из J^a в порядке неубывания весов ресурсов. Данный подход объясняется эмпирическим правилом: работы, использующие самые дорогие ресурсы, нужно начинать как можно позже, а заканчивать как можно раньше.

Данные для эксперимента генерировались следующим образом. Для заданных значений n и K продолжительности работ и веса внешних ресурсов генерировались случайным образом из интервала $[1, 10]$. Ориентированный граф строился с помощью библиотеки NetworkX, вероятность создания ребра задавалась равной $0{,}1$, а далее проводилась процедура переориентации некоторых ребер для исключения цикличности графа. Крайние работы ресурсов выбирались произвольным образом так, чтобы не нарушить ограничения предшествования работ.

В таблице 1 представлены результаты тестирования алгоритма на задачах с 300 работами и разными количествами ресурсов. Несложно видеть, что количество допустимых перестановок крайних работ увеличивается очень быстро с ростом числа внешних ресурсов.

Таблица 1. Характеристики метода ветвей и границ на задачах с 300 *работами*

K	Общее количество допустимых перестановок	Количество рассмотренных перестановок	Количество отсечений
2	2	1	1
3	1	1	0
4	6	4	2
5	4	1	3
6	6	2	4
7	18	5	7
8	72	12	17
9	3	3	0
10	144	6	26
11	1920	13	145
12	1824	7	94
13	141344	21	2467
14	3072	2	260
15	1244160	4	2423
16	9763200	82	9104
17	37440	20	2619
18	90961920	109	30288
19	51612288	47	86362
20	357696000	111	259410

При этом в предлагаемом точном методе количество полностью рассмотренных перестановок, т.е. количество обработанных листьев в дереве поиска, невелико, что достигается срабатыванием нижних оценок и последующим отсечением ветвей (см. последний столбец). Таким образом, можно сделать вывод об эффективности данного подхода по сравнению с перебором.

На втором этапе эксперимента было проведено сравнение предлагаемого метода ветвей и границ с решателем CP Optimizer 22.1.0 (с использованием библиотеки docplex.cp) на базе программирования в ограничениях от IBM ILOG CPLEX [23]. В последние годы программирование в ограничениях стало одним из самых мощных инструментов для решения задач теории расписаний, превосходящим подходы, основанные на целочисленном линейном программировании [15]. Для запуска решателя использовалась стандартная для одноприборных задач модель ограничений (см. алгоритм 2).

Время работы алгоритмов для каждой задачи ограничивалось 5 минутами. На рис. 2 указаны результаты тестирования предлагаемого метода ветвей и границ (BB) и решателя CP Optimizer (CP) на задачах с различными значениями n и K. Цвет ячеек характеризует количество возможных допустимых перестановок крайних работ в сгенерированных задачах.

Алгоритм 2. (CP модель задачи)

1: // ПЕРЕМЕННЫЕ:
2: $x_j \leftarrow \mathit{intervalVar}(size = p_j)\ \forall\ j = 1 \ldots n$ ▷ Интервальные переменные работ
3: // ОГРАНИЧЕНИЯ:
4: $\mathit{endBeforeStart}(x_j, x_i)\ \ \forall\ (i,\ j) \in E(G)$ ▷ Ограничения предшествования
5: $\mathit{noOverlap}([x_j\ \mid j = 1 \ldots n])$ ▷ Запрет перекрытий работ
6: // ЦЕЛЕВАЯ ФУНКЦИЯ
7: **minimize** $\sum_{k \in K} w_k \Big(\mathit{endOf}(x_{i_k^d}) - \mathit{startOf}(x_{i_k^a})\Big)$

n,K	2	5	10	15	20	25	30
100	B<C	B<C	B<C	-	-	-	-
150	B<C	B<C	B<C	-	-	-	-
200	B<C	B<C	B<C	B<C	-	-	-
250	B<C	B<C	B<C	B	-	-	-
300	B<C	B<C	B<C	B	-	-	-
350	B<C	B<C	B<C	B	B	-	-
400	B<C	B<C	B<C	B<C	-	-	-
450	B<C	B<C	B<C	B	-	-	-
500	B<C	B<C	B<C	B	B	-	-
600	B	B	B	B	B	-	-
700	B	B	B	B	B	B	-
800	B	B	B	B	B	-	-
900	B	B	B	B	B	-	-
1000	B	B	B	B	B	-	-

Цвет ячейки	Количество перестановок крайних работ
	<10
	10-1000
	1000-500000
	500000-50000000
	>50000000

Значения ячеек:

B<C	Оба алгоритма завершили работу, BB оказался быстрее CPLEX.
B	BB завешил работу, CPLEX не завершил работу.
-	Оба алгоритма не завершили работу.

Рис. 2. Сравнение BB и CPLEX при ограничении по времени счета в 5 минут

Во всех примерах, в которых оба алгоритма успевали закончить работу, BB находил решение раньше, чем CP. В задачах размерности выше 600 только BB укладывался в 5 мин. Таким образом, CP более чувствителен к увеличению размерности задачи. При этом отчетливо видна зависимость эффективности обоих алгоритмов от количества возможных перестановок крайних работ. Эксперимент показал, что на рассмотренном поле примеров BB демонстрирует бо́льшую эффективность, чем CP с точки зрения поиска гарантированного оптимального решения. Однако стоит отметить, что во всех случаях (кроме примеров $(n, K) = (100, 15), (1000, 25)$), когда оба алгоритма не успели завершить работу, значение целевой функции, полученное с помощью CP было меньше, чем значение, полученное BB. Таким образом, можно заключить, что CP быстрее находит лучшее значение функции (лучшую верхнюю оценку), но, судя по примерам, где BB успел завершить работу, тратит больше времени, чтобы доказать оптимальность решения, т.е. перебрать все не отсеченные вершины дерева поиска.

8. Заключение

В работе рассмотрена одноприборная задача минимизации платы за аренду дополнительных ресурсов. Доказана ее NP-трудность в сильном смысле при равенстве всех продолжительностей работ и цен за арендуемые ресурсы. Предложен метод ветвей и границ, основанный на переборе допустимых перестановок крайних работ, использующих дополнительные ресурсы. Вычислительные эксперименты показали конкурентоспособность разработанного алгоритма по сравнению с ведущим решателем на базе программирования в ограничениях. Дальнейшие направления исследований могут касаться использования данного подхода для разработки эффективных эвристик для решения многоприборных задач минимизации трат на использование арендуемых ресурсов или задач управления проектом при минимизации суммарной длительности выполнения подпроектов.

Литература

1. КИБЗУН А.И., РАССКАЗОВА В.А. *Модель целочисленного линейного программирования как математическое обеспечение системы оптимального планирования потокового производства на этапе оперативного графикования* // Автоматика и телемеханика. – 2023. – №5. – С. 113—132.
2. МУСАТОВА Е.Г., ЛАЗАРЕВ А.А. *Задача минимизации суммарной взвешенной длительности курсов для одного прибора с ограничениями предшествования* // Автоматика и телемеханика. – 2023. – №9. – С. 153–168.
3. BRISKORN D., DAVARI M., MATUSCHKE J. *Single-machine scheduling with an external resource* // European Journal of Operational Research. – 2021. – Vol. 293, No. 2. – P. 457–468.
4. BRUCKER P. *Scheduling algorithms.* – Springer: Heidelberg, 2007. – 371 p..
5. CHEN B., POTTS C.N., WOEGINGER G.J. *A review of machine scheduling: complexity, algorithms and*

approximability // Handbook of Combinatorial Optimization. – Boston, MA: Springer US, 1998. – P. 1493–1641.

6. CSEBFALVI A.B., CSEBFALVI G. *Hammock activities in project scheduling* // Proc. of the Sixteenth Annual Conf. of POMS. – Chicago, IL, 2005.
7. ELIEZER O. *A new bi–objective hybrid metaheuristic algorithm for the resource-constrained hammock cost problem (RCHCP).* – Doctoral Dissertation. – Pecs, 2011. – 111 p.
8. GRAHAM R.L., LAWLER E.L., LENSTRA J.K., KAN A.R. *Optimization and approximation in deterministic sequencing and scheduling: a survey* // Annals of discrete mathematics. – 1998. –Vol. 5 –P. 287–326.
9. HARHALAKIS G. *Special features of precedence network charts* // European Journal of Operational Research. – 1990. – Vol. 49, No. 1. – P. 50–59.
10. LAWLER E.L. *Optimal sequencing of a single machine subject to precedence constraints* // Management Science. – 1973. – Vol. 19, No. 5. – P. 544–546.
11. LAWLER E.L. *Sequencing jobs to minimize total weighted completion time subject to precedence constraints* // Annals of Discrete Mathematics. – 1978. – Vol. 2 – P. 75-–90.
12. LAZAREV A., KHUSNULLIN N., MUSATOVA E. et al. *Minimization of the weighted total sparsity of cosmonaut training courses* // Communications in Computer and Information Science. – 2019. – Vol. 974. – P. 202–215.
13. LENSTRA J.K., RINNOOY KAN A.H.G. *Complexity of scheduling under precedence constraints* // Operations Research. – 1978. – Vol. 26, No. 1. – P. 22–35.
14. MOHRING R.H. *Minimizing costs of resource requirements in project networks subject to a fixed completion time* // Operations Research. – 1984. – Vol. 32, No. 1. – P. 89—120.
15. NADERI B., RUIZ R., ROSHANAEI V. *Mixed-integer programming vs. constraint programming for shop scheduling problems: New results and outlook* // INFORMS Journal on Computing. – 2023. – Vol. 35, No. 4. – P. 817–843.

16. RASSKAZOVA V.A. *LIP model in solving RCPSP at the flow type production* // Communications in Computer and Information Science. – 2024. – Vol. 1913. – P. 75–88.
17. RODRIGUES S.B., YAMASHITA D.S. *An exact algorithm for minimizing resource availability costs in project scheduling* // European Journal of Operational Research. – 2010. – Vol. 206, No. 3. – P. 562–568.
18. TOMCZAK M., JASKOWSKI P. *Scheduling repetitive construction projects: Structured literature review* // Journal of Civil Engineering and Management. – 2022. – Vol. 28, No. 6. – P. 422–442.
19. VANHOUCKE M. *Work continuity constraints in project scheduling* // Journal of Construction Engineering and Management. – 2006. – Vol. 132, No. 1. – P. 14–25.
20. VANHOUCKE M. *Work continuity optimization for the Westerscheldetunnel Project in the Netherlands* // Tijdschrift voor economie en management. – 2007. – Vol. 52, No. 3. – P. 435–449.
21. ZHANG A., ZHEN T., CHEN Y., CHEN G. *An improved algorithm for parallel machine scheduling under additional resource constraints* // Optimization Letters. – 2023. – Vol. 17, No. 3. – P. 753–769.
22. ZOU X., WU G., ZHANG Q. *Work continuity constraints in repetitive project scheduling considering soft logic* // Engineering, Construction and Architectural Management. – 2021. – Vol. 28, No. 6. – P. 1713–1738.
23. https://www.ibm.com/docs/es/icos/22.1.0 (дата обращения: 15.04.2025).

BRANCH-AND-BOUND ALGORITHN FOR SOLVING PROBLEM OF MINIMIZING FEES FOR EXTERNAL RESOURCES

Elena Musatova, V.A. Trapeznikov Institute of Control Sciences of RAS, Moscow, Cand.Sc. (nekolyap@mail.ru).

Alexander Lazarev, V.A. Trapeznikov Institute of Control Sciences of RAS, Moscow, Doctor of Science, professor (jobmath@mail.ru).

Abstract: We consider a problem of scheduling jobs performed on a single machine. Precedence relations between jobs are established, as well as subsets of jobs that require additional external resources, for which a fee is charged. For each external resource, the extreme (first and last) job to be executed using that resource is uniquely determined. It is necessary to order the jobs without violating the precedence relations and minimizing total rent payments. We have proved that this problem is NP-hard in the strong sense, even if the processing times of all jobs are the same and the prices of all external resources are equal. Based on the properties of the objective function, lower bounds and the branch-and-bound method are proposed to solve the problem. In this method, enumeration is performed according to feasible permutations of extreme jobs using external resources. The computational experiment showed the efficiency of the proposed lower bounds of the objective function, which make it possible to cut off unpromising branches in the search tree. We also determined the type of input data for which the developed method is more successful than another known variant of exact methods that enumerate all jobs. In particular, for large-dimensional problems with fewer than 20 external resources, this method proves to be more efficient than CP Optimizer solver which is based on constraint programming.

Keywords: discrete optimization, scheduling, branch-and-bound method, algorithms, minimizing the use of external resources.

УДК 519.854.2
ББК 22.18

Статья представлена к публикации членом редакционной коллегии Е.Н. Хоботовым.

Поступила в редакцию 07.05.2025.
Дата опубликования 30.09.2025.