Российская Академия Наук

Автоматика и Телемеханика

Журнал основан в 1936 году
Выходит 12 раз в год

5
МАЙ
Москва
2020

Учредители журнала:
Отделение энергетики, машиностроения, механики и процессов управления РАН,
Институт проблем управления им. В.А. Трапезникова РАН (ИПУ РАН),
Институт проблем передачи информации им. А.А. Харкевича РАН (ИППИ РАН)

Адрес редакции: 117997, Москва, Профсоюзная ул., 65
Тел./факс: (495) 334-87-70
Электронная почта: redacsia@ipu.ru

Зав. редакцией *Е.А. Мартехина*

Москва
ООО «ИКЦ «АКАДЕМКНИГА»

Тематический выпуск

© 2020 г. А.А. ЛАЗАРЕВ, д-р физ.-мат. наук (jobmath@mail.ru)
(Институт проблем управления им. В.А. Трапезникова РАН, Москва)

МОДЕЛИ И МЕТОДЫ ТЕОРИИ РАСПИСАНИЙ

Вячеслав Сергеевич Танаев
(1940–2002)

DOI: 10.31857/S0005231020050013

Вячеслав Сергеевич Танаев родился 28 марта 1940 г. в д. Акулово Теблешского района Тверской области в семье военнослужащего и учительницы. Семья часто меняла место жительства, и Вячеславу пришлось учиться в школах с разными языками обучения (в Западной Украине, с 1947 г. в Ивано-Франковске, с 1950 г. в Крыму в Балаклаве, с 1952 г. в Симферополе). В 1957 г. после окончания средней школы в Симферополе В.С. Танаев поступил на физико-математический факультет Крымского педагогического института им. М.В. Фрунзе. Он возглавлял в институте и на факультете совет студенческого научного общества (СНО), участвовал во Всесоюзном слете председателей СНО в Москве. Будучи студентом, опубликовал первую научную работу. Во время учебы вначале получал стипендию им. Кирова,

а затем стипендию им. Ленина. В 1962 г. окончил институт и получил диплом с отличием по специальности «Учитель математики и черчения». Был направлен на работу в институт на кафедру математики.

Осенью того же года он поступил в аспирантуру Института тепло- и массообмена АН БССР, правда, через полгода у него сменился научный руководитель. За два с половиной года интенсивных исследований выполнил комплекс работ по теории расписаний, подготовил и в 1965 г. успешно защитил кандидатскую диссертацию.

После защиты диссертации В.С. Танаев перешел на работу в только что образованный Институт технической кибернетики АН БССР. Здесь в 1966 г. он организовал лабораторию дискретного программирования, основными научными направлениями которой были развитие теории расписаний и разработка методов решения сложных оптимизационных задач. С 1971 г. В.С. Танаев возглавляет лабораторию управляющих систем (переименованную позднее в лабораторию математической кибернетики). В лаборатории продолжались исследования в области теории расписаний и разрабатывались основы общей теории параметрической декомпозиции задач оптимизации. Под руководством и при непосредственном участии В.С. Танаева созданы методы решения ряда важных прикладных задач, возникающих в системах автоматизированного проектирования. За исследования в области параметрической декомпозиции оптимизационных задач и ее приложений В.С. Танаеву в 1978 г. присуждена ученая степень доктора физико-математических науки, а в 1980 г. он получил звание профессора.

В 1987 г. Танаева В.С. назначили директором Института технической кибернетики АН БССР – главного кибернетического центра Белоруссии, третьего по значимости в СССР (после Вычислительного центра АН СССР и Института кибернетики АН Украины, который тогда возглавлял академик Глушков В.М.). На этом посту наиболее полно раскрылся его талант организатора науки. В 1996 г. по инициативе В.С. Танаева на базе лабораторий и отделов Института технической кибернетики образован ряд научно-инженерных предприятий, которые вошли в состав созданного Научно-исследовательского объединения (НИО) “Кибернетика” НАН Беларуси. В.С. Танаева назначили генеральным директором НИО Кибернетика НАН Беларуси и оставили директором головной научной организации объединения – Института технической кибернетики. В 2002 г. это объединение преобразовано в Объединенный институт проблем информатики НАН Беларуси, генеральным директором которого был назначен В.С. Танаев.

За 40 лет творческой деятельности В.С. Танаев опубликовал более 130 научных работ, включая 10 монографий. Первая монография по теории расписаний была напечатана в СССР в 1975 г. [1], которая способствовала интенсивному развитию теории расписаний в СССР.

Одновременно он руководил филиалом кафедры математического обеспечения АСУ Белгосуниверситета при ИТК АН БССР, читал курсы лекций по исследованию операций, теории расписаний и методам оптимизации в БГУ. Им подготовлено 18 кандидатов наук, 7 его учеников защитили докторские диссертации. За выдающиеся научные достижения и плодотворную педагоги-

ческую деятельность В.С. Танаеву в 1995 г. присвоено почетное звание заслуженного деятеля науки Республики Беларусь. За большой вклад в работы по космической тематике организация “Ветераны космоса” (г. Королев) в 2000 г. наградила В.С. Танаева Почетным знаком “За освоение космоса”.

В 1994 г. В.С. Танаев избран членом-корреспондентом, а в 2000 г. – действительным членом Национальной академии наук Беларуси. В 1998 г. за цикл научных работ “Модели и методы теории расписаний” В.С. Танаеву и группе его учеников присуждена Государственная премия Республики Беларусь в области естественных наук. За монографию “Теория расписаний. Групповые технологии” В.С. Танаеву с соавторами по итогам конкурса 2001 г. присуждена премия Национальной академии наук Беларуси.

В последние годы жизни В.С. Танаев осуществлял большую научно-организационную работу, являясь координатором Республиканской программы фундаментальных исследований “Интеллект”, председателем научного совета “Информационные технологии”, научным руководителем Государственной программы фундаментальных исследований “Инфотех”, Государственной научно-технической программы “Информационные технологии”, двух совместных белорусско-российских программ “Космос – БР” и “Суперкомпьютер”, заместителем академика-секретаря Отделения физики, математики и информатики НАН Беларуси, заместителем главного редактора журнала “Весці НАН Беларусі. Серыя фізіка-матэматычных навук”, председателем совета по защите диссертаций при Институте технической кибернетики и членом советов по защите диссертаций при Институте математики НАН Беларуси и БГУ, президентом Белорусского общества исследования операций, действительным членом Institution of Electrical Engineers.

Вячеслава Сергеевича отличали высокая ответственность и организованность, простота в общении, доброта и искренность, широкая эрудиция и энциклопедическая образованность. Все это снискало ему всеобщее уважение, заслуженный авторитет и признание.

Умер Вячеслав Сергеевич 19 июля 2002 г. на 63-м году жизни...

Основные труды Танаева В.С.

1. *Танаев В.С., Шкурба В.В.* Введение в теорию расписаний. М.: Наука, 1975.

2. *Танаев В.С., Гордон В.С., Шафранский Я.М.* Теория расписаний: Одностадийные системы. М.: Наука, 1984.

3. *Танаев В.С.* Декомпозиция и агрегирование в задачах математического программирования. Минск: Наука и техника, 1987.

4. *Танаев В.С., Сотсков Ю.Н., Струсевич В.А.* Теория расписаний: Многостадийные системы. М.: Наука, 1989.

5. *Танаев В.С., Сотсков Ю.Н., Струсевич В.А.* Математические модели и методы календарного планирования. Минск: Унівеpсітэцкае, 1994.

6. *Танаев В.С., Ковалев М.Я., Шафранский Я.М.* Теория расписаний: Групповые технологии. Минск: Ин-т техн. кибернетики НАН Беларуси, 1998.

© 2020 г. Б.В. КУПРИЯНОВ, канд. техн. наук, kuprianovb@mail.ru
(Институт проблем управления им. В.А. Трапезникова РАН, Москва)

ОЦЕНКА И ОПТИМИЗАЦИЯ ПРОИЗВОДИТЕЛЬНОСТИ РЕКУРСИВНОГО КОНВЕЙЕРА

Сформулировано понятие производительности рекурсивного конвейера, и задача оптимизации распределения возобновляемых ресурсов решена как задача максимизации производительности путем сведения к задаче целочисленного линейного программирования. Дано определение рекурсивных функций вычисления расписания процесса для некоторого произвольного распределения ресурсов. Результаты могут быть использованы при проектировании конвейера или для вычисления граничных оценок при использовании комбинаторных алгоритмов построения расписания.

Ключевые слова: теория расписаний, рекурсивные конвейеры, производительность конвейера, балансировка конвейера.

DOI: 10.31857/S0005231020050025

1. Введение

В массовом производстве [1] или при проектировании производственных линий распределение ресурсов (станков, инструментов, людей) осуществляется до начала производственного процесса. Так, проектирование сборочного конвейера является долгосрочным решением и обычно требует крупных капиталовложений. Задача расчета необходимого количества ресурсов и их распределения возникает и в тех случаях, когда необходимо выполнение некоторого объема заказа к заданному сроку. Вычисление оптимального распределения ресурсов (если это возможно) до построения расписания процесса позволяет существенно упростить задачу построения расписания процесса. В публикациях по теории расписаний большое количество работ посвящено данному вопросу. Существуют работы, которые фокусируются на методах оптимизации балансировки линии и планирования ресурсов на этапе проектирования сборочной линии [2]. Ghosh и Gagnon [3] делают комплексный обзор и анализ различных методов проектирования и планирования монтажа систем. Большая часть работ сконцентрирована на балансировке сборочного конвейера (ALB). Для ALB модели Хельгесоном (Helgeson) и соавторами было предложено использование методов поиска оптимальных решений, таких как линейное программирование [4, 5] и целочисленное программирование [6]. В [7] рассматривается задача распределения ресурсов как задача оптимального быстродействия. В настоящее время существует большое количество работ по оптимизации планирования ресурсов для традиционных задач теории расписаний, например [8–11].

В статье рассматриваются конвейеры, описываемые ациклическим ориентированным графом и набором рекурсивных функций [12]. Примеры при-

ложений таких конвейеров приводятся в [13]. Вводится понятие производительности конвейера и решается задача максимизации производительности на множестве распределений дискретных возобновляемых ресурсов. Конечный этап решения сводится к задаче целочисленного линейного программирования.

Статья имеет следующее содержание. Во второй главе дано формальное определение рекурсивного конвейера, приведен пример конвейера и временной диаграммы, описаны основные свойства конвейера. В третьей главе вводится определение производительности рекурсивного конвейера и формулируется задача оптимального распределения ресурсов как задача целочисленного программирования. Далее рассматривается пять этапов решения задачи. Приведены модифицированные рекурсивные функции. Каждый этап сопровождается примером. В четвертой главе рассматривается пять примеров решения задачи.

2. Определения

Модель конвейера, отношения временного предшествования которого определяются рекурсивными функциями (далее для краткости просто конвейера), представляет собой связный ациклический ориентированный граф $G = (V, A)$ с единственной конечной вершиной. V – множество вершин графа и n – количество вершин. A – множество дуг графа, упорядоченных пар вида (v, w), где $v, w \in V$. Вершины графа помечены номерами от 1 до n таким образом, что первые n_0 $(1 \leqslant n_0 < n)$ вершин являются начальными, а вершина n конечной. Каждая вершина графа обозначает либо производственную операцию, обрабатывающую заказ с помощью возобновляемого ресурса, либо спусковую функцию, определяющую отношения временного предшествования операций и не потребляющей ресурсы. Под операциями будем подразумевать как производственные операции, так и спусковые функции. Для идентификации типа вершины помечены с помощью отображения *type*: $V \to E$, где $E = \{\mathbf{bop}, \mathbf{op}, \mathbf{and}, \mathbf{mul}, \mathbf{red}, \mathbf{get1}, \mathbf{get2}, \mathbf{put}\}$ – конечное перечислимое множество. Каждая константа множества E обозначает тип вершины графа, а сама вершина имеет соответствующую графическую нотацию (см. рис. 1,*а–ж*).

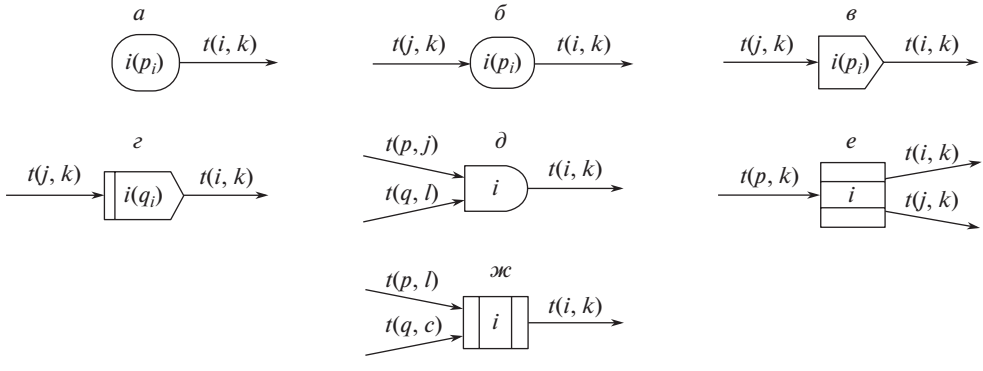


Рис. 1.

Из практики определено пять видов спусковых функций, имеющих применение для широкого круга конвейерных процессов. Однако можно определять новые исходя из конкретных потребностей. Для производственной операции и спусковых функций отношения предшествования описываются с помощью рекурсивных функций. Расписание выполнения операций строится с помощью суперпозиции рекурсивных функций, вычисляющих значение времени завершения обработки i-й операцией k-го заказа ($k \geqslant 0$). Для каждого элемента множества E своя функция, т.е. восемь функций и каждая вершина i графа в соответствии с ее типом ассоциируется с одной из этих функций. Рекурсивная функция $t(i,k)$ обеспечивает суперпозицию и является обращением к одной из них в соответствии с типом вершины.

$$t(i,k) = \begin{cases} bop(i,k), \text{ if } type(i) = \mathbf{bop}; \\ op(i,k), \text{ if } type(i) = \mathbf{op}; \\ and(i,k), \text{ if } type(i) = \mathbf{and}; \\ mul(i,k), \text{ if } type(i) = \mathbf{mul}; \\ red(i,k), \text{ if } type(i) = \mathbf{red}; \\ get1(i,k), \text{ if } type(i) = \mathbf{get1}; \\ get2(i,k), \text{ if } type(i) = \mathbf{get2}; \\ put(i,k), \text{ if } type(i) = \mathbf{put}. \end{cases} \tag{1}$$

Здесь имена рекурсивных функций для взаимно однозначного соответствия обозначаются теми же именами, что и типы вершин, но не жирным шрифтом.

Каждая вершина графа может иметь одну или две входных дуги в зависимости от типа вершины. Переход в процессе вычисления рекурсивной функции от вершины к вершине графа осуществляется с помощью следующих функций:

$pred(i)$ – определена для вершины, имеющей одну предшествующую вершину, и вычисляет номер j вершины графа такой, что существует единственная дуга $(v_j, v_i) \in A$;

(Следующие две функции определены для вершин, имеющих две предшествующие вершины. Условно будем считать: вершина 1 и вершина 2. Способ определения порядка в данном случае не имеет значения. Важно только, что он существует.)

$pred1(i)$ – вычисляет номер p 1-й вершины графа такой, что существует дуга $(v_p, v_i) \in A$;

$pred2(i)$ – вычисляет номер q 2-й вершины графа такой, что существует дуга $(v_q, v_i) \in A$.

Далее описываются все типы вершин: их назначение, тип и соответствующая типу рекурсивная функция. Все рекурсивные функции имеют два параметра: i – номер вершины графа и k – номер заказа. В каждый момент времени производственная операция может выполнять только один заказ и каждый ресурс может выполнять только одну операцию. Рекурсивные функции здесь сформулированы при условии, что каждой операции выделен один ресурс. Задача распределения ограниченного множества ресурсов и со-

ответствующие рекурсивные функции рассматриваются далее. Время начала функционирования конвейера полагается равным 0.

1. Начальная производственная операция (см. рис. 1,*а*) с номером $1 \leqslant i \leqslant n_0$ характеризуется временем выполнения p_i, которое является константой. Прерывание выполнения операции запрещено. Время завершения выполнения данной операцией k-го заказа определяется как время завершения выполнения $(k-1)$-го заказа плюс время выполнения операции p_i. Время выполнения нулевого заказа равно p_i.

$$bop(i,k):$$
$$bop = \begin{cases} bop(i,k-1) + p_i, \ \textbf{if } k > 0, \\ p_i, \ \textbf{if } k = 0. \end{cases}$$

2. Не начальная производственная операция (см. рис. 1,*б*) с номером $n_0 < i$ характеризуется временем выполнения p_i, которое является константой. Время завершения выполнения данной операцией k-го заказа определяется как максимум времен: завершения выполнения данной операцией $(k-1)$-го заказа и времени завершения выполнения предшествующей ей операцией k-го заказа, плюс время выполнения операции p_i. Время выполнения нулевого заказа равно времени выполнения нулевого заказа предшествующей операцией плюс p_i.

$$op(i,k):$$
$$j = pred(i),$$
$$op = \begin{cases} \max(t(j,k), op(i,k-1)) + p_i, \ \textbf{if } k > 0, \\ t(j,k) + p_i, \ \textbf{if } k = 0. \end{cases}$$

3. Спусковая функция and (см. рис. 1,*д*) вычисляет время завершения выполнения k-го заказа для двух предшествующих операций p и q.

$$and(i,k):$$
$$p = pred1(i), \quad q = pred2(i),$$
$$and = \max(t(p,k), t(q,k)).$$

4. Спусковая функция мультиплицирования (см. рис. 1,*в*) для каждого времени завершения предшествующей ей операции j вычисляет q_i времен завершения функции i. Это означает, что на одно выполнение операции j осуществляется q_i выполнений операции, для которой функция i является предшествующей.

$$mul(i,k):$$
$$j = pred(i), \quad l = \lfloor k/q_i \rfloor,$$
$$mul = t(j,l),$$

где $\lfloor x \rfloor$ обозначает целую часть x.

5. Спусковая функция редуцирования (см. рис. 1,*г*) является обратной по отношению к функции мультиплицирования, т.е. она вычисляет время завершения выполнения партии из q_i заказов.

$red(i,k)$:

$$j = pred(i), \quad l = (k+1)q_i - 1,$$
$$red = t(j,l).$$

6. Спусковая функция раздачи (см. рис. 1,*е*) имитирует разделение конвейерных операций на два потока. Для пользователя она выступает как одна функция, а в реализации она разбивается на две (условно верхнюю и нижнюю). Для верхнего варианта спусковая функция вычисляет времена завершений четных заказов, т.е. $t(i,0) = t(p,0),\ t(i,1) = t(p,2),\ t(i,2) = t(p,4),\ \ldots$.

$get1(i,k)$:

$$p = pred(i), \quad l = 2k,$$
$$get1 = t(p,l).$$

Для нижнего вычисляет времена завершения выполнения нечетных заказов, т.е. $t(j,0) = t(p,1),\ t(j,1) = t(p,3),\ t(j,2) = t(p,5),\ldots$.

$get2(i,k)$:

$$p = pred(i), \quad l = 2k+1,$$
$$get2 = t(p,l).$$

7. Спусковая функция приема (см. рис. 1,*ж*) является обратной по отношению к функции раздачи и имитирует слияние двух конвейеров в один. Проще всего это пояснить последовательностью значений $t(i,k)$:
$t(i,0) = t(p,0),\ t(i,1) = t(q,0),\ t(i,2) = t(p,1),\ t(i,3) = t(q,1),\ t(i,4) = t(p,2),$
$t(i,5) = t(q,2),\ldots$.

$put(i,k)$:

$$p = pred1(i),\ l = k/2,$$
$$q = pred2(i),\ c = (k-1)/2,$$
$$put = \begin{cases} t(p,l),\ \textbf{if } k = 0, \\ \max(put(i,k-1), t(q,c)),\ \textbf{if } k \bmod 2 = 1, \\ \max(put(i,k-1), t(p,l)),\ \textbf{if } k \bmod 2 = 0, \end{cases}$$

где $x \bmod y$ – остаток от деления x на y.

Вычисление времени завершения выполнения k-го заказа конвейером осуществляется с помощью обращения к рекурсивной функции $t(n,k)$ по формуле (1). Эта же формула вычисляет время завершения выполнения k-го заказа любой i-й операцией конвейера обращением $t(i,k)$. Рекурсивность в данном случае выступает в двух видах:

— значение функции для некоторого k зависит от значения этой функции для $(k-1)$ или другого меньшего значения;

— в случае суперпозиций функции она может обращаться к самой себе с другим значением аргумента (см. в разделе 4 пример 1).

Важно понимать, что рекурсивные функции определяют отношение временного предшествования и используются в дальнейшем только для вычисления расписания выполнения операций. Никакого отношения к управлению конвейером они не имеют. Управление конвейером аналогично классическим конвейерам, может осуществляться множеством способов и зависит от технологического решения. В разделе 4 в примере 1 показана модель конвейера, соответствующая ему суперпозиция рекурсивных функций и вычисление такой функции для некоторых значений аргументов.

Временная сложность вычисления расписания с помощью таких функций – $O(nk)$, где n – количество вершин графа, k – количество обработанных заказов.

Интервалом обработки k-го заказа i-й операцией конвейера $d(i,k)$ $1 \leqslant n$, $k \geqslant 1$ назовем продолжительность времени между моментами завершений i-й операции при обработке k и $(k-1)$ заказов:

$$d(i,k) = t(i,k) - t(i,k-1).$$

Очевидно, что $d(i,k) \geqslant p_i$. Величины $d(n,k)$ для последовательности значений k определяют промежутки времени, через которые с конвейера сходит продукция. В общем случае интервал операции зависит от номера заказа k. Для описанных выше функций предшествования в [14] показано, что в общем случае интервал операции i конвейера для первых ks_i заказов меняется по некоторому закону переходного процесса, далее процесс переходит в стационарный режим и интервал меняется по закону периодической функции. Каждую операцию i конвейера можно характеризовать следующим кортежем:

$$(t0_i, ks_i, ts_i, D_i, T_i),$$

где i – номер операции;

$t0_i$ – время завершения операции i при обработке нулевого заказа (при i, равном n, это будет время выхода первого изделия с конвейера);

ks_i – номер заказа, начиная с которого операция i переходит в стационарный режим выполнения;

ts_i – время, начиная с которого операция i переходит в стационарный режим выполнения;

D_i – определяется как сумма интервалов операции i за период;

T_i – период колебания интервала операции i, измеряется в количествах заказов и является безразмерной величиной.

Для стационарного режима можно ввести понятие направляющей времени завершения выполнения заказа $t(i,k)$ операции i – луча, выходящего из точки (ks_i, ts_i) на котором лежат точки

$$t(i,k) = ts_i + kD_i, \quad k = 0, T_i, 2T_i, \ldots.$$

Параметры $t0_i, ks_i, ts_i, D_i, T_i$ являются константами, и в случае рассмотрения конвейера $(i = n)$ для удобства будем обозначать их как t_0, k_s, t_s, D, T. В определении не уточняется, относится ли определение стационарного режима работы конвейера ко всем операциям или только к завершающей операции конвейера. В данной статье будем считать, что это относится ко всем операциям конвейера, т.е. после обработки заказа k_s все операции конвейера перешли в стационарный режим выполнения.

3. Решение задачи оптимизации

В данном разделе рассматривается решение задачи оптимизации распределения ресурсов для стационарного режима работы конвейера как задачи целочисленного линейного программирования.

Введем ряд обозначений для ресурсов:

J – количество множеств ресурсов $(1 \leqslant j \leqslant J)$. Каждое множество содержит ресурсы одного типа;

r_j – количество ресурсов в j-м множестве. Ресурс может иметь составную структуру, например, для выполнения операции нужен один станок и два человека. Набор ресурсов, обслуживающий одну операцию, будем называть комплектом и с каждой операцией связывать некоторое целое число комплектов;

a_{ij} – коэффициент комплектации i-й операции j-м ресурсом, например, если выполнение i-й операции требует наличия одного станка и трех рабочих, то $a_{i1} = 1,\ a_{i2} = 3$;

x_i – количество комплектов ресурсов, используемых i-й операцией, является целым числом без размерности. Если i-й операции поставлено в соответствие x_i комплектов, то операция может x_i раз выполняться с совмещением во времени;

W_i – производительность i-й операции конвейера, которая измеряется количеством выполнений операции в составе конвейера в единицу времени и является вещественным числом.

Интервал конвейера $d(n, k)$ определяет его производительность. Чем больше интервал конвейера, тем меньше его производительность. Каждая операция i конвейера [15] может быть охарактеризована кратностью выполнения ω_i. Данная величина показывает, сколько раз операция i выполняется в пересчете на одно выполнение конечной операции конвейера. Иначе, сколько раз выполняется операция i при производстве одного изделия конвейера. Кратность операции увеличивает время выполнения операции в ω_i раз и соответственно уменьшает производительность операции в ω_i раз. В общем случае производительность является величиной переменной и может вычисляться по формуле

$$W_i(k) = g_i \frac{x_i}{d(i,k)\omega_i},$$

т.е. прямо пропорционально зависит от количества комплектов ресурсов x_i и обратно пропорционально зависит от произведения интервала опера-

ции $d(i,k)$ на ее кратность ω_i, g_i является коэффициентом пропорциональности. В случае стационарного процесса можно рассматривать среднюю производительность (не зависящую от k) за период колебания интервала

$$W_i = g_i \frac{T_i x_i}{D_i \omega_i},$$

где D_i – приращение интервала за период, T_i – величина периода и $\frac{D_i}{T_i}$ – средний интервал i-й операции за период, для которого справедливо неравенство $\frac{D_i}{T_i} \geqslant p_i$. Далее в статье рассматривается только средняя производительность в стационарном режиме, которую для краткости будем называть производительностью. В связи с последним неравенством введем неравенство для производительности

$$W_i \leqslant g_i \frac{x_i}{p_i \omega_i}.$$

Поскольку p_i, g_i и ω_i являются константами, для упрощения введем новую константу c_i, которая вычисляется по формуле

$$c_i = \frac{g_i}{p_i \omega_i}$$

и является вещественным числом. В этом случае для производительности справедливо неравенство

$$W_i \leqslant c_i x_i.$$

В терминах введенных определений задача формулируется следующим образом. Пусть имеется конвейер, выполняющий n операций, каждая продолжительностью $p_i \geqslant 0$, $1 \leqslant i \leqslant n$. Имеется некоторое множество ресурсов, разбитое на J групп по r_j, $1 \leqslant j \leqslant J$ в каждой группе. В этом случае задача оптимального распределения ресурсов является задачей целочисленного линейного программирования с целевой функцией

$$W \underset{x_i}{\longrightarrow} \max$$

и системой ограничений

$$\sum_{i=1}^{n} a_{ij} x_i \leqslant r_j, \quad 1 \leqslant i \leqslant n, \quad 1 \leqslant j \leqslant J,$$

$$W_i \leqslant c_i x_i, \quad 1 \leqslant i \leqslant n,$$
$$W \leqslant W_i, \quad 1 \leqslant i \leqslant n,$$
$$1 \leqslant x_i, \quad 1 \leqslant i \leqslant n.$$

Здесь W– производительность конвейера. Ограничение $x_i \geqslant 1$ обусловлено тем, что при $x_i = 0$ конвейер не может функционировать. Решение задачи разбивается на 5 этапов, от модели в виде цепочки операций, далее путем

последовательного рассмотрения спусковых функций к решению модели общего вида.

Этап 1. Рассмотрим конвейер как цепочку из n производственных операций. Известно, что производительность такого конвейера определяется его узким местом, т.е. операцией с самым большим временем выполнения. С учетом данного свойства максимум производительности достигается максимизацией минимальной производительности операций конвейера. Решим эту задачу методом линейного программирования. Целевая функция W – производительность конвейера, которая равна минимальной производительности операций конвейера, т.е.

$$W = \min_{1 \leqslant i \leqslant n} W_i.$$

Для каждого ресурса можно составить неравенство ограничения на ресурс

$$\sum_{i=1}^{n} a_{ij} x_i \leqslant r_j, \quad 1 \leqslant i \leqslant n, \quad 1 \leqslant j \leqslant J.$$

Для производительности W_i i-й операции конвейера выполняется неравенство

$$W_i \leqslant c_i x_i, \quad 1 \leqslant i \leqslant n.$$

Запишем эту постановку задачи формально в виде задачи линейного программирования

$$W \underset{x_i}{\longrightarrow} \max,$$
$$W \leqslant W_i,$$
$$\sum_{i=1}^{n} a_{ij} x_i \leqslant r_j, \quad 1 \leqslant i \leqslant n, \quad 1 \leqslant j \leqslant J,$$
$$W_i \leqslant c_i x_i, \quad 1 \leqslant i \leqslant n,$$
$$1 \leqslant x_i, \quad 1 \leqslant i \leqslant n.$$

Все переменные x_i принимают целочисленные значения, W может быть вещественным. Таким образом, задача является разновидностью задачи целочисленного линейного программирования. Описанный метод рассмотрен в примере 3 раздела 4.

Модифицируем рекурсивные функции для случая, когда i-я операция располагает x_i комплектами ресурсов. Основу нового вида рекурсивных функций определяет тот факт, что если i-я операция обеспечена x_i количеством комплектов, то x_i выполнений этой операции могут пересекаться во времени. Для начальной операции (см. рис. 1,*а*) новые рекурсивные функции определяются следующим образом:

$$t(i,k) = p_i, \quad \text{если} \quad 1 \leqslant i \leqslant n_0 \quad \text{и} \quad 0 \leqslant k < x_i;$$
$$t(i,k) = t(i, k - x_i) + p_i, \quad \text{если} \quad 1 \leqslant i \leqslant n_0 \quad \text{и} \quad k \geqslant x_i.$$

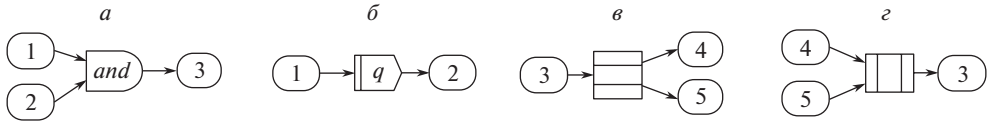


Рис. 2.

Если операция не начальная (см. рис. 1,*б*), то рекурсивные функции имеют вид

$$t(i,k) = t(j,k) + p_i, \quad \text{если} \quad i > n_0 \quad \text{и} \quad 0 \leqslant k < x_i;$$
$$t(i,k) = \max(t(j,k), t(i,k-x_i)) + p_i, \quad \text{если} \quad i > n_0 \quad \text{и} \quad k \geqslant x_i.$$

Изменение расписания при использовании модифицированных рекурсивных функций хорошо видно на временных диаграммах в системе координат. По оси абсцисс отложим номер заказа, а по оси ординат – время. Время выполнения i-й операции при обработке k-го заказа будет задаваться отрезком с координатами $(t(i,k) - p_i,\ t(i,k))$ и отмечаться номером операции. В примере 2 на рис. 5,*б,в* приведены временные диаграммы для рассматриваемого конвейера.

Этап 2. В том случае, когда в модели конвейера используется спусковая функция mul, она изменяет кратность ω_i выполнения следующих за ней операций. В этом случае кратности выполнения некоторых операций будут отличны от единицы. Это влечет за собой изменение коэффициентов c_i в соответствии с формулой (2). В остальном задача остается без изменений. В примере 5 рассматривается конвейер со спусковой функцией mul, решение задачи и соответствующие временные диаграммы.

Этап 3. Из [15] известно, что спусковые функции: редукции, раздачи и приема, по сути как и операция мультиплицирования, осуществляют либо увеличение кратности операции в два раза (get), либо уменьшение (red, put). Поэтому для данных спусковых функций метод решения будет аналогичным. Так, для моделей на рис. 2,*б,в,г* выполняются соотношения

$$\omega_1 = q\omega_2, \quad \omega_3 = 2\omega_4 = 2\omega_5, \quad \omega_8 = 2\omega_6 = 2\omega_7.$$

Для данных соотношений и существующих ресурсных ограничений решается задача максимизации для моделей, содержащих данные спусковые функции. Ограничения по ресурсам специфики не имеют, а ограничения по производительности имеют следующий вид:

для варианта а)

$$W \leqslant c_1 x_1;$$
$$W \leqslant c_2 x_2,$$

для варианта б)

$$W \leqslant c_3 x_3;$$
$$W \leqslant c_4 x_4;$$
$$W \leqslant c_5 x_5,$$

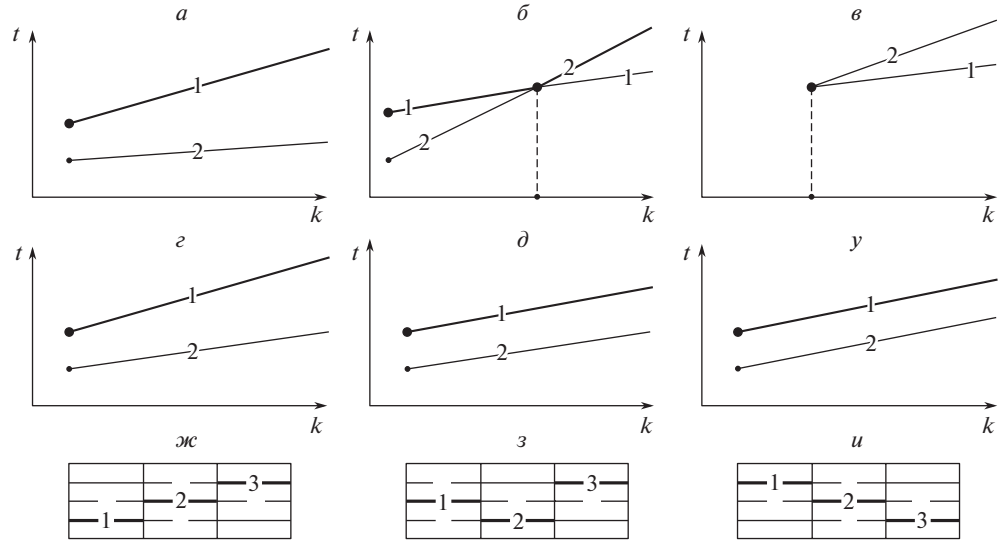


Рис. 3.

для варианта в)

$$W \leqslant c_6 x_6;$$
$$W \leqslant c_7 x_7;$$
$$W \leqslant c_8 x_8.$$

В примере 7 используются все спусковые функции, в том числе и данные.

Этап 4. Рассмотрим спусковую функцию *and*. На рис. 2,*а* представлена конструкция конвейера, в которой 1, 2 и 3 будем интерпретировать как некоторые конвейеры. Поскольку рассматривается стационарный конвейерный процесс и определены соответствующие направляющие функции $t(i,k)$, то на рис. 3,*а,б,в* отобразим направляющие для двух конвейеров 1–2 и функции *and*. График изменения направляющей спусковой функции *and* выделен жирной линией и в данном случае совпадает с графиком 1 (в соответствии с определением рекурсивной функции для *and*). Направляющие конвейеров 1 и 2 могут выходить из одной точки, быть параллельны или направляющая 2 может быть выше направляющей 1, но они не могут пересекаться. Пересечение направляющих 1 и 2, как показано на рис. 3,*б*, означает, что переходный процесс завершился в точке с абсциссой $k = k_s$. Следовательно ситуация сводится к случаю, когда обе направляющие выходят из одной точки рис. 3,*в*. Таким образом ситуация, показанная на рис. 3,*а*, является основной, а остальные сводятся к ней. Проанализируем данную ситуацию. Так как $t(i,k)$ функции *and* определяется $t(j,k)$ конвейера, который имеет меньшую производительность (в данном случае конвейер 1), то существующие ресурсы, если они есть, необходимо выделить конвейеру 1. На рис. 3,*г* показаны исходные графики, после выделения ресурсов конвейеру 1 его производительность вырастет и направляющая станет более пологой, как показано на рис. 3,*д*. Если ресурсы заимствованы не из свободного резерва, а у конвейера 2, то

производительность конвейера 2 упадет и его направляющая станет более крутой (см. рис. 3,*д*). Данные примеры и рассуждения показывают, что распределение ресурсов между конвейерами 1 и 2 должны быть такими, чтобы их производительности были равны и, как следствие, направляющие были параллельны (см. рис. 3,*е*).

Если после данного предела в какой-либо конвейер добавить ресурс, то направляющие пересекутся и ситуация вернется к исходной. Более того, если рассматривать соотношения производительностей трех конвейеров, то возможны три существенных варианта, представленные на рис. 3,*ж,з,и*. В варианте *ж* производительность конвейера определяется конвейером 1, соотношение двух других не имеет значения. В варианте *з* аналогичным образом производительность определяется конвейером 2, а в варианте *и* – конвейером 3.

Окончательным итогом данных рассуждений является равенство трех производительностей $W_1 = W_2 = W_3$ как условие оптимальности, а система ограничений на производительности выглядит следующим образом:

$$\begin{gathered} W \longrightarrow \max; \\ W \leqslant W_1; \\ W \leqslant W_2; \\ W \leqslant W_3. \end{gathered}$$

Этап 5. Прежде чем рассмотреть заключительный этап, сформулируем теорему.

Теорема 1. Если рекурсивный конвейер, состоит из n описанных выше операций и спусковых функций и W – производительность конвейера, то для любых двух операций $i \neq j$ $(1 \leqslant i, j \leqslant n)$ условие $W \longrightarrow \max$ означает, что

$$\left| \frac{x_i}{p_i \omega_i} - \frac{x_j}{p_j \omega_j} \right| \longrightarrow \min .$$

Доказательство теоремы фактически было рассмотрено выше для всех видов спусковых функций и операции. Данная теорема играет важную роль при формулировании на основании графической модели конвейера системы неравенств соответствующей задачи линейного программирования. Описание системы неравенств, относящейся к ограничениям на ресурсы, не представляет проблемы. На основании данной теоремы система неравенств для производительности будет выглядеть следующим образом:

$$W \leqslant \frac{x_i}{p_i \omega_i},$$

где i $(1 \leqslant i \leqslant n)$ является номером производственной операции (не спусковой функции). Поставленная задача полностью решена. Для иллюстрации метода в примере 5 рассматривается оптимизация распределения ресурсов для конвейера, содержащего производственные операции и все описанные спусковые функции.

4. Примеры решения задач

Пример 1. Рассмотрим пример конвейера, представленный на рис. 4,*а*, на рис. 4,*б* представлена соответствующая конвейеру суперпозиция рекурсивных функций, вычисляющая время завершения конвейером выполнения k-го заказа.

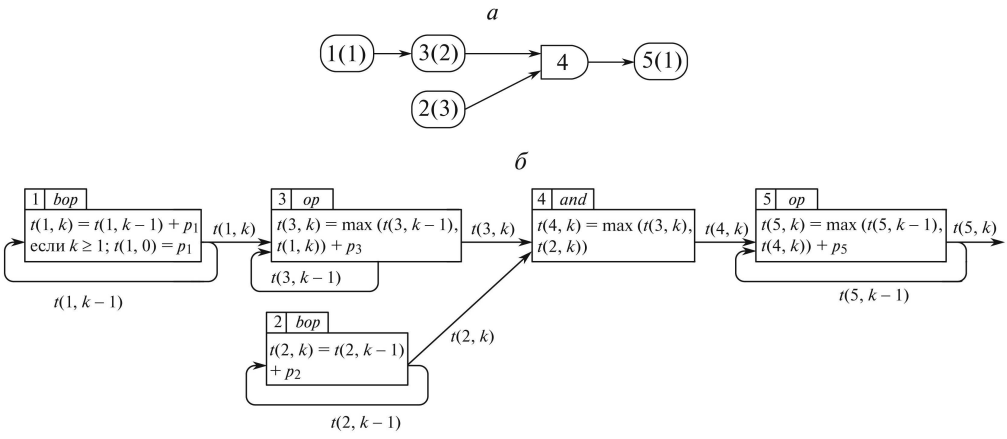


Рис. 4.

Ниже приведен пример вычисления времени завершения выполнения 5-й операцией 2-го заказа $t(5,2)$ для представленной модели. Для каждой операции – одна исполнительная машина. Другие параметры: $n = 5$, $n_0 = 2$. Приведем последовательность вычисления рекурсивных функций:

$$t(5,2) = \max(t(4,2), t(5,1)) + 1;$$
$$t(5,1) = \max(t(4,1) + t(5,0)) + 1;$$
$$t(5,0) = t(4,0) + 1;$$
$$t(4,2) = \max(t(2,2), t(3,2));$$
$$t(4,1) = \max(t(2,1), t(3,1));$$
$$t(4,0) = \max(t(2,0), t(3,0));$$
$$t(3,2) = \max(t(1,2), t(3,1)) + 2;$$
$$t(3,1) = \max(t(1,1), t(3,0)) + 2;$$
$$t(3,0) = t(1,0) + 2;$$
$$t(2,2) = t(2,1) + 4;$$
$$t(2,1) = t(2,0) + 4;$$
$$t(2,0) = 4;$$
$$t(1,2) = t(1,1) + 1;$$
$$t(1,1) = t(1,0) + 1;$$
$$t(1,0) = 1.$$

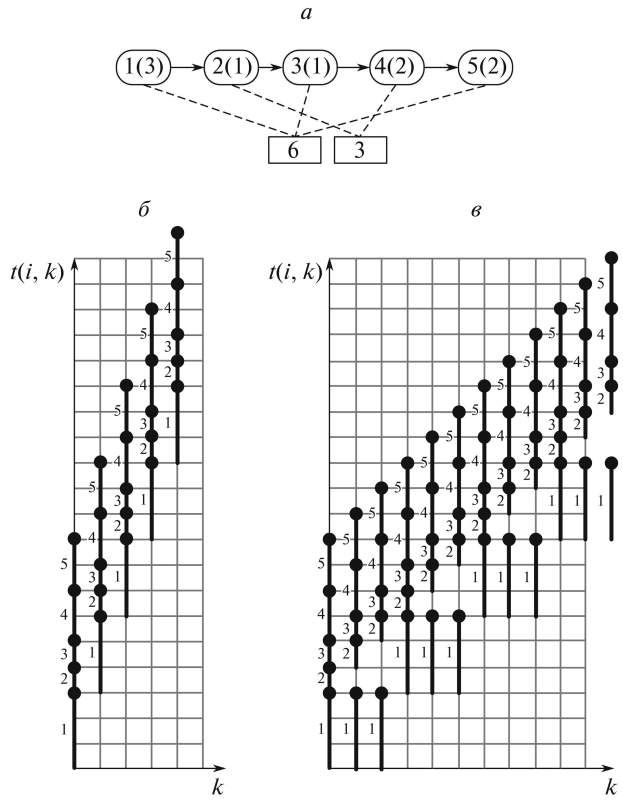


Рис. 5.

Пример 2. Пусть у нас имеется конвейер в виде цепочки из пяти операций (см. рис. 5). Времена выполнения операций указаны в скобках. Имеется два комплекта ресурсов. Первый – 6 единиц, второй – 3. Использование ресурсов операциями указаны на рисунке пунктирными линиями. В комплекты операций входит не больше чем по одному ресурсу (т.е. все $0 \leqslant a_{ij} \leqslant 1$). Целевая функция $W \longrightarrow \max$. Система ограничений будет выглядеть следующим образом:

а) ограничения на ресурсы $\begin{cases} x_1 + x_3 + x_5 \leqslant 6; \\ x_2 + x_4 \leqslant 3; \end{cases}$

б) поскольку для решения задачи оптимизации представляют интерес не абсолютные значения производительностей, а относительные, коэффициенты производительности c_i для удобства преобразуем в целые числа по формуле

$$c_i = \frac{lcm(c_1, c_2, \ldots c_5)}{c_i},$$

где lcm – наименьшее общее кратное. Осуществляя данное преобразование c_i, сохраняем пропорции между коэффициентами, но преобразуем их в целые числа. Вычисленные таким образом коэффициенты представлены в табл. 1.

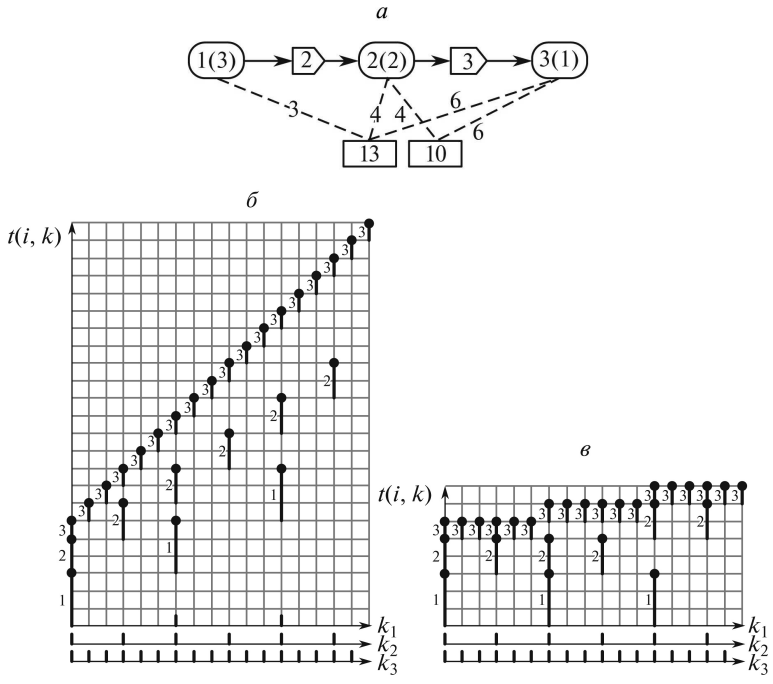


Рис. 6.

Ограничения на производительность будут выглядеть следующим образом:

$$W \leqslant 2x_1;$$
$$W \leqslant 6x_2;$$
$$W \leqslant 6x_3;$$
$$W \leqslant 3x_4;$$
$$W \leqslant 3x_5.$$

Решение данной системы дает следующий результат:

$$x_1 = 3,\ x_2 = 1,\ x_3 = 1,\ x_4 = 2,\ x_5 = 2,\ W = 6.$$

Построенная на основании вычисленного распределения ресурсов и новых рекурсивных функций временная диаграмма обработки двенадцати заказов будет иметь вид, представленный на рис. 5,*в*. На рис. 5,*б* представлена диаграмма для случая, когда каждая операция использует по одному комплекту ресурсов.

Пример 3. Рассмотрим пример конвейера рис. 6,*а*. В данном случае присутствуют спусковые функции mul и кратности выполнения операций будут отличны от единицы.

Таблица 1. Параметры линейного конвейера

Номер операции i	1	2	3	4	5
Время выполнения операции t_i	3	1	1	2	2
Коэффициент производительности c_i	1/3	1	1	1/2	1/2
Приведенный коэффициент производительности c_i	2	6	6	3	3

Таблица 2. Параметры конвейера с функцией мультиплицирования

Номер операции i	1	2	3
Время выполнения операции t_i	9	4	1
Кратность операции ω_i	1/6	1/3	1
Приведенная кратность ω_i	1	2	6
Коэффициент производительности c_i	1/9	1/8	1/6
Приведенный коэффициент производительности c_i	8	9	12

В табл. 2 представлены параметры модели. В общем случае кратности являются дробными величинами, и перейти к целым приведенным значениям можно по формуле

$$\omega_i = \frac{lcm(\omega_1, \omega_2, \ldots, \omega_5)}{\omega_i'},$$

где ω_i' – исходная кратность операции. Задача линейного программирования будет иметь следующий вид:

$$W \longrightarrow \max;$$
$$\begin{cases} x_1 + x_2 + x_3 \leqslant 13; \\ x_2 + x_3 \leqslant 10; \end{cases}$$
$$W \leqslant 4x_1;$$
$$W \leqslant 3x_2;$$
$$W \leqslant 2x_3.$$

Решение данной системы дает следующий результат:

$$x_1 = 3,\ x_2 = 4,\ x_3 = 6,\ W = 12.$$

Рекурсивные функции вычисления $t(i,k)$ остаются прежними. На рис. 6,*б,в* показаны две диаграммы: вариант диаграммы *б*, когда каждая операция имеет один комплект ресурсов, и вариант *в*, когда ресурсы распределены в соответствии с найденным решением. В общем случае каждая операция должна отображаться на отдельной временной диаграмме, однако для наглядности их имеет смысл совмещать, при этом для каждой операции рисовать свою ось абсцисс.

Пример 4. Рассмотрим пример модели, представленной на рис. 7,*а*. Из предыдущих рассуждений следует, что операции 1, 2 должны иметь равную производительность, а так как функция *and* включена последовательно с операцией 3, то они все должны быть равны между собой для выполнения условия оптимальности. Производительность конвейера должна быть максимизирована. Параметры модели представлены в табл. 3. Целевая функция $W \longrightarrow \max$. Система ограничений будет выглядеть следующим образом:

а) ограничения на ресурсы $\begin{cases} x_1 + x_2 + x_3 \leqslant 5; \\ x_2 + x_3 \leqslant 2; \end{cases}$

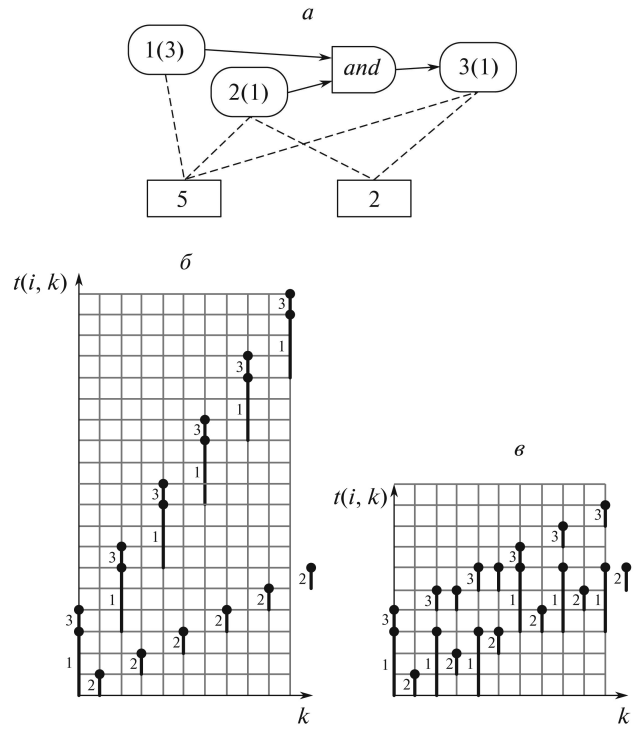


Рис. 7.

б) ограничения на производительность

$$W \leqslant x_1;$$
$$W \leqslant 3x_2;$$
$$W \leqslant 3x_3.$$

Решение данной системы дает следующий результат:

$$x_1 = 3,\ x_2 = 1,\ x_3 = 1,\ W = 3.$$

На рис. 7,*б,в* показаны диаграммы выполнения конвейера *а* – с одним комплектом на каждую операцию, а *б* – в соответствии с вычисленным распределением ресурсов.

Ось абсцисс у всех операций общая. Отрезки, соответствующие операции 2, смещены вправо от своих делений. Это сделано для того, чтобы не сливались диаграммы разных операций.

Таблица 3. Параметры конвейера с функцией and

Номер операции i	1	2	3
Время выполнения операции p_i	3	1	1
Кратность операции ω_i	1	1	1
Приведенный коэффициент производительности c_i	1	3	3

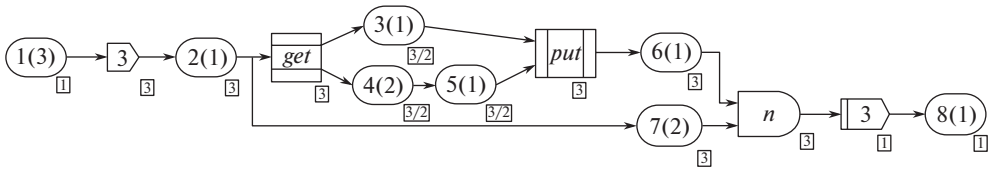


Рис. 8.

Пример 5. На рис. 8 представлен пример конвейера, использующий все рассмотренные спусковые функции. Справа снизу у каждой операции представлена неприведенная кратность ее выполнения. Целевая функция определяется как $W \longrightarrow \max$. Используется три множества ресурсов. В первом 44 единицы, во втором 26 и в третьем 12. Каждый комплект может содержать только по одному ресурсу. В табл. 4 представлены характеристики конвейера и матрица использования ресурсов операциями конвейера. Система ограничений на ресурсы выглядит следующим образом

$$\begin{cases} x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7 + x_8 \leqslant 44; \\ x_1 + x_2 + x_7 + x_8 \leqslant 26; \\ x_3 + x_4 + x_5 \leqslant 12. \end{cases}$$

Ограничения на производительность в соответствии с теоремой следующие:

$$\begin{gathered} W \leqslant c_1 x_1; \\ W \leqslant c_2 x_2; \\ W \leqslant c_3 x_3; \\ W \leqslant c_4 x_4; \\ W \leqslant c_5 x_5; \\ W \leqslant c_6 x_6; \\ W \leqslant c_7 x_7; \\ W \leqslant c_8 x_8. \end{gathered}$$

Решение задачи имеет следующий вид:

$$x_1 = 6,\ x_2 = 6,\ x_3 = 3,\ x_4 = 6,\ x_5 = 3,\ x_6 = 6,\ x_7 = 12,\ x_8 = 2,\ W = 12.$$

Таблица 4. Характеристики конвейера и матрица использования ресурсов

Номер операции i	1	2	3	4	5	6	7	8
Время выполнения операции p_i	3	1	1	2	1	1	2	1
Приведенная кратность операции ω_i	2	6	3	3	3	6	6	2
Приведенный коэффициент производительности c_i	2	2	4	2	4	2	1	6
Использование 1-го ресурса	1	1	1	1	1	1	1	1
Использование 2-го ресурса	1	1	0	0	0	0	1	1
Использование 3-го ресурса	0	0	1	1	1	0	0	0

5. Заключение

Разработанная в статье методика решения задачи распределения ресурсов позволяет в автоматическом режиме на основе графической модели конвейера и матрицы использования операциями ресурсов формировать исходные данные для задачи целочисленного линейного программирования. Решение данной задачи позволяет вычислить верхнюю оценку времени изготовления k изделий как частное от деления числа изделий на производительность конвейера при условии вычисления параметров в реальных величинах. Полученные результаты могут быть использованы не только при непосредственном проектировании конвейера, но и осуществлять перераспределение ресурсов при настройке конвейера на новый заказ. Еще одно возможное применение метода – получение граничных оценок для других, более сложных и эффективных методов оптимизации в теории расписаний. Решение задачи в данной постановке открывает перспективу для решения соответствующей двойственной задачи линейного программирования. Следует отметить, что полный список спусковых функций, приведенный в [12, 14], включает дизъюнкцию *or*. Однако для данной функции отсутствует общее решение задачи оптимизации распределения ресурсов. Поэтому она в данной статье не рассматривается.

СПИСОК ЛИТЕРАТУРЫ

1. *Сачко Н.С.* Организация и оперативное управление машиностроительным производством. Минск: ООО «Новое знание», 2005.
2. *Rekiek B., Dolgui A., Delchambre A., Bratcu A.* State of art of optimization methods for assembly line design // Ann. Rev. Control. 2002. No. 26. C. 163–174.
3. *Ghosh S., Gagnon R.J.* A comprehensive literature review and analysis of the design, balancing and scheduling of assembly systems // Int. J. Product. Res. 1989. V. C-27. P. 637–670.
4. *Helgeson W.B., Salveson M.E., Smith W.W.* How to balance an assembly line // Techni. Report, Carr Press. 1954. New Caraan, Conn.
5. *Bowman E.H.* Assembly line balancing by linear programming // Oper. Res. 1960. No. 8(3). P. 385–389.
6. *Salveson M.E.* The assembly line balancing problem // J. Indust. Engineer. 1955. No. 6. P. 18–25.
7. *Бурков В.Н.* Распределение ресурсов как задача оптимального быстродействия // АиТ 1966. Т. 27. Вып. 7. С. 119–129.
8. *Ranjbar M., Kianfar F., Shadrokh S.* Solving the resource availability cost problem in project scheduling by path relinking and genetic algorithm // Appl. Math. Comput. 2008. Т. 196. Вып. 2. С. 879–888.
9. *Kaabi J., Harrath Y.* A survey of parallel machine scheduling under availability constraints // Int. J. Comput. Inform. Technol. 2014. Т. 3. Is. 2. P. 238–245.
10. *Боброва Е.А., Сервах В.В.* Построение циклических расписаний при наличии параллельных машин // Дискретный анализ и исследование операций. 2017. № 1. Т. 24. С. 5–20.
11. *Мезенцев Ю.А., Эстрайх И.В.* Эффективный параметрический алгоритм оптимизации расписаний параллельных систем с заданным расписанием начала обслуживания // Научн. вест. НГТУ. 2018. Т. 72. № 3. С. 87–106.

12. *Куприянов Б.В.* Рекурсивные конвейерные процессы – основные свойства и характеристики // Экономика, статистика и информатика. Вестн. УМО. 2015. № 1. С. 163–170.

13. *Куприянов Б.В.* Применение модели конвейерных процессов рекурсивного типа для решения прикладных задач // Экономика, статистика и информатика. Вестн. УМО. 2014. № 5. С. 170–179.

14. *Куприянов Б.В.* Метод эффективного анализа модели рекурсивного конвейерного процесса // АиТ. 2017. № 3. С. 63–79.

    Kupriyanov B.V. Method of Efficient Analysis of the Recursive Conveyor Process Models // Autom. Remote Control. 2017. V. 78. No. 3. C. 435–449.

15. *Куприянов Б.В.* Вычисление некоторых производственных характеристик рекурсивного конвейера // Открытое образование. 2016. № 1. Т. 20. С. 11–16.

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

Поступила в редакцию 17.07.2019

После доработки 23.10.2019

Принята к публикации 28.11.2019

© 2020 г. А.Б. ДОЛГИЙ, д-р техн. наук (alexandre.dolgui@imt-atlantique.fr),
(Высшая национальная школа горного дела и телекоммуникаций Бретани и земель Луары — ИМТ Атлантик, Нант),
Г.М. ЛЕВИН, д-р техн. наук (levin@newman.bas-net.by),
Б.М. РОЗИН, канд. техн. наук (rozin@newman.bas-net.by),
(Объединенный институт проблем информатики НАН Беларуси, Минск)

СТРУКТУРНО-ПАРАМЕТРИЧЕСКАЯ ОПТИМИЗАЦИЯ КОМПЛЕКСА ПЕРЕСЕКАЮЩИХСЯ МНОЖЕСТВ ОПЕРАЦИЙ В УСЛОВИЯХ НЕСТАЦИОНАРНОГО СПРОСА[1]

Рассматривается задача оптимизации агрегирования операций, программы выпуска и интенсивностей выполнения операций при групповой серийной обработке заданного семейства продуктов на многопозиционной реконфигурируемой производственной системе в заданных (временных) интервалах при нестационарном детерминированном спросе. Допускается отложенный спрос. Агрегирование операций неизменно на весь период планирования, состав группы и интенсивности выполнения операций обработки неизменны в пределах интервала, но могут меняться от интервала к интервалу.

В качестве целевой функции используется планируемая прибыль. Инвестиционные затраты определяются вариантом агрегирования. Материальные и временные затраты на выпуск группы продуктов в каждом интервале зависят от агрегирования операций, интенсивностей их выполнения и затрат на реконфигурацию оборудования. Логистические затраты включают стоимость хранения невостребованных продуктов, упущенную добавленную стоимость и штрафы за неудовлетворенный спрос.

Предложен трехуровневый декомпозиционный метод решения задачи.

Ключевые слова: производственная система, группа продуктов, агрегирование операций, интенсивность обработки, нестационарный спрос, максимизация прибыли, декомпозиционный метод.

DOI: 10.31857/S0005231020050037

1. Введение

Задачам планирования процессов выполнения комплексов операций в системах различного назначения в последние десятилетия уделялось значительное внимание [1–9]. В ряде публикаций (в том числе в работах [7–9] авторов настоящей статьи) предлагаются модели и методы решения ряда задач, связанных с оптимизацией управления интенсивностью выполнения комплекса взаимосвязанных операций в предположении, что структура этого комплекса, а также структура реализующей его системы уже определены.

Вместе с тем значительный научный и практический интерес представляет также разработка моделей и методов решения более сложных задач совместной оптимизации как структуры комплекса операций в многопозиционных

[1] Авторы благодарны региональному правительству Земель Луары (Франция) за финансовую поддержку этого проекта.

производственных системах групповой обработки продуктов, так и программы выпуска семейства продуктов наряду с интенсивностями выполнения операций комплекса. Под «операцией» понимается набор взаимосвязанных шагов обработки, рассматриваемый как неделимое действие, выполняемое на одной рабочей станции системы посредством общего исполнительного органа. Под интенсивностью операции подразумевается некоторый параметр, определяющий время выполнения единицы ее объема.

В данной статье исследуется одна из ситуаций, когда структура комплекса определяется выбираемым вариантом агрегирования его операций в непересекающиеся группы (блоки операций), каждая из которых выполняется посредством своего исполнительного органа системы, причем все операции одного блока в любой момент времени выполняются с одной и той же интенсивностью. Объединение операций в блоки, как правило, способствует снижению инвестиционных затрат на производственную систему и на ее обслуживание, но одновременно приводит к возрастанию текущих эксплуатационных материальных и временных затрат на выполнение каждой из операций комплекса в отдельности. Последнее объясняется тем, что объединение операций в блоки исключает возможность индивидуального выбора для каждой из них наилучших (с точки зрения текущих затрат) интенсивностей их выполнения.

Динамически изменяющиеся потребности рынка диктуют необходимость модификации от одного (временного) интервала к другому состава группы выпускаемых продуктов посредством реконфигурации системы.

Классификации задач планирования производства партий продуктов с учетом затрат на их выпуск и хранение посвящены, в частности, обзоры [10, 11] и др.

Большая часть исследований в области детерминированных задач планирования выпуска продуктов на конечном горизонте с динамически изменяющимся спросом и ограничениями на мощность производственной системы касается выпуска продуктов одного наименования [12]. Вместе с тем большое внимание уделяется также многопродуктовым задачам [13]. Поскольку классические постановки обоих типов этих задач являются NP-трудными [14–16], большинство предложенных алгоритмов их решения являются эвристическими (см., например, [17–19]). Здесь особо следует отметить эвристики, предполагающие использование при их формулировке и реализации методов математического программирования ([18] и др.), а также эвристики, основанные на методе «ветвей и границ» [19].

Для решения однопродуктовых задач с динамически изменяющимся спросом часто применялись методы динамического программирования [20–22]. В последнее время внимание многих исследователей однопродуктовых задач с ограниченной мощностью системы привлекали вполне полиномиальные многошаговые аппроксимационные схемы [12], позволяющие находить приближенное решение задачи с заданной относительной погрешностью оптимального значения целевой функции.

Исследуемая в статье комплексная задача заключается в выборе варианта агрегирования операций в блоки, в определении для каждого временного интервала состава группы, количества выпускаемых групп и интенсивностей

выполнения операций, максимизирующих в совокупности планируемую прибыль при ограничениях на общую длительность выполнения комплексов операций в каждом временном интервале.

Компонента рассматриваемой задачи, связанная с планированием выпуска семейства продуктов, относится к задачам среднесрочного планирования серийного выпуска системой ограниченной мощности групп переменного состава продуктов заданной номенклатуры с детерминированным динамическим спросом. Невыполненные заказы по каждому из продуктов остаются в системе и могут быть удовлетворены на последующих интервалах (допускается отложенный спрос).

В качестве основных особенностей исследуемой постановки можно отметить следующие. Выбранный вариант агрегирования операций в блоки остается неизменным на всем периоде планирования. Как состав группы продуктов, так и интенсивности операций не изменяются в рамках текущего интервала, но могут быть изменены в следующем интервале. Объемы спроса на различные продукты семейства в различных интервалах не связаны между собой. Решения о количестве выпускаемых групп в любом интервале принимаются одновременно с выбором интенсивностей операций обработки продуктов группы, определяющих как возможность такого размера выпуска, так и стоимость самого выпуска. Следует отметить, что если пропорции спроса на различные продукты группы не изменяются от интервала к интервалу, то такую задачу можно свести к однопродуктовой, где в качестве продукта рассматривается группа в целом. Таким образом, рассматриваемая задача занимает промежуточное место между многопродуктовыми и однопродуктовыми задачами.

Рассмотренные в [7, 9] задачи являются фрагментами исследуемой, получаемыми при фиксации варианта агрегирования операций на единственном временном интервале и при некоторых дополнительных предположениях. Наличие в исследуемой задаче комбинаторной составляющей, связанной с поиском наилучшего агрегирования операций, а также дискретной составляющей, связанной с выбором варианта состава группы, делает эту задачу достаточно сложной, требующей специальных методов решения. Ниже предлагается один из возможных подходов к разработке таких методов.

2. Общая постановка задачи и ее математическая модель

Рассматривается одна из задач, возникающих при проектировании производственных систем для группового выпуска продуктов d заданной номенклатуры D на периоде $\Im$ планирования, состоящем из n временных интервалов с длительностями T_t, в условиях, когда известен ожидаемый спрос λ_{dt} на продукт $d \in D$ в интервале $t \in T = \{1, \ldots, n\}$. Система состоит из ряда специализированных рабочих станций, на каждой из которых может выполняться некоторое множество операций. Операции, выполняемые на различных станциях, считаются различными. В дальнейшем J – множество всех операций в системе.

Продукты в систему поступают последовательно. Входная последовательность в интервале t состоит из циклически повторяющихся идентичных под-

последовательностей (групп) $\pi_t = (d_{t1}, \ldots, d_{tr}, \ldots, d_{t\bar{h}(\pi_t)})$ из $\bar{h}(\pi_t)$ продуктов $d_{tr} \in D$, причем отдельные продукты могут входить в группу π_t в количестве $h_d(\pi_t)$ экземпляров. Каждый продукт группы последовательно один за другим проходит обработку на каждой станции системы в порядке нумерации станций, причем в каждый момент времени на каждой станции обрабатывается один продукт. Работа системы состоит из последовательности тактов. В каждом такте при заданной группе π_t на каждой рабочей станции параллельно выполняются множества операций над соответствующими (станции, группе и моменту времени) продуктами. По завершении такта все находящиеся в системе продукты синхронно перемещаются на следующие (для каждого из них) станции, с последней станции сходит готовый продукт, а на первую станцию поступает очередной продукт из входной последовательности. Затем выполняется очередной такт. Обработка одной группы π_t в интервале t состоит из $\bar{h}(\pi_t)$ тактов. Эта последовательность тактов образует цикл обработки группы π_t в интервале t. В общем случае цикл может содержать идентичные такты, но для простоты изложения будем считать все такты цикла различными. В дальнейшем $I(\pi_t) = \{1, \ldots, i, \ldots, \bar{h}(\pi_t)\}$ – множество номеров тактов обработки группы π_t продуктов. Без ограничения общности будем считать, что в интервале t в такте 1 цикла на первой рабочей станции обрабатывается продукт d_{t1} текущей группы π_t. Тогда в такте i операция j будет выполняться для продукта $d_{t\chi(\pi_t,i,j)} \in D$ этой группы, где $\chi(\pi_t, i, j) = 1 + \mathrm{mod}((\bar{h}(\pi_t) + i - \mathrm{mod}(k(j), \bar{h}(\pi_t))), \bar{h}(\pi_t))$ и $k(j)$ – номер станции, на которой выполняется операция $j \in J$.

Поскольку в разных тактах на одной и той же станции обрабатываются, вообще говоря, разные продукты операциями из одного и того же набора операций, то в данном случае цикл обработки группы представляет собой выполнение последовательности пересекающихся множеств операций, причем все операции одного множества выполняются параллельно. Эти множества образуют совокупности операций соответствующих тактов. Следует отметить, что одни и те же операции для разных продуктов могут иметь различные параметры, а для некоторых продуктов отдельные операции могут вообще не выполняться.

При проектировании системы операции могут агрегироваться в один либо несколько блоков операций. Каждый блок выполняется с единой интенсивностью посредством одного общего для операций блока исполнительного органа.

Агрегирование операций способствует, как правило, снижению капитальных затрат на систему и затрат на ее обслуживание, но одновременно может приводить к возрастанию текущих материальных и временных затрат на выполнение каждой из агрегируемых операций в отдельности. Последнее объясняется, в частности, тем, что такое агрегирование операций исключает возможность индивидуального выбора для каждой из них наилучших (с точки зрения производственных затрат) интенсивностей их выполнения.

Возможности агрегирования операций задаются семейством W непересекающихся неединичных подмножеств из множества J, каждое из которых является потенциальным блоком операций, причем любое подмножество $w \in W$

может содержать такие операции, выполнение которых для некоторых продуктов не требуется. В данной работе ограничимся случаем, когда каждое из множеств операций $w \in W$ может выполняться либо только полностью агрегированным (т.е. как один блок), либо полностью разагрегированным (т.е. каждая операция автономна). В этом случае множество возможных вариантов агрегирования представимо множеством Q допустимых значений двоичного вектора $\mathbf{q} = (q_w | w \in W)$, где $q_w = 0$ при агрегированном выполнении операций множества $w \in W$ и $q_w = 1$ в противном случае.

В работе исследуется ситуация, когда структура системы (число рабочих станций, множества операций, выполняемых на каждой станции, и выбранный вариант $\mathbf{q}$ их агрегирования) остается неизменной на всем периоде планирования, в то время как состав группы π_t продуктов, планируемое количество x_t выпуска групп и интенсивности z_{jt} выполнения операций $j \in J$ могут изменяться от интервала к интервалу. Ограничимся случаем, когда интенсивность z_{jt} выполнения каждой операции (а значит, и блока операций) для каждого интервала выбирается однократно, не зависит от такта, в составе которого эта операция выполняется, и не изменяется во времени.

Искомыми параметрами в рамках рассматриваемой проектной задачи являются вариант $\mathbf{q}$ агрегирования операций, вектора $\boldsymbol{\pi} = (\pi_1, \ldots, \pi_t, \ldots, \pi_n)$ и $\mathbf{x} = (x_1, \ldots, x_t, \ldots, x_n)$ планируемых к выпуску групп продуктов и планов выпуска этих групп на периоде планирования, а также вектор $\mathbf{z} = (\mathbf{z}_t = (z_{jt} | j \in J) | t \in T)$ интенсивностей выполнения операций в различных интервалах.

Диапазон $Z_{jd} = [z_{1jd}, z_{2jd}]$ возможных значений интенсивности z_{jt} выполнения операции $j \in J$ для продукта $d \in D$ предполагается заданным. Поскольку интенсивность z_{jt} выполнения операции $j \in J$ в интервале $t \in T$ одинакова для всех $d \in D$, то предполагается, что диапазон $Z_j = \bigcap\limits_{d \in D} Z_{jd}$ возможных значений интенсивности z_{jt} операции $j \in J$ не пуст, в дальнейшем $Z_j = [z_{1j}, z_{2j}]$. Аналогично, предполагается, что для любого $w \in W$ диапазон $Z_w = [z_{1w} = \max\{z_{1j} | j \in w\}, z_{2w} = \min\{z_{2j} | j \in w\}]$ возможных значений интенсивности z_{wt} операций блока в интервале t также не пуст. В противном случае все операции из w не могут быть агрегированы в одном блоке и, следовательно, это множество должно быть исключено из W.

В работе при выборе наиболее рационального значения набора $(\mathbf{q}, \boldsymbol{\pi}, \mathbf{x}, \mathbf{z})$ искомых проектных параметров учитываются как добавленная стоимость продукции, выпущенной системой за общий период планирования, так и общие производственные и логистические затраты за этот период.

Указанная добавленная стоимость $Д(\boldsymbol{\pi}, \mathbf{x})$ может быть оценена как суммарная добавленная стоимость каждого произведенного и поставленного (в соответствии с ожидаемым спросом) продукта $d \in D$:

$$
\begin{aligned}
Д(\boldsymbol{\pi}, \mathbf{x}) &= \sum_{d \in D} \sigma_d \min \left\{ \sum_{t=1}^{n} h_d(\pi_t) x_t, \sum_{t=1}^{n} \lambda_{dt} \right\} = \\
&= \sum_{d \in D} \sigma_d \sum_{t=1}^{n} \lambda_{dt} - \sum_{d \in D} \sigma_d \max \left\{ 0, \sum_{t=1}^{n} \lambda_{dt} - \sum_{t=1}^{n} h_d(\pi_t) x_t \right\},
\end{aligned} \tag{1}
$$

где $\sigma_d > 0$ – добавленная стоимость одного продукта d. Здесь $Д(\boldsymbol{\pi}, \mathbf{x}) = \sum\limits_{d \in D} \sigma_d \max\left\{0, \sum\limits_{t=1}^{n} \lambda_{dt} - \sum\limits_{t=1}^{n} h_d(\pi_t) x_t\right\}$ можно рассматривать как недополученную добавленную стоимость (что эквивалентно недополученной прибыли).

Анализ реальных ситуаций показывает, что для многих из них с достаточной для практики точностью можно считать, что общие как материальные, так и временные затраты на выпуск продукции на всем периоде планирования складываются из:

— инвестиционных затрат на систему за вычетом ее остаточной стоимости, амортизационных затрат, включая затраты на оборудование, производственные площади и т.п.;

— текущих эксплуатационных затрат в процессе функционирования системы (далее – эксплуатационные затраты), включающих оплату труда основного и вспомогательного персонала и накладные расходы на нее, стоимости расходуемых ресурсов и затрат на их восстановление, затраты на реконфигурацию системы при переходе системы на выпуск в очередном временном интервале другой группы продуктов;

— логистических затрат, включающих затраты на хранение произведенной, но невостребованной продукции и/или штрафы за неудовлетворенный спрос для каждого из интервалов времени периода планирования.

Инвестиционные и амортизационные затраты зависят от принимаемого в процессе проектирования варианта $\mathbf{q}$ агрегирования операций. Во многих реальных ситуациях с достаточной для практики точностью эта зависимость может быть представлена функцией $C^{\text{ин}}(\mathbf{q}) = C_0 + \sum\limits_{w \in W} C_w q_w$, где C_0 и C_w предполагаются известными для всех $w \in W$.

Рассмотрим зависимости материальных и временных затрат в процессе эксплуатации проектируемой системы от значений набора $(\mathbf{q}, \boldsymbol{\pi}, \mathbf{x}, \mathbf{z})$ искомых проектных параметров для каждого интервала $t \in T$.

Длительность выполнения в интервале t для продукта $d \in D$ операции $j \in J$ при фиксированной ее интенсивности z_{jt} равна $V_{jd} z_{jt}$, где V_{jd} – заданный объем операции $j \in J$ для продукта d. Предполагается, что $V_{jd} = 0$, если операция j для продукта d не выполняется. Учитывая параллельность выполнения операций как на каждой станции, так и операций различных станций, длительность выполнения всех операций такта i равна $\max\{V_{ij}(\pi_t) z_{jt} | j \in J\}$, где $V_{ij}(\pi_t) = V_{jd_{t\chi(\pi_t,i,j)}}$ – объем операции $j \in J$ для обрабатываемого в этом такте продукта $d_{t\chi(\pi_t,i,j)}$ из группы π_t. В дальнейшем условно предполагается, что при $V_{jd} = 0$ диапазон Z_{jd} не ограничен.

Анализ реальных ситуаций показывает, что для многих из них с достаточной для практики точностью можно считать, что общие эксплуатационные как материальные, так и временные затраты складываются из двух основных частей. Первой является сумма затрат, связанных с восстановлением ресурсов, расходуемых при выполнении каждой из операций. Вторую часть представляют дополнительные материальные и временные затраты, связанные

со вспомогательными временами тактов, зарплатой персонала с накладными расходами, обслуживанием оборудования, реконфигурацией системы и т.д.

Рассматривается ситуация, когда максимально возможный объем каждого из расходуемых ресурсов в системе ограничен и заранее известен. Предполагается, что восстановление любого из ресурсов осуществляется после завершения последнего обеспеченного соответствующим ресурсом такта. Выполнение следующего такта может начаться лишь после восстановления этого ресурса. Материальные и временные затраты на восстановление каждого из ресурсов, как правило, известны. Известна обычно также и зависимость скорости расхода соответствующих ресурсов на выполнение единицы объема операции $j \in J$ для продукта $d \in D$ от интенсивности z_{jt} ее выполнения [23, 24]. На базе этой информации для каждой пары (j, d) могут быть построены определенные на диапазоне Z_{jd} функции $f_{pjd}(z_{jt})$, представляющие зависимости удельных (отнесенных к единице объема операции) материальных ($p = 1$) и временных ($p = 2$) затрат на ресурсы и их восстановление от принимаемой интенсивности z_{jt} операции j в интервале t. Как правило, функции $f_{pjd}(z_{jt})$ являются невозрастающими в области их определения для любых j, d, p.

Материальные ($p = 1$) и временные ($p = 2$) затраты на реконфигурацию системы в начале интервала t зависят от групп продуктов, планируемых к выпуску в интервалах t и $t - 1$. Эти затраты $c_p(\pi_{t-1}, \pi_t)$ предполагаются известными.

Остальные общие материальные и временные затраты при эксплуатации системы можно рассматривать относительно каждого цикла. Эти затраты также можно считать состоящими из двух частей, первая из которых пропорциональна числу тактов в цикле, а вторая пропорциональна сумме длительностей всех тактов цикла. Коэффициенты пропорциональности $E_{pt}(\mathbf{q})$, $R_{pt}(\mathbf{q})$ этих зависимостей для группы π_t и интервала t предполагаются заданными для каждого из возможных вариантов агрегирования $\mathbf{q} \in Q$, где $p = 1$ для материальных затрат и $p = 2$ для временных. Аналогично предыдущему во многих реальных ситуациях с достаточной для практики точностью можно считать, что зависимости коэффициентов $E_{pt}(\mathbf{q})$ (как и коэффициентов $R_{pt}(\mathbf{q})$) от варианта агрегирования $\mathbf{q}$ имеют следующую структуру:

$$E_{pt}(\mathbf{q}) = e'_{pt} + \sum_{w \in W} e_{ptw} q_w,$$

где $e'_{pt} > 0$ – величина удельных затрат в случае агрегированного выполнения операций для всех множеств $w \in W$; $e_{piw} > 0$ – дополнительные удельные затраты при автономном выполнении операций множества $w \in W$. Здесь параметры e'_{pt} и e_{piw} предполагаются известными для всех $w \in W$.

Таким образом, в интервале $t \in T$ общие эксплуатационные материальные ($p = 1$) и временные ($p = 2$) затраты на цикл обработки группы π_t продуктов при фиксированных значениях искомых параметров $\mathbf{q}$ и $\mathbf{z}_t$ равны

$$F_{pt}(\mathbf{q}, \pi_t, \mathbf{z}_t) = E_{pt}(\mathbf{q})\bar{h}(\pi_t) + R_{pt}(\mathbf{q}) \sum_{i \in I(\pi_t)} \max\{V_{ij}(\pi_t) z_{jt} | j \in J\} +$$

$$+ \sum_{i \in I(\pi_t)} \sum_{j \in J} V_{ij}(\pi_t) f_{pij}(z_{jt}),$$

где $f_{pij}(z_{jt}) = f_{pjd_{t\chi(\pi_t,i,j)}}(z_{jt})$ – материальные ($p = 1$) и временные ($p = 2$) затраты на единицу объема операции j для обрабатываемого в такте i продукта $d_{t\chi(\pi_t,i,j)}$ при фиксированном значении z_{jt}. В функциях $F_{pt}(\mathbf{q}, \pi_t, \mathbf{z}_t)$ целочисленность числа тактов за время расходования восстановленного ресурса не учитывается, поскольку это число обычно достаточно велико.

При оценке логистических затрат рассматривается ситуация, когда допускается отложенный спрос, т.е. заказы, не выполненные к концу текущего интервала, могут быть удовлетворены в последующие интервалы с выплатой штрафов за просрочку. Логистические затраты рассматриваются по каждому продукту в отдельности и задаются посредством функций $H_{dt}(y)$, представляющих затраты на хранение (при $y > 0$) либо штраф за неудовлетворенный спрос (при $y < 0$) для $|y|$ единиц продуктов $d \in D$ в интервале t. Предполагается, как обычно, что функции $H_{dt}(y)$ не возрастают при $y < 0$, не убывают при $y > 0$ и $H_{dt}(0) = 0$.

Суммарные логистические затраты в системе, выпускающей заданную номенклатуру D продуктов, за весь период планирования при фиксированных значениях искомых проектных параметров $\mathbf{q}$, $\boldsymbol{\pi}$, $\mathbf{x}$, $\mathbf{z}$ равны

$$L(\mathbf{q}, \boldsymbol{\pi}, \mathbf{x}) = \sum_{t=1}^{n} \sum_{d \in D} H_{dt} \left(\sum_{r=1}^{t} h_d(\pi_r) x_r - \Lambda_{dt} \right),$$

где

$$\Lambda_{dt} = \sum_{r=1}^{t} \lambda_{dr}$$

— ожидаемый кумулятивный спрос на продукт $d \in D$ за первые t интервалов.

Общие затраты (инвестиционные, эксплуатационные и логистические) на выпуск и реализацию заданной номенклатуры D продуктов за весь период планирования при фиксированных значениях искомых проектных параметров $\mathbf{q}$, $\boldsymbol{\pi}$, $\mathbf{x}$, $\mathbf{z}$ равны

$$\Phi(\mathbf{q}, \boldsymbol{\pi}, \mathbf{x}, \mathbf{z}) = C_0 + \sum_{w \in W} C_w q_w + \sum_{t=1}^{n} c_1(\pi_{t-1}, \pi_t) +$$
$$+ \sum_{t=1}^{n} x_t F_{1t}(\mathbf{q}, \pi_t, \mathbf{z}_t) + \sum_{t=1}^{n} \sum_{d \in D} H_{dt} \left(\sum_{r=1}^{t} h_d(\pi_r) x_r - \Lambda_{dt} \right).$$

Рассматриваемая проектная задача заключается в нахождении такого допустимого набора $(\mathbf{q}, \boldsymbol{\pi}, \mathbf{x}, \mathbf{z})$ проектных параметров, который максимизирует планируемую прибыль $\mathit{П}(\mathbf{q}, \boldsymbol{\pi}, \mathbf{x}, \mathbf{z}) = \mathit{Д}(\boldsymbol{\pi}, \mathbf{x}) - \Phi(\mathbf{q}, \boldsymbol{\pi}, \mathbf{x}, \mathbf{z})$ от эксплуатации системы за весь период планирования.

С учетом (1) максимизации прибыли $\mathit{П}(\mathbf{q}, \boldsymbol{\pi}, \mathbf{x}, \mathbf{z})$ соответствует минимизация суммы общих затрат $\Phi(\mathbf{q}, \boldsymbol{\pi}, \mathbf{x}, \mathbf{z})$ и недополученной добавленной стоимости $\mathit{Д}(\boldsymbol{\pi}, \mathbf{x})$. Последнюю условно можно рассматривать как сумму дополнительных штрафов за недопоставку каждого из продуктов $d \in D$ в конце

интервала n. В дальнейшем предполагается, что эти дополнительные штрафы включены в модифицированные функции $H_{dn}(\bullet)$. В этом случае решение рассматриваемой проектной задачи сводится к решению следующей задачи смешанного математического программирования:

$$\tilde{\Phi}(\mathbf{q}, \pi_t, \mathbf{x}, \mathbf{z}) = \sum_{t=1}^{n} c_1(\pi_{t-1}, \pi_t) + \sum_{t=1}^{n} F_{1t}(\mathbf{q}, \pi_t, \mathbf{z}_t)x_t + \\ + \sum_{t=1}^{n} \sum_{d \in D} H_{dt} \left(\sum_{r=1}^{t} h_d(\pi_r)x_r - \Lambda_{dt} \right) \to \min \tag{2}$$

при ограничениях

$$F_{2t}(\mathbf{q}, \pi_t, \mathbf{z}_t)x_t \le T_t - c_2(\pi_{t-1}, \pi_t), \quad t \in T, \tag{3}$$

$$\mathbf{q} \in Q, \tag{4}$$

$$\pi_t \in \boldsymbol{D}_t, \quad t \in T, \tag{5}$$

$$x_t \le \bar{x}_t(\pi_t), \quad t \in T, \tag{6}$$

$$z_t \in Z(\mathbf{q}), \quad t \in T, \tag{7}$$

где $\boldsymbol{D}_t$ – множество допустимых вариантов групп π_t во временном интервале t, определяемое возможностями системы; $\bar{x}_t(\pi_t)$ – предельное количество групп π_t продуктов из D, которое может быть произведено системой в интервале t; $Z(\mathbf{q})$ – множество таких векторов $(z_j | j \in J) \in \prod_{j \in J} Z_j$, что для любых $w \in W$ и $j \in w$ выполняется $z_j = z_w \in Z_w$ при $q_w = 0$.

Здесь ограничение (3) учитывает возможность выпуска x_t групп π_t продуктов с учетом длительности T_t интервала t и общих временных затрат $F_{2t}(\bullet)$ на цикл выпуска этой группы. Задачу (2)–(7) в дальнейшем будем называть задачей **A**.

3. Декомпозиционная схема метода решения задачи

Задача **A** представляет собой достаточно сложную задачу смешанного нелинейного программирования, в которой искомыми являются переменные различной природы: $\mathbf{q}$ – $|W|$-мерный двоичный вектор, π_t – размещение с повторениями элементов множества D, $\mathbf{x}$ – n-мерный целочисленный вектор, а $\mathbf{z}_t$ – $|J|$-мерный вещественный вектор, $t \in T$.

Отметим некоторые особенности задачи **A**:

— для любого $t \in T$ структура функции $F_{2t}(\mathbf{q}, \pi_t, \mathbf{z}_t)x_t$ в левой части ограничения (3) совпадает со структурой слагаемого $F_{1t}(\mathbf{q}, \pi_t, \mathbf{z}_t)x_t$ в первой части целевой функции (2);

— если векторы $\mathbf{q}', \mathbf{q}'' \in Q$ таковы, что $q'_w \le q''_w$ для всех $w \in W$, то $Z(\mathbf{q}') \subseteq Z(\mathbf{q}'')$ и для всех $z \in Z(\mathbf{q}')$, $t \in T$ и $p = 1, 2$ выполняется $F_{pt}(\mathbf{q}', \pi_t, z_t) \le F_{pt}(\mathbf{q}'', \pi_t, z_t)$, если при этом $\mathbf{q}' \ne \mathbf{q}''$, то $F_{pt}(\mathbf{q}', \pi_t, \mathbf{z}_t) < F_{pt}(\mathbf{q}'', \pi_t, \mathbf{z}_t)$.

Учитывая указанные особенности задачи **A** предлагается следующий трехуровневый подход к ее решению.

На нижнем уровне для фиксированных значений набора $(\mathbf{q}, \pi_t, x_t)$ искомых переменных и $t \in T$ решаются автономные подзадачи $\mathbf{B1}_t(\mathbf{q}, \pi_t, x_t)$ нелинейного программирования, каждая из которых заключается в определении такого значения $z_t^*(\mathbf{q}, \pi_t, x_t)$ вектора $\mathbf{z}_t \in Z(\mathbf{q})$, которое минимизирует функцию $\tilde{F}_{1t}(\mathbf{q}, \pi_t, \mathbf{z}_t)$ при условии, что $\tilde{F}_{2t}(\mathbf{q}, \pi_t, \mathbf{z}_t) \le T_t^0(\mathbf{q}, \pi_{t-1}, \pi_t, x_t)$, где

$$\tilde{F}_{pt}(\mathbf{q}, \pi_t, \mathbf{z}_t) = F_{pt}(\mathbf{q}, \pi_t, \mathbf{z}_t) - \bar{h}(\pi_t) E_{pt}(\mathbf{q}) =$$
$$= R_{pt}(\mathbf{q}) \sum_{i \in I(\pi_t)} \max\{V_{ij}(\pi_t) z_{jt} | j \in J\} + \sum_{i \in I(\pi_t)} \sum_{j \in J} V_{ij}(\pi_t) f_{pij}(z_{jt}), \quad p = 1, 2,$$

и

$$T_t^0(\mathbf{q}, \pi_{t-1}, \pi_t, x_t) = \frac{T_t - c_2(\pi_{t-1}, \pi_t)}{x_t} - \bar{h}(\pi_t) E_{2t}(\mathbf{q}).$$

Обозначим через $\Theta(\mathbf{q}, \pi_t, x_t)$ оптимальное значение $\tilde{F}_{1t}(\mathbf{q}, \pi_t, \mathbf{z}_t^*(\mathbf{q}, \pi_t, x_t))$ целевой функции $\tilde{F}_{1t}(\mathbf{q}, \pi_t, \mathbf{z}_t)$ в этой подзадаче. Предполагается, что $\Theta(\mathbf{q}, \pi_t, x_t) = \infty$, если подзадача $\mathbf{B1}_t(\mathbf{q}, \pi_t, x_t)$ не имеет решения.

На среднем уровне для фиксированных значений вектора $\mathbf{q}$ решаются подзадачи $\mathbf{B2}(\mathbf{q})$ дискретного программирования по поиску такого значения $(\boldsymbol{\pi}^*(\mathbf{q}), \mathbf{x}^*(\mathbf{q}))$ набора $(\boldsymbol{\pi}, \mathbf{x})$, который минимизирует функцию

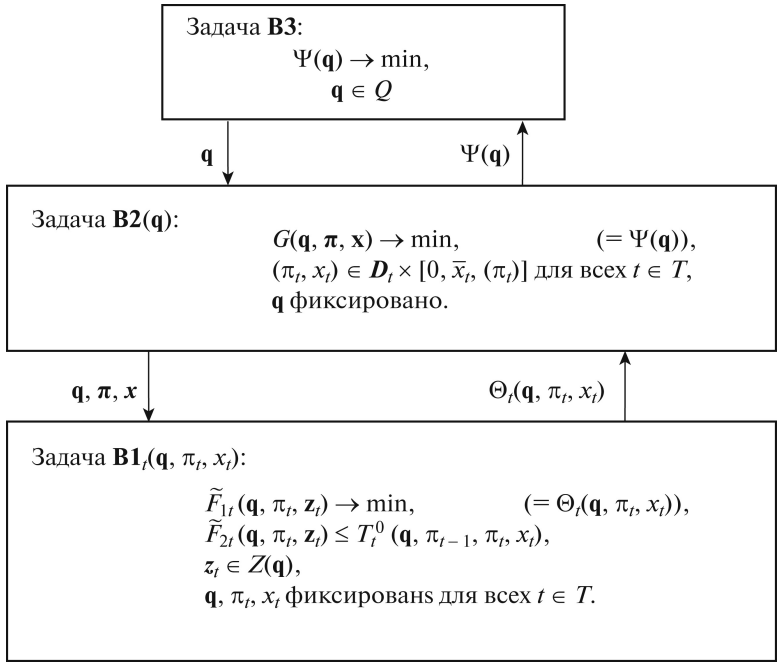
$$G(\mathbf{q}, \boldsymbol{\pi}, \mathbf{x}) = \sum_{t=1}^{n} c_1(\pi_{t-1}, \pi_t) +$$
$$+ \sum_{t=1}^{n} \left[\left(\Theta_t(\mathbf{q}, \pi_t, x_t) + \bar{h}(\pi_t) E_{1t}(\mathbf{q}) \right) x_t + \sum_{d \in D} H_{dt} \left(\sum_{r=1}^{t} h_d(\pi_r) x_r - \Lambda_{dt} \right) \right]$$

при условии, что $(\pi_t, x_t) \in \boldsymbol{D}_t \times [0, \bar{x}_t(\pi_t)]$ для всех $t \in T$. Оптимальное значение $G(\mathbf{q}, \boldsymbol{\pi}^*(\mathbf{q}), \mathbf{x}^*(\mathbf{q}))$ функции $G(\mathbf{q}, \boldsymbol{\pi}, \mathbf{x})$ обозначим через $\Psi(\mathbf{q})$. Если $\Psi(\mathbf{q}) = \infty$ для некоторого $\mathbf{q} \in Q$, то исходная задача $\mathbf{A}$ с этим значением $\mathbf{q}$ не имеет решения.

На верхнем уровне решается подзадача $\mathbf{B3}$ дискретного программирования по определению значения $\mathbf{q}^*$ двоичного вектора $\mathbf{q} \in Q$, минимизирующего функцию $\Psi(\mathbf{q})$. Очевидно, если $\Psi(\mathbf{q}) = \infty$ для всех $\mathbf{q} \in Q$, то исходная задача $\mathbf{A}$ не имеет решения. В противном случае, если $\mathbf{q}^*, \boldsymbol{\pi}^*(\mathbf{q}), \mathbf{x}^*(\mathbf{q})$ и $\mathbf{z}^*(\mathbf{q}, \boldsymbol{\pi}, \mathbf{x})$ являются точными решениями соответствующих подзадач, то набор $(\mathbf{q}^*, \boldsymbol{\pi}^*(\mathbf{q}^*), \mathbf{x}^*(\mathbf{q}^*), \mathbf{z}^*(\mathbf{q}^*, \boldsymbol{\pi}^*(\mathbf{q}^*), \mathbf{x}^*(\mathbf{q}^*)))$ искомых параметров $\mathbf{q}, \boldsymbol{\pi}^*, \mathbf{x}, \mathbf{z}$ является точным решением исходной задачи. Если какие-либо из подзадач $\mathbf{B1}_t(\bullet)$, $\mathbf{B2}(\bullet)$ или $\mathbf{B3}$ решаются приближенно, то набор $(\mathbf{q}^*, \boldsymbol{\pi}^*(\mathbf{q}^*), \mathbf{x}^*(\mathbf{q}^*), \mathbf{z}^*(\mathbf{q}^*, \boldsymbol{\pi}^*(\mathbf{q}^*), \mathbf{x}^*(\mathbf{q}^*)))$ можно принять в качестве приближенного решения этой задачи.

Укрупненная схема предлагаемого подхода приведена на рисунке.

Рассмотрим некоторые подходы к решению выделенных подзадач. Целесообразность того или иного подхода к решению подзадач $\mathbf{B1}_t(\bullet)$ во многом зависит от свойств функций $f_{pjd}(z)$ на областях их определения. Остановимся



Укрупненная схема декомпозиции задачи **A**.

на широко распространенном на практике случае, когда эти функции (следовательно, и функции $f_{pij}(z)$) выпуклы на отрезках Z_j для всех $j \in J$, $d \in D$ и $p = 1, 2$. В этом случае задача $\mathbf{B1}_t(\bullet)$ является задачей выпуклого программирования и для ее решения могут быть использованы известные общие методы решения задач этого класса. Вместе с тем специфика этих задач (вид первых слагаемых функций $\tilde{F}_{1t}(\bullet)$ и $\tilde{F}_{2t}(\bullet)$, сепарабельность их последних слагаемых и невозрастание функций $f_{pij}(z_{jt})$ на отрезках Z_j) позволяют предложить для их решения специальные методы.

Один из них основан на параметрической декомпозиции [24] задачи $\mathbf{B1}_t(\bullet)$ с параметризацией $\max\{V_{ij}(\pi_t)z_{jt} | j \in J\} = \tau_i$, $i \in I(\pi_t)$. Пусть

$$W^0(\mathbf{q}) = \{w \in W \mid q_w = 0\}, \quad J^0(\mathbf{q}) = \{j \in w \mid w \in W^0(\mathbf{q})\}, \quad \boldsymbol{\tau} = (\tau_i \mid i \in I(\pi_t)),$$

$$\tau_{ki}(\mathbf{q}, \pi_t) = \max\left\{\max\left\{V_{ij}(\pi_t)z_{kj} \mid j \in J \backslash J^0(\mathbf{q})\right\}, \max\left\{V_{ij}(\pi_t)z_{kw} \mid w \in W^0(\mathbf{q})\right\}\right\},$$

$$z^*_{jt}(\boldsymbol{\tau}) = \min\left\{z_{2j}, \min\left\{\tau_{2i}(\mathbf{q}, \pi_t)/V_{ij}(\pi_t) \mid i \in I(\pi_t)\right\}\right\} \quad \text{при} \quad j \in J \backslash J^0(\mathbf{q})$$

и

$$z^*_{jt}(\boldsymbol{\tau}) = \min\left\{z_{2w}, \min\left\{\tau_{2i}(\mathbf{q}, \pi_t)/V_{ij}(\pi_t) \mid i \in I(\pi_t)\right\}\right\} \quad \text{при} \quad j \in w \in W^0(\mathbf{q}).$$

Тогда задача $\mathbf{B1}_t(\bullet)$ сводится к следующей задаче выпуклого программирования:

$$(8) \qquad \hat{F}_{1t}(\mathbf{q}, \pi_t, \boldsymbol{\tau}) = R_{1t}(\mathbf{q}) \sum_{i \in I(\pi_t)} \tau_i + \sum_{i \in I(\pi_t)} \sum_{j \in J} V_{ij}(\pi_t) f_{1ij}(z^*_{jt}(\boldsymbol{\tau})) \to \min,$$

$$\hat{F}_{2t}(\mathbf{q},\pi_t,\boldsymbol{\tau}) = R_{2t}(\mathbf{q}) \sum_{i\in I(\pi_t)} \tau_i + \\ + \sum_{i\in I(\pi_t)} \sum_{j\in J} V_{ij}(\pi_t) f_{2ij}(z^*_{jt}(\boldsymbol{\tau})) \le T^0_t(\mathbf{q},\pi_{t-1},\pi_t,x_t), \tag{9}$$

$$\tau_i \in [\tau_{1i}(\mathbf{q},\pi_t), \tau_{2i}(\mathbf{q},\pi_t)], \quad i \in I(\pi_t). \tag{10}$$

В качестве начального решения задачи (8)–(10) можно принять такое значение $\boldsymbol{\tau}^0$ вектора $\boldsymbol{\tau}$, которое минимизирует функцию $\hat{F}_{2t}(\mathbf{q},\pi_t,\boldsymbol{\tau})$. Если $\hat{F}_{2t}(\mathbf{q},\pi_t,\boldsymbol{\tau}^0) > T^0_t(\mathbf{q},\pi_{t-1},\pi_t,x_t)$, то задача $\mathbf{B1}_t(\bullet)$ не имеет решения. Заметим, что $\bar{x}_t(\mathbf{q},\pi_t) = (T_t - c_2(\pi_{t-1},\pi_t))/(\hat{F}_{2t}(\mathbf{q},\pi_t,\boldsymbol{\tau}^0) + \bar{h}(\pi_t)E_{2t}(\mathbf{q}))$ является верхней оценкой допустимого значения параметра x_t в задаче $\mathbf{B2}(\mathbf{q})$.

Для приближенного решения задачи $\mathbf{B1}_t(\bullet)$ может быть адаптирован подход, предложенный в [9] для решения аналогичных задач и основанный на их аппроксимации задачей линейного программирования.

Подзадача $\mathbf{B2}(\mathbf{q})$ может быть сформулирована в терминах многошаговой оптимизации с последующим использованием методов динамического программирования для ее решения. При этом в качестве управляющего на шаге $t \in T$ может быть принят вектор $\mathbf{u}_t = (\pi_t, x_t)$, а в качестве состояния – вектор $\mathbf{s}_t = (\pi_t, \mathbf{a}_t)$, где $\mathbf{a}_t = \left(a_{dt} = \sum_{r=1}^{t} h_d(\pi_t) x_r | d \in D \right)$. При большой размерности пространства состояний для приближенного решения подзадачи $\mathbf{B2}(\mathbf{q})$ можно воспользоваться соответствующими приближенными методами (например, методом «блуждающих трубок») в сочетании с некоторыми эвристиками. Одной из эвристик при построении начального приближения является выбор на очередном шаге $t \in T$ такого управления $\mathbf{u}_t = (\pi_t, x_t)$ из всех возможных, которое для получаемого кумулятивного выпуска и ожидаемого кумулятивного спроса минимизирует сумму затрат на реконфигурацию, производство, хранение запасов продуктов и штрафов за их недопоставку либо (в более упрощенном варианте) лишь два последних слагаемых этой суммы.

В подзадаче **B3** верхнего уровня верхняя граница числа возможных значений вектора $\mathbf{q}$ равна $2^{|W|}$, поэтому ее решение полным перебором всего множества Q требует значительных затрат времени даже при сравнительно небольших значениях $|W|$ и практически нереализуемо при больших $|W|$. Для сокращения перебора могут быть использованы известные методы, основанные на идеях случайного поиска, эвристиках и метаэвристиках в сочетании с предлагаемым ниже вариантом метода последовательной фиксации переменных (МПФП).

Пусть имеется некоторое значение $\mathbf{q}^0$ вектора $\mathbf{q} \in Q$, полученное с использованием перечисленных выше эвристических подходов. Для последующей оптимизации $\mathbf{q}^0$ предлагается следующий алгоритм МПФП. Алгоритм формирует последовательность векторов из Q такую, что каждый следующий вектор отличается от предыдущего только значением ровно одной его компоненты. Компоненты, значения которых были изменены, считаются зафиксированными и в дальнейшем не меняются. Выбор очередного вектора последовательности на текущей итерации выполняется следующим образом. Для

текущего вектора $\mathbf{q}^c$ формируется подмножество $Q(\mathbf{q}^{\text{тек}}) \subset Q$ векторов, которые отличаются от $\mathbf{q}^{\text{тек}}$ одной из его нефиксированных компонент. Для каждого вектора $\mathbf{q} \in Q(\mathbf{q}^{\text{тек}})$ решается подзадача $\mathbf{B2}(\mathbf{q})$. В качестве следующего вектора последовательности выбирается вектор $\mathbf{q}' \in Q(\mathbf{q}^{\text{тек}})$, которому соответствует минимальное значение функции $\Psi(\mathbf{q})$ среди этих подзадач. Компонента, по которой векторы $\mathbf{q}'$ и $\mathbf{q}^{\text{тек}}$ различаются между собой, фиксируется. Число векторов в подмножествах $Q(\mathbf{q}^c)$ от итерации к итерации уменьшается с $|W|$ до 1. Соответственно сокращается и число решаемых подзадач. На каждой итерации в общем случае получается новый лучший вектор $\mathbf{q} \in Q$. Алгоритм завершает работу, когда либо значение $\Psi(\mathbf{q}^{\text{тек}})$ не уменьшилось, либо подмножество $Q(\mathbf{q}^{\text{тек}}) = \varnothing$. Полученный вектор $\mathbf{q}^c$ принимается в качестве решения подзадачи $\mathbf{B3}$. Общее число подзадач $\mathbf{B2}(\mathbf{q})$ вычисления функции $\Psi(\mathbf{q})$ для различных $\mathbf{q}$ не превышает числа $0{,}5(|W|^2 + |W|)$.

Алгоритм МПФП может быть применен повторно для нового начального значения $\mathbf{q}^0$. В качестве такового может быть принято, в частности, полученное ранее решение.

Одно из возможных направлений развития алгоритма МПФП связано с проверкой целесообразности изменения одновременно нескольких компонент вектора $\mathbf{q}^{\text{тек}}$, отбираемых по полученным значениям $\Psi(\mathbf{q})$.

4. Заключение

Предложены математическая модель и метод решения задачи комплексной оптимизации агрегирования операций, программы выпуска и интенсивностей выполнения пересекающихся множеств операций при групповой серийной обработке заданного семейства продуктов на многопозиционной реконфигурируемой производственной системе в заданных временных интервалах при нестационарном детерминированном спросе.

Рассмотрен частный случай задачи, когда выбранный вариант агрегирования операций системы остается неизменным на всем периоде планирования, все операции каждого из заданных подмножеств могут выполняться либо агрегированно с общей для этого множества интенсивностью, либо каждая операция со своей индивидуальной интенсивностью, состав группы продуктов и интенсивности операций в рамках текущего интервала неизменны, но могут быть изменены при переходе к следующему интервалу, зависимости материальных и временных затрат на выполнение любой операции от ее интенсивности представимы выпуклыми функциями, объемы спроса на различные продукты в различных интервалах известны и не связаны между собой.

Описана трехуровневая декомпозиционная схема решения рассматриваемой задачи. Метод приближенного решения задачи верхнего уровня основан на комбинации эвристических методов и методе последовательной фиксации переменных. Метод для задачи среднего уровня выбора состава группы обрабатываемых продуктов и плана выпуска групп на каждом из временных интервалов при фиксированном варианте агрегирования операций в блоки с учетом затрат на сам выпуск, на хранение излишне произведенных продуктов, штрафов за не поставленные в срок заказчику продукты и упущенной

добавленной стоимости основан на идеях метода динамического программирования.

Для подзадачи нижнего уровня поиска оптимальных интенсивностей операций в случае выпуклых функций затрат на операции при фиксированных варианте агрегирования, составе группы продуктов и объеме выпуска предложен метод параметрической декомпозиции для точного ее решения и метод аппроксимации задачей линейного программирования для приближенного решения.

В дальнейшем предполагается исследовать более общие постановки рассмотренной выше задачи, в которых, в частности, варианты агрегирования операций в составе заданных подмножеств операций определяются различными комбинациями подмножеств этих операций, фактический спрос и степень его удовлетворения на отдельные продукты в начальные временные интервалы влияет на последующих интервалах как на возможный спрос на эти продукты, так и на штрафы за их возможную недопоставку. Интерес представляет также исследование возможностей эффективного решения подзадачи нижнего уровня для невыпуклых (в частности, квазивыпуклых, вогнутых, и др.) функций $f_{pjd}(z_{jt})$, представляющих зависимости удельных материальных и временных затрат на ресурсы и их восстановление от принимаемой интенсивности z_{jt} операции j.

СПИСОК ЛИТЕРАТУРЫ

1. *Alting L., Zhang H.* Computer Aided Process Planning: the state-of-the-art survey // Int. J. Prod. Res. 1989. V. 27. No. 4. P. 553–585.
2. *Halevi G.* Process and Operation Panning. Springer, 2003.
3. *Bukchin J., Tzur M.* Design of flexible assembly line to minimize equipment cost // IIE Transact. 2000. V. 32. P. 585–598.
4. *Burkov V.N., Novikov D.A.* Models and methods of multiprojects' management // Syst. Sci. 1999. V. 256. No. 2. P. 5–4.
5. *Dolgui A., Guschinsky N., Levin G.* Enhanced mixed integer programming model for a transfer line design problem / A. Dolgui // Comput. Industr. Engineer. 2012. V. 62. No. 2. P. 570–578.
6. *Левин Г.М., Розин Б.М.* Оптимизация режимов параллельной многоинструментальной обработки деталей на агрегатном оборудовании с учетом групповой смены инструментов // Информатика. 2011. № 3. С. 33–47.
7. *Левин Г.М., Розин Б.М.* Оптимизация последовательно-параллельного выполнения комплекса взаимосвязанных операций // Весці НАН Беларусі. Сер. фіз.-мат. навук. 2013. № 1. С. 111–116.
8. *Dolgui A., Rozin B., Levin G.* Optimisation of processing conditions for multi-product batch production lines with series-parallel operations under uncertainty on demands for finished products // Proc. 18th APIEMS Conf. Industr. Engineer. Management Syst. Bandung, Indonesia, 2017. P. A2-7–A2-12.
9. *Левин Г.М., Розин Б.М., Долгий А.Б.* Линейная аппроксимация задачи оптимизации интенсивностей последовательно-параллельного выполнения пересекающихся множеств операций // Информатика. 2014. № 3. С. 44–51.
10. *Karimi B., Fatemi Ghomi S.M.T., Wilson J.M.* The capacitated lot sizing problem: a review of models and algorithms // Omega. 2003. V. 31. No. 5. P. 365–378.

11. *Ullah H., Parveen S.* A Literature Review on Inventory Lot Sizing Problems // Global J. Res. Engineer. 2010. V. 10. No. 5. P. 21–36.

12. *Chubanov S., Pesch E.* An FPTAS for the single-item capacitated economic lot-sizing problem with supply and demand // Oper. Res. Lett. 2012. V. 40. No. 6. P. 445–449.

13. *Absi N., Kedad-Sidhoum S.* The multi-item capacitated lot-sizing problem with safety stocks and demand shortage costs // Comput. Oper. Res. 2009. V. 36. No. 11. P. 2926–2936.

14. *Florian M., Lenstra J.K., Kan A.H.G.R.* Deterministic production planning: Algorithms and complexity // Management Sci. 1980. V. 26. P. 669–679.

15. *Bitran G.R., Yanasse H.H.* Computational complexity of the capacitated lot size problem // Management Sci. 1982. V. 28. No. 10. P. 1174–1186.

16. *Chen W.H., Thizy J.M.* Analysis of relaxations for the multi-item capacitated lot-sizing problem // Ann. Oper. Res. 1990. V. 26. P. 29–72.

17. *Dixon P.S., Silver E.A.* A heuristic solution procedure for the multi-item, single level, limited capacity, lot sizing problem // J. Oper. Management. 1981. V. 21. No. 1. P. 23–40.

18. *Thizy J.M., Van Wassenhove L.N.* Lagrangean relaxation for the multi-item capacitated lot-sizing problem: a heuristic implementation // IIE Transact. 1985. V. 17. No. 4. P. 308–313.

19. *Gelders L.F., Maes J., Van Wassenhove L.N.* A branch and bound algorithm for the multi item single level capacitated dynamic lotsizing problem / Axaster S., Schneeweiss CH., Silver E., ed. Multistage production planning and inventory control. Lecture notes in economics and mathematical systems. V. 266. Berlin: Springer, 1986. P. 92–108.

20. *Wagelmans A., Van Hoesel S., Kolen A.* Economic lotsizing: an O(n log n) algorithm that runs in linear time in the Wagner–Whitin case // Oper. Res. 1992. V. 40. P. 145–155.

21. *Aggarwal A., Park J.K.* Improved algorithms for economic lot-size problem // Oper. Res. 1993. V. 41. No. 3. P. 549–571.

22. *Federgruen A., Tzur M.A.* A simple forward algorithm to solve general dynamic lot sizing models with n periods in O(n log n) or O(n) time // Management Sci. 1991. V. 37. No. 8. P. 909–925.

23. *Левин Г.М. Розин Б.М., Долгий А.Б.* Оптимизация выпуска и интенсивностей обработки группы деталей при нестационарном спросе // Весці Нац. акад. навук Беларусі. Сер. фіз.-мат. навук. 2016. № 3. С. 102–109.

24. *Левин Г.М., Танаев В.С.* Декомпозиционные методы оптимизации проектных решений. Минск.: Наука и техника, 1978.

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

Поступила в редакцию 17.07.2019

После доработки 12.10.2019

Принята к публикации 28.11.2019

**© 2020 г. М.Я. КОВАЛЁВ, д-р физ.-мат. наук (kovalyov_my@newman.bas-net.by),
Б.М. РОЗИН, канд. техн. наук (rozin@newman.bas-net.by),
Н.Н. ГУЩИНСКИЙ, канд. физ.-мат. наук (gyshin@newman.bas-net.by)
(Объединенный институт проблем информатики НАН Беларуси, Минск)**

МАТЕМАТИЧЕСКАЯ МОДЕЛЬ И АЛГОРИТМ СЛУЧАЙНОГО ПОИСКА ДЛЯ ЗАДАЧИ ОПТИМАЛЬНОГО ПЛАНИРОВАНИЯ ЗАМЕНЫ ТРАДИЦИОННОГО ОБЩЕСТВЕННОГО ТРАНСПОРТА ЭЛЕКТРИЧЕСКИМ[1]

Исследуется сложная оптимизационная задача, возникающая при планировании процесса перехода от традиционного общественного транспорта к электрическому. Описываются предположения, входные и выходные параметры задачи, а также ее математическая модель и рандомизированный алгоритм решения. Приводится библиография публикаций по исследуемой задаче.

Ключевые слова: электрический транспорт, оптимизация, математическое программирование, рандомизированный алгоритм.

DOI: 10.31857/S0005231020050049

1. Введение

В статье исследуется оптимизационная задача, возникающая при планировании процесса замены парка традиционного общественного транспорта электробусами для заданного множества маршрутов. Предполагается, что электробус оборудован устройством хранения электрической энергии – батареей, которая требует периодической зарядки. В статье рассматривается только технология зарядки, при которой батареи электробусов заряжаются на стационарных станциях зарядки. Замена батарей и зарядка во время движения не предусматриваются.

Парк электробусов характеризуется типами электробусов и количеством электробусов каждого типа. С каждым типом электробуса связаны следующие характеристики такого же типа: множество подходящих типов станции зарядки; время зарядки до рекомендуемого уровня состояния заряда (State Of Charge (SOC)) при отправлении от станции зарядки одного и того же типа, расположенной в одном и том же месте (депо или остановка); индикаторы допустимого движения между любыми двумя остановками одного и того же маршрута; потребление энергии для одного и того же маршрута в течение года; годовые капитальные и эксплуатационные расходы; пассажировместимость.

[1] Работа выполнена в рамках проекта PLATON инициативы ERA-NET Cofund Electric Mobility Europe.

Маршрут характеризуется указанием депо и цикла маршрута, представляющего последовательность остановок, циклически посещаемых электробусами этого маршрута. Интенсивностью пассажиропотока цикла маршрута является историческая или запланированная общая вместимость пассажирских транспортных средств, отправляющихся с любой остановки этого цикла в единицу времени. Блоком циклов маршрута будем называть последовательность следующих друг за другом циклов этого маршрута, моделируемых одной и той же интенсивностью пассажиропотока и одними и теми же индикаторами допустимого движения. Для каждого маршрута будем рассматривать обеспечивающий глобальную допустимость (ОГД) блок циклов, такой что допустимое решение, принятое для этого блока циклов, является допустимым для любого другого блока циклов в течение года. С некоторой степенью неопределенности ОГД блок циклов может быть охарактеризован максимальными потерями уровня SOC при движении по тем же сегментам маршрута в сравнении с другими блоками циклов.

Для заданного маршрута частота отправления электробусов некоторого типа равна количеству электробусов этого типа, отправляющихся с любой одной и той же остановки этого маршрута в единицу времени. Задача оптимизации заключается в определении

- парка электробусов,
- частоты отправления электробусов каждого типа в ОГД блоке циклов для каждого маршрута,
- мест установки станций зарядки и трансформаторов,
- количества станций зарядки каждого типа в каждом выбранном для них месте,
- назначения мест установки станций зарядки трансформаторам и
- назначения станций зарядки маршрутам

таким образом, чтобы электробусам хватало заряда для движения по маршрутам и не была превышена выделенная (на зарядку электробусов) мощность любого трансформатора. В качестве критерия рассматривается максимизация отношения общей эффективности (положительного экологического или социально-экологического эффекта, выраженного количественно) к сумме общих капиталовложений и эксплуатационных расходов (включая стоимость потребляемой энергии), т.е. максимизация удельной эффективности инвестиций. В дальнейшем обозначим эту задачу через **P**. Предполагается, что решение задачи **P** будет повторяться для нескольких последовательных периодов планирования. Элементы решения, принятые в предыдущий период, включаются во входные данные для будущего периода. Такой подход последовательной оптимизации основан на предположении, что оптимизация для ближайшего периода времени более эффективна, чем оптимизация для последующих периодов, когда не все входные данные достаточно точно определены.

Задача **P** сложна как с точки зрения вычислений, так и с точки зрения построения адекватной математической модели. Чтобы решить ее с приемлемым качеством за приемлемое время, в статье сделан ряд предположений, которые приведены в разделе 3. Входные и выходные данные описаны в разделах 4 и 5 соответственно. Математическая модель представлена в разделе 6. В разделе 7 предложен рандомизированный эвристический алгоритм

для решения поставленной задачи. В разделе 2 приводится список публикаций, известных авторам настоящей статьи по рассматриваемой проблеме, классифицированных по тематике.

2. Классификация публикаций по тематике

Подходящие публикации классифицируется по категориям, связанным с планированием внедрения и эксплуатацией электрических транспортных средств (EVs). Названия категорий приведены далее и сопровождаются списком соответствующих публикаций. Если публикация может быть отнесена к нескольким категориям, то она расположена в той, которая, по мнению авторов, является более подходящей.

- История и статистика использования EVs и соответствующей инфраструктуры: ZeEUS eBus Report [1], Nikolič и Živanovič [2], Li [3], Ahmad и др. [4], Anderson и др. [5], Todorovic и Simic [6].
- Анализ тестирования и эксплуатации EVs в реальных условиях: Barnitt [7], Wang и González [8], Erkkilä и др. [9], Schmidt и др. [10], ZeEUS Demonstrations [11], Foltiński [12], Rogge и др. [13], Hanlin [14], Olsson и др. [15], Eudy и Jeffers [16], Gao и др. [17], Leou и Hung [18], Christensen и др. [19], Khan и др. [20], Xylia и Silveira [21], Gallet и др. [22], Morganti и Browne [23].
- Сравнение EVs и транспортных средств с другими источниками энергии: Feng и Figliozzi [24, 25], Hallmark и др. [26], Lajunen [27], Mohamed и др. [28].
- Имитационное моделирование эксплуатации EVs: Schoch [29], Teoh и др. [30, 31], Mohamed и др. [32], Marmaras и др. [33], Xylia и др. [34], Fiori и др. [35].
- Оптимизация эксплуатации EVs и требуемой инфраструктуры: Alonso и др. [36], Wen и др. [37], Yu и др. [38], Juan и др. [39], Hiermann и др. [40], Quak и др. [41], Desaulniers и др. [42], Wielinski и др. [43], Kunith и др. [44], Bruglieri и др. [45], Pelletier и др. [46–49], Froger и др. [50, 51], Xylia и др. [52], Liu и др. [53], Hosseini и Sarder [54], Wang и др. [55], Wang и др. [56].
- Экологические проблемы: доклад Всемирной организации здравоохранения [57], Sydbom и др. [58], доклады института Health Effects Institute [59, 60], Ali [61], Chan и др. [62], монография IARC [63], Mohner [64], McClennan [65], Gaskins и др. [66], статьи специального выпуска журнала Transportation Research Part D под редакцией Johem и др. [67].

Анализ приведенных публикаций показывает, что количество математических моделей и алгоритмов решения задач оптимизации эксплуатации электрического общественного транспорта и необходимой инфраструктуры недостаточно, чтобы охватить огромное число разнообразных реальных практических ситуаций.

3. Предположения и допущения

В задаче **P** предполагается следующее:

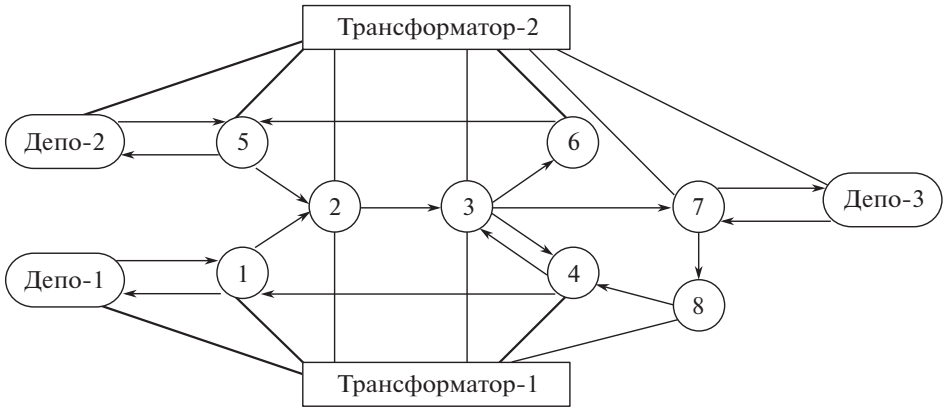
1. Для каждого маршрута известны депо, промежуточные и конечные остановки, а также порядок их посещения соответствующими электробусами;

2. Парк традиционных автобусов может замещаться частично;
3. Каждый маршрут обслуживает единственное депо. Если на некоторый маршрут назначен хотя бы один электробус, то по крайней мере одна подходящая станция зарядки должна быть открыта в депо этого маршрута;
4. Маршруты могут пересекаться в депо, на конечных и промежуточных остановках;
5. Любой электробус некоторого типа, назначенный на некоторый маршрут, заряжается до рекомендуемого уровня SOC каждый раз, когда он посещает место со станцией зарядки того типа, который назначен этому типу электробусов и этому маршруту. Если указанное назначение не сделано для подходящей четверки (тип электробуса, маршрут, место, тип станции зарядки), то при моделировании электробус посещает указанное место без зарядки;
6. В месте зарядки один и тот же тип станции зарядки может быть назначен разным типам электробусов. В этом случае электробусы этих типов совместно используют станции зарядки этого типа в этом месте;
7. Электробусы одного и того же типа и маршрута в одном и том же месте заряжаются на станциях одного типа;
8. Каждая станция зарядки в одном и том же месте соединяется с одними и теми же m трансформаторами. В каждый момент времени только одно произвольное соединение места расположения станции зарядки с трансформаторами является активным;
9. Некоторые электробусы, станции зарядки и их связи с трансформаторами к моменту принятия решения могут уже использоваться. В дальнейшем будем их называть “старые”, а электробусы и элементы инфраструктуры, относительно которых принимается решение, будем называть “новые”;
10. Время, затрачиваемое любым электробусом на прохождение каждого из ОГД блоков циклов, не меняется для одного и того же маршрута, и оно определяется как некоторая оценка или достаточно точная верхняя граница этого времени.

4. Входные данные

Строчные буквы используются для обозначения числовых входных данных (параметров), а прописные – для обозначения множеств и переменных.

- Транспортная и электрическая сеть $G = (NN, R, EE)$, которая представляет собой взвешенный смешанный мультиграф с множеством вершин (мест для станций зарядки и трансформаторов) NN, множеством ориентированных циклов (маршрутов) R и множеством ребер (связей с трансформаторами) EE, см. рисунок в качестве иллюстрации. На этом рисунке маршрут 1 определяется циклом (Депо-1, 1, 2, 3, 4, 1, Депо-1), маршрут 2 – (Депо-2, 5, 2, 3, 6, 5, Депо-2) и маршрут 3 – (Депо-3, 7, 8, 4, 3, 7, Депо-3).
- Множество NN разбито на множество T “трансформаторных” вершин, соответствующих приемлемым местам расположения трансформаторов, и множество N “нетрансформаторных” вершин.



Сеть для трех маршрутов.

- Множество N разбито на подмножества ND и NE вершин депо и вершин остановок соответственно, в которых могут быть установлены станции зарядки.
- Множество N содержит подмножество NO нетрансформаторных вершин, в каждой из которых открыта хотя бы одна старая станция зарядки любого типа.
- Множество T содержит подмножество TO трансформаторных вершин, в каждой из которых находится по крайней мере один старый трансформатор с ненулевой выделенной мощностью для новых электробусов.
- Множество маршрутов R построено на основе множеств ND и NE. Одна и та же вершина может принадлежать разным маршрутам.
- Подмножество $R_0 \subseteq R$ маршрутов, обслуживаемых хотя бы одним старым электробусом.
- Дуга $(i, j) \in r$, $r \in R$, представляет собой ориентированный участок маршрута, идущий из нетрансформаторной вершины i в нетрансформаторную вершину j.
- Ребро $(i, j) \in EE$ представляет собой приемлемое соединение трансформаторной вершины i и нетрансформаторной вершины j.
- Множество B типов электробусов.
- Подмножество BO, $BO \subseteq B$, старых типов электробусов.
- Множество C типов станций зарядки.

Тип $c \in C$ характеризуется следующими параметрами:

- Номинальная мощность po_c одной станции зарядки;
- Капитальные затраты cc_c^{cap}, включающие стоимость приобретения и установки одной станции зарядки без учета затрат, относящихся к подключению к трансформаторам;
- Расходы cc_c^{ope} на эксплуатацию одной станции зарядки в течение года;
- Множество $N_c \subseteq N$ вершин, приемлемых для открытия станции зарядки типа c;
- Множество $NO_c \subseteq NO$ вершин, в каждой из которых открыта хотя бы одна старая станция зарядки типа c;

- Множество $B_c \subseteq B$ типов электробусов, подходящих для станции зарядки типа c.

  Каждый тип электробуса $b \in B$ характеризуется следующими параметрами:
- Множество C_b подходящих типов станций зарядки. Электробус типа b может заряжаться только на станциях типа $c \in C_b$;
- Множество NM_b нетрансформаторных вершин, в каждой из которых должна быть открыта хотя бы одна станция зарядки типа $c \in C_b$, если вершина принадлежит маршруту, обслуживаемому электробусами типа b;
- Множество R_b допустимых маршрутов, $R_b \subseteq R$;
- Пассажировместимость cap_b одного электробуса;
- Капитальные затраты cv_b^{cap} на один электробус;
- Расходы cv_b^{ope} на эксплуатацию одного электробуса в течение года, не включая стоимость потребляемой энергии;

  Каждая нетрансформаторная вершина $j \in N$ характеризуется следующими параметрами:
- Множество C_{jb} подходящих типов станций зарядки электробусов типа b. Электробус типа b в вершине j может заряжаться только на станции типа $c \in C_{jb}$;
- Число m соединений любой вершины $j \in N \backslash NO$, в которой будет открыта хотя бы одна новая станция зарядки, с трансформаторными вершинами. Предполагается, что $m \leq |T|$;
- Множество маршрутов R_j, пересекающихся в j;
- Множество $TE_j \subseteq T$ трансформаторных вершин, допустимых для соединения с вершиной j;
- Количество nc_{jc} старых станций зарядки типа c, которые открыты в $j \in N_c$, $c \in C$. Напомним, что если $j \in NM_b$ и j принадлежит маршруту, обслуживаемому старым электробусом типа b, то $nc_{jc} \geq 1$ для $c \in C_b$;
- Множество RO_{jcb} маршрутов r таких, что их старые электробусы типа b заряжаются на станциях типа c в вершине j, $RO_{jcb} \subseteq R_j$, $j \in N_c$, $b \in R_b \cap B_c$, $c \in C$;
- Верхняя граница uc_{jc} на количество станций зарядки типа c, которые могут быть открыты в $j \in N_c$, $c \in C$. Этот параметр может быть опущен или равен бесконечности, если отсутствует необходимость в его использовании;
- Верхняя оценка ct_{jbc} времени зарядки одного электробуса типа b на станции типа $c \in C_b$, открытой в $j \in N_c$, до рекомендованного уровня SOC, либо оценка этого времени, которое для промежуточной остановки определяется средним временем посадки/высадки пассажиров, для конечной остановки – максимальным временем замены или отдыха водителя, для депо – продолжительностью пребывания в депо;
- Длительность t_j^{depot} интервала времени I_j, в котором количество всех электробусов, находящихся в депо $j \in ND$, в единицу времени является наибольшим.

  Каждая трансформаторная вершина $i \in T \backslash TO$ характеризуется:
- запасами мощности o_i для питания новых станций зарядки в ОГД блоке циклов;

- капитальной стоимостью трансформатора cb_i ($cb_i = 0$ для $i \in TO$).

Каждое ребро $(i, j) \in EE$ связано с:

- затратами cl_{ij} на соединение трансформаторной вершины i и нетрансформаторной вершины j.

Маршрут $r \in R$ характеризуется следующими параметрами:

- Множество B_r типов электробусов, допустимых для обслуживания маршрута r;
- Последовательность $\pi_r = (j_0, j_1, \ldots, j_r, j_0)$ вершин, где j_0 – вершина депо и $j_1, \ldots, j_r$ – вершины остановок, которые посещаются циклически в указанном порядке в течение ОГД блока циклов. Будем использовать записи $j \in r$ и $(i, j) \in r$ для обозначения того, что вершина j и дуга (i, j) принадлежат маршруту r соответственно;
- Индикатор допустимого движения $ei_{r(i,j)b}$: $ei_{r(i,j)b} = 1$, если электробус типа b может доехать из вершины i в вершину j маршрута r в ОГД блоке циклов при условии, что станция зарядки типа $c \in C_b$ установлена в i, иначе $ei_{r(i,j)b} = 0$, $i \in r$, $i \in N_c$, $j \in r$, $b \in B_r$. Для заданного типа электробуса этот индикатор вычисляется на основе рекомендуемого уровня SOC, минимального уровня SOC и условий движения по сегменту (i, j);
- Частота fr_{rb} движения старых электробусов типа b на маршруте r в ОГД блоке циклов, $fr_{rb} = 0$ тогда и только тогда, когда ни один старый электробус типа b не назначен на маршрут r, $b \in B_r$;
- Нижняя граница lfr_{rb}, $lfr_{rb} > 0$, частоты движения новых электробусов типа b на маршруте r в ОГД блоке циклов, если маршрут r выбран для конверсии, $b \in B_r$;
- Верхняя граница ufr_{rb}, $lfr_{rb} \leq ufr_{rb}$, частоты движения новых электробусов типа b на маршруте r в ОГД блоке циклов, если маршрут r выбран для конверсии, $b \in B_r$. Этот параметр может быть опущен или равен бесконечности, если отсутствует необходимость в его использовании;
- Наибольшая доля α^{depot}_{rbj}, $0 < \alpha^{depot}_{rbj} \leq 1$, всех электробусов типа b, $b \in B_r$, прибывающих в депо $j \in ND$ за интервал времени длительности t^{depot}_j. Если t^{depot}_j и α^{depot}_{rbj} сложно определить, то можно положить длительность t^{depot}_j равной длительности интервала времени между прибытием последнего электробуса в депо j вечером предыдущего дня и отправлением первого электробуса из депо j утром следующего дня, а также положить $\alpha^{depot}_{rbj} = 1$;
- Количество nv_{rb} старых электробусов типа b, назначенных на маршрут r в ОГД блоке циклов, $b \in B_r$;
- Верхняя граница uv_{rb} количества новых электробусов типа b, назначенных на маршрут r в ОГД блоке циклов, $b \in B_r$. Этот параметр может быть опущен или равен бесконечности, если отсутствует необходимость в его использовании;
- Верхняя оценка d_r продолжительности одного цикла любого электробуса в ОГД блоке циклов;
- Верхняя оценка dc_{rb} суммарного времени зарядки одного электробуса типа b в одном цикле ОГД блока циклов, $b \in B_r$;

- Стоимость энергии (или ее оценка) ce_{rb}, потребляемой одним электробусом типа b на маршруте r в течение года, $b \in B_r$;
- Коэффициент предпочтения (вес) w_r, $w_r \geq 0$;
- Нижняя граница $lc > 0$ суммарной капитальной и операционной стоимости.
- Верхняя граница uc, $uc > lc$, суммарной капитальной и операционной стоимости.
- Интенсивность пассажиропотока pas_r в ОГД блоке циклов, которая поддерживается традиционными автобусами, и, следовательно, может поддерживаться новыми электробусами;
- Функция $co_r(Z)$, аппроксимирующая суммарные вредные выбросы традиционных автобусов, которые перевозят Z пассажиров на маршруте r в единицу времени ОГД блока циклов;
- Функция $fu_r(Z)$, аппроксимирующая суммарное потребление топлива традиционных автобусов, которые перевозят Z пассажиров на маршруте r в единицу времени ОГД блока циклов.

Замечание. Пусть Z_r обозначает максимальное количество пассажиров, отправляющихся в новых электробусах от любой конечной или промежуточной остановки маршрута r в единицу времени ОГД блока циклов. Очевидно, что $Z_r \leq pas_r$. Предполагается, что значение эффективности (частичной) конверсии для маршрута r выражается функцией $v_r(Z_r)$ и предлагаются три подхода для вычисления $v_r(Z_r)$: 1) $v_r(Z_r) = w_r Z_r$, 2) $v_r(Z_r) = w_r co_r(Z_r)$ и 3) $v_r(Z_r) = w_r fu_r(Z_r)$.

5. Выходные данные

Решение X задачи **P** может быть представлено следующими переменными:

- Частота $FR_{rb}(X)$ новых электробусов типа b на маршруте r в ОГД блоке циклов, где $FR_{rb} = 0$ тогда и только тогда, когда ни один новый электробус типа b не обслуживает маршрут r, $r \in R_b$, $b \in B$;
- Множество $R_{jcb}(X)$ маршрутов r таких, что $fr_{rb}+FR_{rb}(X) > 0$ и их старые и новые электробусы типа b заряжаются на станциях типа c в вершине j, $RO_{jcb} \subseteq R_{jcb}(X) \subseteq R_j$, $j \in N_c$, $b \in R_b \cap B_c$, $c \in C$;
- Множество $S_c(X)$ вершин $j \in N_c$, в каждой из которых открыта по крайней мере одна новая станция зарядки типа c;
- Количество $NC_{jc}(X)$ новых станций зарядки типа $c \in C$, которые должны быть установлены в нетрансформаторной вершине $j \in S_c(X)$, $c \in C$;
- Множество $L_j(X)$ трансформаторных вершин $i \in TE_j$, которые должны быть соединены с нетрансформаторной вершиной j, $j \in N \backslash NO$.

Решение X может быть использовано для вычисления следующих значений:

- $B_r(X) = \{b \in B_r \mid FR_{rb}(X) > 0\}$ – множество типов электробусов таких, что хотя бы один новый электробус такого типа назначен на маршрут $r \in R$;
- $B(X) = \cup_{r \in R} B_r(X)$;

- $R_b(X) = \{r \in R_b \mid FR_{rb}(X) > 0\}$ – множество маршрутов, каждый из которых обслуживается хотя бы одним новым электробусом типа b, $b \in B$;
- $R(X) = \cup_{b \in B} R_b(X)$;
- $C(X)$ – множество типов старых и новых станций зарядки;
- $S(X) = \cup_{c \in C(X)} S_c(X)$ – множество вершин, в каждой из которых установлена хотя бы одна новая станция зарядки любого типа;
- $R_j(X)$ – множество маршрутов, пересекающихся в вершине j и обслуживаемых по крайней мере одним старым или новым электробусом, $j \in S(X)$;
- $SR_{rb}(X)$ – множество дуг маршрута $r \in R_b(X)$, в каждой вершине j которых установлена хотя бы одна старая или новая станция зарядки типа $c \in C_{jb}$ для обслуживания электробусов типа $b \in B(X)$, назначенных на этот маршрут;
- $c_{jrb}(X)$ – единственный тип новой или старой станции зарядки в вершине j новых и старых электробусов типа b, назначенных на маршрут r;
- $B_{jc}(X) = \{b \mid c_{jrb}(X) \neq False, b \in B, r \in R_b\}$ – множество типов электробусов всех маршрутов, пересекающихся в j, которые будут заряжаться на новых и старых станциях типа c, $j \in S(X)$, $c \in C(X)$;
- $Z_r(X) = \sum_{b \in B_r(X)} cap_b FR_{rb}(X)$ – интенсивность пассажиропотока маршрута r, поддерживаемая новыми электробусами (максимальное количество пассажиров, отправляющихся в новых электробусах с любой остановки маршрута r в единицу времени в ОГД блоке циклов), $r \in R(X)$;
- $NV_{rb}(X) = \lceil d_r FR_{rb}(X) \rceil$ – количество новых электробусов типа b, назначенных на маршрут r в ОГД блоке циклов;
- $FN_{jb}(X) = \sum_{r \in R_j(X)} (fr_{rb} + FR_{rb}(X))$ – суммарная частота прибытия новых и старых электробусов типа b в вершину j в ОГД блоке циклов, $j \in$ $\in NE \cap (S(X) \cup NO)$, $b \in B(X)$;
- $FN_{jb}(X) = \sum_{r \in R_j(X)} \alpha_{rbj}^{depot} (nv_{rb} + NV_{rb}(X))/t_j^{depot}$ – суммарная частота прибытия новых и старых электробусов типа $b \in B(X)$ в депо j, $j \in$ $\in ND \cap S(X)$, $b \in B(X)$ за интервал времени I_j длительности t_j^{depot};
- $M_i(X) = \{j \in S(X) \mid i \in L_j(X)\}$ – множество новых нетрансформаторных вершин, связанных с трансформаторной вершиной i;
- $T(X)$ – множество трансформаторных вершин, каждая из которых связана с по крайней мере одной новой станцией зарядки;
- $TP_i(X) = \sum_{j \in M_i(X) \cap N} \sum_{c \in C(X)} po_c NC_{jc}(X)$ – суммарная мощность новых станций зарядки, соединенных с трансформаторной вершиной $i \in T(X)$, в ОГД блоке циклов;
- $TP(X) = \sum_{i \in T(X)} TP_i(X)$ – суммарная мощность всех новых станций зарядки в ОГД блоке циклов;
- $V(X) = \sum_{r \in R(X)} v_r(Z_r(X))$ – общая эффективность;
- $CC(X) = \sum_{c \in C(X)} \sum_{j \in S_c(X)} cc_c^{cap} NC_{jc}(X) + \sum_{r \in R(X)} \sum_{b \in B_r(X)} cv_b^{cap} NV_{rb}(X) +$ $+ \sum_{j \in S(X) \setminus NO} \sum_{i \in L_j(X)} cl_{ij} + \sum_{i \in T(X) \setminus TO} cb_i$ – капитальные затраты;
- $OC(X) = \sum_{c \in C(X)} \sum_{j \in S_c(X)} cc_c^{ope} NC_{jc}(X) + \sum_{r \in R(X)} \sum_{b \in B_r(X)} (cv_b^{ope} +$ $+ ce_{rb}) NV_{rb}(X)$ – операционные затраты, включая стоимость электроэнергии.

6. Постановка задачи

Обозначим через $\mathcal{X}$ область допустимых решений X задачи **P**. Она определяется системой соотношений:

$$lc \le CC(X) + OC(X) \le uc, \tag{1}$$

$$Z_r(X) = \sum_{b \in B_r(X)} cap_b FR_{rb}(X) \le pas_r, \quad r \in R(X), \tag{2}$$

$$TP_i(X) = \sum_{j \in M_i(X)} \sum_{c \in C(X)} po_c NC_{jc}(X) \le o_i, \quad i \in T(X), \tag{3}$$

$$lfr_{rb} \le FR_{rb}(X) \le \min \left\{ ufr_{rb}, \frac{pas_r - \sum_{b' \in B(X), b' \ne b} cap_{b'} lfr_{rb'}}{cap_b} \right\}, \quad r \in R_b(X), \quad b \in B(X), \tag{4}$$

$$c_{jrb}(X) \in C_{jb}, \quad j \in SR_{rb}(X), \quad r \in R_b(X), \quad b \in B(X), \tag{5}$$

$$ei_{r(i,j)b} = 1, \quad (i,j) \in SR_{rb}(X), \quad r \in R_b(X), \quad b \in B(X), \tag{6}$$

$$\sum_{j \in SR_{rb}(X) \setminus ND} ct_{jbc^*} \le dc_{rb}, \quad c^* = c_{jrb}(X), \quad r \in R_b(X), \quad b \in B(X), \tag{7}$$

$$nv_{rb} + NV_{rb}(X) = nv_{rb} + \lceil d_r FR_{rb}(X) \rceil \le uv_{rb}, \quad r \in R_b(X), \quad b \in B(X), \tag{8}$$

$$nc_{jc} + NC_{jc}(X) \le uc_{jc}, \quad j \in S_c(X), \quad c \in C(X), \tag{9}$$

$$\sum_{c \in C_b} (nc_{jc} + NC_{jc}(X)) \ge 1, \quad j \in NM_b \cap S(X), \quad b \in B(X) \cup BO, \tag{10}$$

$$nc_{jc} + NC_{jc}(X) \ge \sum_{b \in B_{jc}(X)} ct_{jbc} FN_{jb}(X), \quad j \in S(X) \cup NO, \quad c \in C, \tag{11}$$

$$|L_j(X)| = m, \quad j \in S(X) \setminus NO. \tag{12}$$

Соотношения (1) ограничивают суммарные капитальные и операционные затраты снизу и сверху. Соотношения (2) ограничивают сверху пассажиропоток для каждого маршрута, обслуживаемого новыми электробусами. Ограничения (3) гарантируют, что общая мощность новых станций зарядки, соединенных с одним и тем же трансформатором, не превышает выделенной мощности этого трансформатора. Ограничения (4) определяют нижние и верхние границы частоты движения новых электробусов. Ограничения (5) гарантируют, что подходящая станция зарядки установлена в каждой вершине дуг множества $SR_{rb}(X)$. При выполнении ограничений (6) любой новый электробус может допустимо двигаться по своему маршруту, если подходящие станции зарядки установлены в вершинах дуг множества $SR_{rb}(X)$. Ограничения (7) гарантируют, что общее время зарядки любого нового электробуса некоторого типа на некотором маршруте в одном цикле не превосходит верхнюю границу для этого типа и маршрута. Ограничения (8) обеспечивают то, что

общее количество старых и новых электробусов каждого типа, обслуживающих один и тот же маршрут, не превосходит верхней границы, установленной для этого типа электробуса и маршрута. Соотношения (9) ограничивают сверху общее количество новых и старых станций зарядки определенного типа для каждой вершины. Ограничения (10) указывают, что по крайней мере одна новая или старая станция зарядки типа $c \in C_b$ должна быть установлена в вершине множества NM_b, если эта вершина принадлежит маршруту, обслуживаемому по крайней мере одним новым электробусом. Депо является одной из таких вершин. Ограничения (11) гарантируют, что количества новых и старых станций зарядки типа c, установленных в вершине j для обслуживания электробусов типа $B_{jc}(X)$ (всех маршрутов), достаточно для зарядки равномерно прибывающих электробусов без простоя в предположении, что электробусы прибывают равномерно. Отметим, что равномерное прибытие электробусов является упрощением реальной ситуации. При выполнении ограничений (12) количество новых связей нетрансформаторной вершины, в которой установлена хотя бы одна новая станция зарядки и нет старых станций зарядки, с трансформаторными вершинами равно m.

Задача **P** формулируется так:

$$\max_{X \in \mathcal{X}} \frac{V(X)}{CC(X) + OC(X)}.$$

Следует отметить, что оптимальное решение задачи **P** является Парето-оптимальным решением трехкритериальной задачи максимизации $V(X)$ и минимизации $CC(X)$ и $OC(X)$ для $X \in \mathcal{X}$, см. терминологию и результаты по решению многокритериальных задач в Steuer [68], Vincke [69], Roy [70], Collette и Siarry [71] и Ehrgott [72]. Поскольку задача **P** является сложной с вычислительной точки зрения, в разделе 7 предлагается рандомизированный эвристический алгоритм ее решения.

7. Рандомизированный эвристический алгоритм

Из-за вычислительной сложности задачи **P** предлагается следующий подход к ее решению. Путем случайного выбора частей допустимых или недопустимых решений строится множество допустимых решений $\mathcal{Q} \in \mathcal{X}$, которое, как ожидается, будет содержать решения, близкие к оптимальному. Затем из построенных решений выбирается наилучшее.

Формальное описание алгоритма, называемого далее алгоритмом **A**, приводится далее. Предполагается, что шаги алгоритма **A** выполняются последовательно, если не указано другое. Алгоритм **A** использует вероятности для выбора определенного значения численной характеристики решения. Эти вероятности являются управляющими параметрами алгоритма. Они могут быть определены лицом, принимающим решение, или установлены одинаковыми для всех возможных значений одной и той же характеристики решения. В последнем случае предполагается равномерное распределение вероятностей или распределение, настраиваемое по результатам численных экспериментов.

Алгоритм **А**.

Шаг 1 (инициализация). Положить $\mathcal{Q} = \varnothing$. На шагах 2–6 формируется частичное решение Q. Оно может быть достроено до допустимого или недопустимого полного решения.

Шаг 2 (формирование множества маршрутов $R(Q)$, обслуживаемых по крайней мере одним электробусом). Определить вероятности p_r, $0 \le p_r \le 1$, включения $r \in R$ в $R(Q)$. Положить $p_r = 1$ для старых маршрутов $r \in R_0$. Сформировать множество $R(Q)$, $|R(Q)| \ge 1$, с помощью этих вероятностей. Определить множество вершин $N(Q) = \{j \mid j \in R(Q)\}$.

Шаг 3 (формирование множества $B_r(Q)$ типов электробусов для обслуживания маршрута $r \in R(Q)$). Для каждого маршрута $r \in R(Q)$ определить вероятность p_{rb} использования электробусов типа b на маршруте r, $b \in B_r$. Положить $p_{rb} = 1$, если электробусы типа b уже используются на маршруте r. Сформировать множества $B_r(Q)$, $r \in R(Q)$, используя эти вероятности таким образом, чтобы $|B_r(Q)| \ge 1$ для всех $r \in R(Q)$. Сформировать множество типов электробусов $B(Q) = \cup_{r \in R(Q)} B_r(Q)$ и множества $R_b(Q)$ маршрутов, обслуживаемых по крайней мере одним электробусом типа $b \in B(Q)$.

Шаг 4 (формирование мест для станций зарядки и определение типов $c_{jrb}(Q)$ станций зарядки в вершине j для старых и новых электробусов типа b, назначенных на маршрут r). Для каждого типа электробусов $b \in B(Q)$ и каждого маршрута $r \in R_b(Q)$ сформировать множество дуг $SR_{rb}(Q)$, в каждой вершине j которых хотя бы одна старая или новая станция зарядки типа $c \in C_{jb}$ назначена типу электробусов b и маршруту r. Для заданных r и b процесс формирования $SR_{rb}(Q)$ начинается с включения в $S_{rb}(Q)$ дуг, соединяющих "обязательные" вершины множества $NM_b \cap r$ и вершины со старыми станциями зарядки $c \in C_b$, назначенными b и r. Предполагается, что типы $c_{jrb}(Q)$ станций зарядки заданы либо случайно сгенерированы для этих обязательных вершин. Далее вычисляем текущее суммарное время зарядки $CT_{rb}(Q) = \sum_{j \in SR_{rb}(Q) \setminus ND} ct_{jbc^*}$, где $c^* = c_{jrb}(Q)$. Если $CT_{rb}(Q) > dc_{rb}$, то Q не может быть достроено до полного допустимого решения. В этом случае если время позволяет, то выполнить шаг 2, иначе выполнить шаг 7. Если $CT_{rb}(Q) \le dc_{rb}$, то выполнить следующие вычисления. Рассмотрим произвольную дугу $(i_1, i_2) \in SR_{rb}(Q)$. Если $ei_{r(i_1,i_2)b} = 1$, то дополнительная станция зарядки для электробусов типа b на маршруте r между вершинами i_1 и i_2 не нужна. Если $ei_{r(i_1,i_2)b} = 0$, то включить в $SR_{rb}(Q)$ дуги (i_1, j) и (j, i_2), $j \in N_c \cap r$, $c \in C_{jb}$, такие что $CT_{rb}(Q) + \min_{c \in C_b}\{ct_{jbc}\} \le dc_{rb}$ и $ei_{r(i_1,j)b} = 1$ или $ei_{r(j,i_2)b} = 1$ с некоторой вероятностью, которая может быть больше, если расстояние от i_1 до j (соответственно от j до i_2) больше. Для допустимости подходящая станция зарядки должна быть установлена по крайней мере в одной вершине j между i_1 и i_2 на этой итерации для пары (r, b). Если никакая вершина не может быть включена и время позволяет, то выполнить шаг 2, иначе выполнить шаг 7. Если дуги (i_1, j) и (j, i_2) включены в $SR_{rb}(Q)$, то определить тип станции зарядки $c^* = c_{jrb}(Q) \in C_b$ для вершины j с некоторой вероятностью так, чтобы $CT_{rb}(Q) + ct_{jbc^*} \le dc_{rb}$. Изменить текущее общее время зарядки $CT_{rb}(Q) := CT_{rb}(Q) + ct_{jbc^*}$. Повторять приведенный процесс включения дуг до тех пор, пока не будет выполнено $ei_{r(i_1,i_2)b} = 1$ для

любой дуги $(i_1, i_2) \in SR_{rb}(Q)$. Сформировать множества $S(Q)$, $S_c(Q)$, $C(Q)$ и $B_{jc}(Q)$, которые являются аналогами таких же множеств, определенных для решения X. Отметим, что в конце этого шага будут выполнены соотношения

$$c_{jrb}(Q) \in C_{jb}, \quad j \in SR_{rb}(Q), \quad r \in R_b(Q), \quad b \in B(Q),$$
$$ei_{r(i,j)b} = 1, \quad (i,j) \in SR_{rb}(Q), \quad r \in R_b(Q), \quad b \in B(Q),$$
$$\sum_{j \in SR_{rb}(Q) \setminus ND} ct_{jbc^*} \le dc_{rb}, \quad c^* = c_{jrb}(Q), \quad r \in R_b(Q), \quad b \in B(Q),$$
$$\sum_{c \in C_b} (nc_{jc} + NC_{jc}(Q)) \ge 1, \quad j \in NM_b \cap S(Q), \quad b \in B(Q) \cup BO,$$

которые являются аналогами ограничений (5), (6), (7) и (10). Также отметим, что $NC_{jc}(Q)$ в последнем соотношении не определено явно, но это соотношение выполняется по определению шага 4.

Шаг 5 (формирование количеств $NC_{jc}(Q)$ новых станций зарядки и частот отправления $FR_{rb}(Q)$ новых электробусов). Определить $NC_{jc}(Q)$ и $FR_{rb}(Q)$ как решение задачи

$$\max_{(NC(Q),FR(Q)) \in \mathcal{NF}(Q)} \frac{V(FR(Q))}{CC(NC(Q),FR(Q)) + OC(NC(Q),FR(Q))},$$

где

$$V(FR(Q)) = \sum_{r \in R(Q)} v_r(Z_r(Q)),$$

$$CC(NC(Q),FR(Q)) = \sum_{c \in C(Q)} \sum_{j \in S_c(Q)} cc_c^{cap} NC_{jc}(Q) + \sum_{r \in R(Q)} \sum_{b \in B_r(Q)} cv_b^{cap} NV_{rb}(Q),$$

$$OC(NC(Q),FR(Q)) = \sum_{c \in C(Q)} \sum_{j \in S_c(Q)} cc_c^{ope} NC_{jc}(Q) + \sum_{r \in R(Q)} \sum_{b \in B_r(Q)} (cv_b^{ope} + ce_{rb}) NV_{rb}(Q),$$

$$Z_r(Q) = \sum_{b \in B_r(Q)} cap_b FR_{rb}(Q),$$
$$NV_{rb}(Q) = \lceil d_r FR_{rb}(Q) \rceil,$$
$$FN_{jb}(Q) = \sum_{r \in R_j \cap R_b(Q)} (fr_{rb} + FR_{rb}(Q)), \quad j \in NE,$$
$$FN_{jb}(Q) = \sum_{r \in R_j \cap R_b(Q)} \alpha_{rbj}^{depot} (nv_{rb} + NV_{rb}(Q))/t_j^{depot}, \quad j \in ND,$$

и допустимая область $\mathcal{NF}(Q)$ определяется ограничениями:

$$lc \le CC(Q) + OC(Q) \le uc, \tag{13}$$

$$Z_r(Q) \le pas_r, \quad r \in R(Q), \tag{14}$$

$$TP_i(Q) = \sum_{j \in M_i(Q) \cap N} \sum_{c \in C(Q)} po_c NC_{jc}(Q) \le o_i, \quad i \in T(Q), \tag{15}$$

$$lfr_{rb} \le FR_{rb}(Q) \le \min \left\{ ufr_{rb}, \frac{pas_r - \sum_{b' \in B(Q), b' \ne b} cap_{b'} lfr_{rb'}}{cap_b} \right\}, \quad r \in R_b(Q), \quad b \in B(Q), \tag{16}$$

$$nv_{rb} + NV_{rb}(Q) = nv_{rb} + \lceil d_r FR_{rb}(Q) \rceil \le uv_{rb}, \quad r \in R_b(Q), \quad b \in B(Q), \tag{17}$$

$$nc_{jc} + NC_{jc}(Q) = \left\lceil \sum_{b \in B_{jc}(Q)} ct_{jbc} FN_{jb}(Q) \right\rceil, \quad j \in S(Q) \cup NO, \quad c \in C(Q), \tag{18}$$

$$\frac{1}{FR_{rb}(Q)} \in Z_+, \quad NC_{jc}(Q) \in Z_+, \quad j \in S(Q) \cup NO, \quad r \in R(Q), \quad b \in B_r(Q), \quad c \in C(Q). \tag{19}$$

Для решения приведенной задачи используется метод роя частиц (Particle Swarm Optimization), см. Clerc [73], Kennedy и Eberhart [74] и Pedersen и Chipperfield [75]. Если решение системы (13)–(19) найдено, то выполнить шаг 6. Если решение не найдено и время позволяет, то выполнить шаг 2, иначе выполнить шаг 7.

Шаг 6 (выбор новых связей станций зарядки и трансформаторов). Для каждой вершины $j \in S(Q) \backslash NO$ обозначим через $L_j(Q)$ множество трансформаторных вершин, связанных с вершиной j в соответствии с частичным решением Q. Обозначим через $M_i(Q) = \{j \in S(Q) \mid i \in L_j(Q)\}$ множество новых нетрансформаторных вершин, связанных с трансформаторной вершиной i. Множества $L_j(Q)$, $j \in S(Q) \backslash NO$, множества $M_i(Q)$, $i \in T$, и множество $T(Q)$ новых трансформаторов отыскиваются случайно. Для каждой вершины $j \in S(Q) \backslash NO$ определяется вероятность p_{ij} связи трансформаторной вершины $i \in TE_j$ с вершиной j. Эта вероятность может быть выше для большей выделенной мощности o_i и она может быть выше для меньшей стоимости $cl_{ij} + cb_i y_i$, где $y_i = 1$, если $i \notin TO$, и $y_i = 0$, если $i \in TO$. Далее связать вершину $j \in S(Q) \backslash NO$ с m вершинами $i \in T$. Вычислить капитальные затраты $CC(Q) = \sum_{j \in S(Q) \backslash NO} \sum_{i \in L_j(Q)} cl_{ij} + \sum_{i \in T(Q) \backslash TO} cb_i + CC\,(NC(Q),\, FR(Q))$. Установить $\mathcal{Q} := \mathcal{Q} \cup \{Q\}$. Если время позволяет, то выполнить шаг 2, иначе выполнить шаг 7.

Шаг 7. Найти $Q^* \in \mathcal{Q}$ такое, что

$$\frac{V(Q^*)}{CC(Q^*) + OC(Q^*)} = \max_{Q \in \mathcal{Q}} \frac{V(Q)}{CC(Q) + OC(Q)}.$$

В рассматриваемых в настоящее время реальных ситуациях мощности множеств, генерируемых на шагах 1–4, ограничены следующими значениями. Количество маршрутов: $|R| \leq 18$. Количество депо: $|ND| \leq 3$. Количество типов станций зарядки: $|C| \leq 2$. Количество маршрутов, пересекающихся в одном месте, подходящем для установки станций зарядки: $|R_j| \leq 8$ для $j \in ND$, $|R_j| \leq 5$ для $j \in NE$. Количество остановок одного маршрута, подходящих для открытия станций зарядки: $|N \cap r| \leq 5$, $r \in R$. Количество типов электробусов, подходящих для обслуживания одного маршрута: $|B_r| \leq 5$, $r \in R$. Количество мест, в которых должна быть открыта хотя бы одна станция зарядки для электробусов одного типа и маршрута: $1 \leq |NM_b \cap r| \leq 3$, $b \in B$, $r \in R$. Количество соединений с трансформаторами: $m \in \{1, 2\}$. Количество мест размещения трансформаторов, подходящих для соединения с одним и тем же местом размещения станций зарядки: $|TE_j| \leq 2$, $j \in N$.

8. Заключение

Исследована сложная оптимизационная задача, возникающая при планировании процесса перехода от традиционного общественного транспорта к электрическому. Описаны принятые предположения, входные и выходные параметры задачи, а также ее математическая модель и рандомизированный алгоритм решения. Дальнейшая работа будет сконцентрирована на разработке детального алгоритма, его программной реализации и тестировании на реальных примерах участников проекта PLATON.

Авторы выражают благодарность ведущему начному сотруднику ОИПИ НАН Беларуси Я.М. Шафранскому за обсуждение постановки задачи и полезные советы по ее моделированию.

СПИСОК ЛИТЕРАТУРЫ

1. http://zeeus.eu/uploads/publications/documents/zeeus-report2017-2018-final.pdf
2. *Nikolič Z., Živanovič Z.* The Contribution and Prospects of the Technical Development on Iimplementation of Electric and Hybrid Vehicles // New Generat. Electric Vehicles. 2012. Chapter 2. P. 27–66.
3. *Li J.-Q.* Battery-Electric Transit Bus Developments and Operations: A Review // Int. J. Sustain. Transp. Technol. 2016. V. 10. P. 157–169.
4. *Ahmad A., Khan Z.A., Alam M.S., Khateeb S.* A Review of the Electric Vehicle Charging Techniques, Standards, Progression and Evolution of EV Technologies in Germany // Smart Sci. 2018. V. 6. No. 1. P. 36–53.
5. *Anderson J.E., Lehne M., Hardinghaus M.* What Electric Vehicle Users Wwant: Real-World Preferences for Public Charging Infrastructure // Int. J. Sustain. Transp. 2018. V. 12. No. 5. P. 341–352.
6. *Todorovic M., Simic M.* Current State of the Transition to Electrical Vehicles // Smart Innovat. Syst. Technol. 2019. V. 98. P. 130–139.
7. *Barnitt R.* Case Study: Ebus Hybrid Electric Buses and Trolleys // National Renewable Energy Laboratory. Technical Report NREL/TP–540–38749. 2006.
8. *Wang X., González J.A.* Assessing Feasibility of Electric Buses in Small and Medium-Sized Communities // Int. J. Sustain. Transp. 2013. V. 7. P. 431–448.

9. *Erkkilä K., Nylund N.-O., Pellikka A.-P., Kallio M., Kallonen S., Ojamo S., Ruotsalainen S., Pietikäinen O., Lajunen A.* eBUS – Electric Bus Test Platform in Finland // EVS27 International Battery, Hybrid and Fuel Cell Electric Vehicle Sympos. Barcelona, Spain, 2013.
10. *Schmidt J., Eisel M., Kolb L.M.* Assessing the Potential of Different Charging Strategies for Electric Vehicle Fleets in Closed Transport Systems // Energ. Policy. 2014. V. 74. P. 179–189.
11. http://zeeus.eu/uploads/publications/documents/zeeus-local-demo-brochures-mergedcompressed.pdf
12. *Foltyński M.* Electric Fleets in Urban Logistics // Procedia – Social and Behavioral Sciences. 2014. V. 151. P. 48–59.
13. *Rogge M., Wollny S., Sauer D.U.* Fast Charging Battery Buses for the Electrification of Urban Public Transport – a Feasibility Study Focusing on Charging Infrastructure and Energy Storage Requirements // Energies. 2015. V. 8. No. 5. P. 4587–4606.
14. *Hanlin J.* Battery Electric Buses Smart Deployment // Zero Emission Bus conf. London, 2016.
15. *Olsson O., Grauers A., Pettersson S.* Method to Analyze Cost Effectiveness of Different Electric Bus Systems // EVS29 Int. Battery, Hybrid and Fuel Cell Electric Vehicle Sympos. Montreal. 2016. P. 1–12.
16. *Eudy L., Jeffers M.* Foothill Transit Battery Electric Bus Demonstration Results: Second Report // National Renewable Energy Laboratory. Technical Report NREL/TP-5400-67698. 2017.
17. *Gao Z., Lin Z., LaClair T.J., Liu C., Li J.-M., Birky A.K., Ward J.* Battery Capacity and Recharging Needs for Electric Buses in City Transit Service // Energy. 2017. V. 122. P. 588–600.
18. *Leou R.-C., Hung J.-J.* Optimal Charging Schedule Planning and Economic Analysis for Electric Bus Charging Stations // Energies. 2017. V. 10. No. 4. 483.
19. *Christensen L., Klauenberg J., Kveiborg O., Rudolph C.* Suitability of Commercial Transport for a Shift to Electric Mobility with Denmark and Germany as Use Cases // Res. Transp. Econ. 2017. V. 64. P. 48–60.
20. *Khan W., Ahmad A., Ahmad F., Alam M.S.* A Comprehensive Review of Fast Charging Infrastructure for Electric Vehicles // Smart Sci. 2018. V. 6. No. 3. P. 256–270.
21. *Xylia M., Silveira S.* The Role of Charging Technologies in Upscaling the Use of Electric Buses in Public Transport: Experiences from Demonstration Projects // Transport Res. A – Pol. 2018. V. 118. P. 399–415.
22. *Gallet M., Massier T., Hamacher T.* Estimation of the Energy Demand of Electric Buses Based on Real-World Data for Large-Scale Public Transport Networks // Appl. Energ. 2018. V. 230. P. 344–356.
23. *Morganti E., Browne M.* Technical and Operational Obstacles to the Adoption of Electric Vans in France and the UK: An Operator Perspective // Transport Policy. 2018. V. 63. P. 90–97.
24. *Feng W., Figliozzi M.* Conventional vs Electric Commercial Vehicle Fleets: A Case Study of Economic and Technological Factors Affecting the Competitiveness of Electric Commercial Vehicles in the USA // Procedia – Social and Behavioral Sciences. 2012. V. 39. P. 702–711.
25. *Feng W., Figliozzi M.* An Economic and Technological Analysis of the Key Factors Affecting the Competitiveness of Electric Commercial Vehicles: A Case Study from the USA Market // Transport Res. C – Emer. 2013. V. 26. P. 135–145.

26. *Hallmark S.L., Wang B., Sperry R.* Comparison of On-Road Emissions for Hybrid and Regular Transit Buses // J. Air Waste Manage. 2013. V. 63. No. 10. P. 1212–1220.

27. *Lajunen A.* Energy Consumption and Cost-Benefit Analysis of Hybrid and Electric City Buses // Transport Res. C – Emer. 2014. V. 38. P. 1–15.

28. *Mohamed M., Garnett R., Ferguson M.R., Kanaroglou P.* Electric Buses: A Review of Alternative Powertrains // Renew. Sust. Energ. Rev. 2016. V. 62. P. 673–684.

29. *Schoch J.* Modeling of Battery Life Optimal Charging Strategies Based on Empirical Mobility Data // IT – Information Technology. 2016. V. 58. No. 1. P. 22–28.

30. *Teoh T., Kunze O., Teo C.-C.* Methodology to Evaluate the Operational Suitability of Electromobility Systems for Urban Logistics Operations // Transportation Res. Procedia. 2016. V. 12. P. 288–300.

31. *Teoh T., Kunze O., Teo C.-C.* Scenario-Based Electric Bus Operation: A Case Study of Putrajaya, Malaysia // Int. J. Transp. Sci. Technol. 2018. V. 7. No. 1. P. 10–25.

32. *Mohamed M., Farag H., El-Taweel N., Ferguson M.* Simulation of Electric Buses on a Full Transit Network: Operational Feasibility and Grid Impact Analysis // Electr. Pow. Syst. Res. 2017. V. 142. P. 163–175.

33. *Marmaras C., Xydas E., Cipcigan E.* Simulation of Electric Vehicle Driver Behaviour in Road Transport and Electric Power Networks // Transport Res. C – Emer. 2017. V. 80. P. 239–256.

34. *Xylia M., Leduc S., Patrizio P., Silveira S., Kraxner F.* Developing a Dynamic Optimization Model for Electric Bus Charging Infrastructure // Transport. Res. Procedia. 2017. V. 27. P. 776–783.

35. *Fiori C., Ahn K., Rakha H.A.* Optimum Routing of Battery Electric Vehicles: Insights Using Empirical Data and Microsimulation // Transport Res. D – Tr. E. 2018. V. 64. P. 262–272.

36. *Alonso M., Amaris H., Germain J.G., Galan J.M.* Optimal Charging Scheduling of Electric Vehicles in Smart Grids by Heuristic Algorithm // Energies. 2014. V. 7. P. 2449–2475.

37. *Wen M., Laporte G., Madsen O.B.G., Norrelund A.V., Olsen A.* Locating Replenishment Stations for Electric Vehicles: Application to Danish Traffic Data // J. Oper. Res. Soc. 2014. V. 65. No. 10. P. 1555–1561.

38. *Yu Z., Chen S., Tong L.* An Intelligent Energy Management System for Large-Scale Charging of Electric Vehicles // CSEE J. Power Energy. 2016. V. 2. No. 1. P. 47–53.

39. *Juan A.A., Mendez C.A., Faulin J., de Armas J., Grasman S.E.* Electric Vehicles in Logistics and Transportation: A Survey on Emerging Environmental, Strategic, and Operational Challenges // Energies. 2016. V. 9. No. 2. 86.

40. *Hiermann G., Puchinger G., Ropke S., Hartl R.F.* The Electric Fleet Size and Mix Vehicle Routing Problem with Time Windows and Recharging Stations // Eur. J. Oper. Res. 2016. V. 252. P. 995–1018.

41. *Quak H., Nesterova N., van Rooijen T.* Possibilities and Barriers for Using Electric-Powered Vehicles in City Logistics Practice // Transport. Res. Procedia. 2016. V. 12. P. 157–169.

42. *Desaulniers G., Errico F., Irnich S., Schneider M.* Exact Algorithms for Electric Vehicle-Routing Problems with Time Windows // Oper Res. 2016. V. 64. No. 6. P. 1388–1405.

43. *Wielinski G., Trépanier M., Morency C.* Electric and Hybrid Car Use in a Free-Floating Car Sharing System // Int. J. Sustain. Transp. 2017. V. 11. No. 3. P. 161–169.

44. *Kunith A., Mendelevitch R., Goehlich D.* Electrification of a City Bus Network – An Optimization Model for Cost-Effective Placing of Charging Infrastructure and Battery Sizing of Fast-Charging Electric Bus Systems // Int. J. Sustain. Transp. 2017. V. 11. No. 10. P. 707–720.

45. *Bruglieri M., Mancini S., Pezzella F., Pisacane O., Suraci S.* A Three-Phase Matheuristic for the Time-Effective Electric Vehicle Routing Problem with Partial Recharges // Electron. Notes Discrete Math. 2017. V. 58. P. 95–102.

46. *Pelletier S., Jabali O., Laporte G.* Battery Electric Vehicles for Freight Distribution: A Survey of Vehicle Technology, Market Penetration, Incentives and Practices // CIRRELT, CIRRELT-2014-43. Montreal. 2014.

47. *Pelletier S., Jabali O., Laporte G.* 50th Anniversary Invited Article. Goods Distribution with Electric Vehicles: Review and Research Perspectives // Transport Sci. 2016. V. 50. No. 1. P. 3–22.

48. *Pelletier S., Jabali O., Laporte G.* Charge Scheduling for Electric Freight Vehicle // Transport Res. B – Meth. 2018. V. 115 P. 246–269.

49. *Pelletier S., Jabali O., Laporte G., Veneroni M.* Battery Degradation and Behaviour for Electric Vehicles: Review and Numerical Analyses of Several Models // Transport Res. B – Meth. 2017. V. 103. P. 158–187.

50. *Froger A., Mendoza J., Jabali O., Laporte G.* New Formulations for the Electric Vehicle Routing Problem with Nonlinear Charging Functions // CIRRELT, CIRRELT-2017-30. Montreal. 2017.

51. *Froger A., Mendoza J., Jabali O., Laporte G.* A Matheuristic for the Eectric Vehicle Routing Problem with Capacitated Charging Stations // CIRRELT, CIRRELT-2017-31. Montreal. 2017.

52. *Xylia M., Leduc S., Patrizio P., Kraxner F., Silveira S.* Locating Charging Infrastructure for Electric Buses in Stockholm // Transport Res. C – Emer. 2017. V. 78. P. 183–200.

53. *Liu Z., Song Z., He Y.* Planning of Fast-Charging Stations for a Battery Electric Bus System under Energy Consumption Uncertainty // Transport Res. Rec. 2018. https://doi.org/10.1177/0361198118772953

54. *Hosseini S., Sarder M.D.* Development of a Bayesian Network Model for Optimal Site Selection of Electric Vehicle Charging Station // Int. J. Elec. Power. 2019. V. 105. P. 110–122.

55. *Wang Y., Bi J., Guan W., Zhao X.* Optimising Route Choices for the Travelling and Charging of Battery Electric Vehicles by Considering Multiple Objectives // Transport Res. D – Tr E. 2018. V. 64. P. 246–261.

56. *Wang Y.-W., Lin C.-C., Lee T.-J.* Electric Vehicle Tour Planning // Transport Res. D – Tr E. 2018. V. 63. P. 121–136.

57. World Health Organization. Quantification of the Health Effects of Exposure to Air Pollution. Report of a WHO Working Group. Bilthoven, Netherlands, 2000.

58. *Sydbom A., Blomberg A., Parnia S., Stenfors N., Sandström T, Dahlén S.E.* Health Effects of Diesel Exhaust Emissions // Eur. Respir. J. 2001. V. 17. No. 4. P. 733–746.

59. HEI Panel on the Health Effects of Traffic-Related Air Pollution. Traffic-Related Air Pollution: A Critical Review of the Literature on Emissions, Exposure, and Health Effects. HEI Special Report 17. Health Effects Institute, Boston, MA, 2010.

60. HEI Diesel Epidemiology Panel. Executive Summary. Diesel Emissions and Lung Cancer: An Evaluation of Recent Epidemiological Evidence for Quantitative Risk Assessment. HEI Special Report 19. Health Effects Institute, Boston, MA, 2015.

61. *Ali R.* Effect of Diesel Emissions on Human Health: A Review // Int. J. Appl. Engin. Res. 2011. V. 6. No. 11. P. 1333–1342.

62. *Chan S., Miranda-Moreno L.F., Patterson Z.* Analysis of GHG Emissions for City Passenger Trains: Is Electricity an Obvious Option for Montreal Commuter Trains? // J. Transport. Technol. 2013. V. 3. No. 2A. P. 17–29.

63. IARC Working Group on the Evaluation of Carcinogenic Risk to Humans. Diesel and Gasoline Engine Exhausts and Some Nitroarenes. IARC Monographs on the Evaluation of Carcinogenic Risks to Humans, No. 105 // Int. Agency for Research on Cancer. Lyon, France, 2014.

64. *Mohner M.* The Hidden Impact of a Healthy-Worker Effect on the Results of the Diesel Exhaust in Miners Study // Eur. J. Epidemiol. 2016. V. 31. No. 8. P. 803–804.

65. *McClellan R.O.* Critique of Health Effects Institute Special Report 19, "Diesel Emissions and Lung Cancer: An Evaluation of Recent Epidemiological Evidence for Quantitative Risk Assessment" // Report. Albuquerque, NM, 2016.

66. *Gaskins A.J., Hart J.E., Mínguez-Alarcón L., Chavarro J.E., Laden F., Coull B.A., Ford J.B., Souter I., Hauser R.* Residential Proximity to Major Roadways and Traffic in Relation to Outcomes of in vitro Fertilization // Environ. Int. 2018. V. 115. P. 239–246.

67. *Jochem P., Plötz P., Ng W.-S., Rothengatter W.* The Contribution of Electric Vehicles to Environmental Challenges in Transport // Transport Res. D – Tr E. 2018. V. 64. P. 1–4.

68. *Steuer R.* Multiple Criteria Optimization: Theory, Computation and Application. N.Y.: John Wiley & Sons, 1985.

69. *Vincke P.* Multicriteria Decision Aid. N.Y.: John Wiley & Sons, 1992.

70. *Roy B.* Multicriteria Methodology for Decision Aiding / Nonconvex Optim. Appl. V. 12. Dordrecht: Kluwer Acad. Publishers, 1996.

71. *Collette Y., Siarry P.* Multiobjective Optimization: Principles and Case Studies. Springer Science & Business Media, 2004.

72. *Ehrgott M.* Multicriteria Optimization. Springer Verlag, 2005.

73. *Clerc M.* Particle Swarm Optimization. John Wiley & Sons, 2010.

74. *Kennedy J., Eberhart R.* Particle Swarm Optimization // Proc. IEEE Int. Conf. on Neural Networks. IV. 1995. P. 1942–1948.

75. *Pedersen M.E.H., Chipperfield A.J.* Simplifying Particle Swarm Optimization // Appl. Soft Comput. 2010. V. 10. P. 618–628.

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

Поступила в редакцию 09.07.2019

После доработки 19.10.2019

Принята к публикации 28.11.2019

© 2020 г. Ю.Н. СОТСКОВ, д-р физ.-мат. наук (sotskov48@mail.ru)
(Объединенный институт проблем информатики НАН Беларуси, Минск)

ОБЛАСТЬ ОПТИМАЛЬНОСТИ ПЕРЕСТАНОВКИ ОБСЛУЖИВАНИЯ НА ОДНОМ ПРИБОРЕ ТРЕБОВАНИЙ С НЕОПРЕДЕЛЕННЫМИ ДЛИТЕЛЬНОСТЯМИ

Исследуется задача оптимизации расписания обслуживания заданного множества требований на одном приборе. При составлении расписания для каждого требования известны нижняя граница и верхняя граница допустимой длительности его обслуживания. В качестве критерия оптимальности расписания рассматривается минимизация суммарного времени обслуживания заданного множества требований. Исследованы свойства области оптимальности перестановки обслуживания требований. Разработаны полиномиальные алгоритмы построения области оптимальности перестановки обслуживания требований и вычисления объема области оптимальности. Определены условия существования пустой области оптимальности для перестановки обслуживания требований. Установлен критерий существования перестановки обслуживания требований с максимально возможным объемом области оптимальности.

Ключевые слова: теория расписаний, неопределенные длительности обслуживания требований, минимизация суммарного времени, область оптимальности.

DOI: 10.31857/S0005231020050050

1. Введение

Оперативно-календарное планирование производства включает этап составления календарных планов выполнения поступивших заказов (составление расписаний обслуживания заданного множества требований) на имеющемся оборудовании (на множестве обслуживающих приборов). Оптимальное расписание производственного процесса является важным фактором его эффективности, поскольку позволяет сократить производственные расходы предприятия, уменьшить время реализации заявок заказчиков на продукцию предприятия, своевременно снабжать прозводственный процесс сырьем, материалами и комплектующими деталями, необходимыми для изготовления конечной продукции предприятия. Оптимальное расписание производственного процесса позволяет уменьшить расходы на хранение сырья, комплектующих деталей и материалов и в итоге повысить эффективность использования имеющихся ресурсов (обслуживающих приборов) и капитала.

Для практических задач оперативно-календарного планирования, как правило, не удается заранее определить точные значения длительностей обслуживания заданных требований, однако имеется возможность оценить снизу и сверху длительности их обслуживания. Для решения таких задач требу-

ются алгоритмы построения расписаний, близких к оптимальным, в условиях неопределенности числовых параметров [1–9].

В разделе 2 данной статьи рассматривается задача построения близкого к оптимальному расписания обслуживания требований множества $\mathcal{J} = \{J_1, \ldots, J_n\}$ на одном приборе с критерием $\sum C_i$ минимизации суммы моментов C_i завершения обслуживания всех требований $J_i \in \mathcal{J}$ при условии, что при построении расписания известны только нижняя граница $p_i^L > 0$ и верхняя граница $p_i^U \geqslant p_i^L$ возможной длительности $p_i \in [p_i^L, p_i^U]$ обслуживания требования J_i. В разделе 3 определяется область оптимальности для перестановки $\pi_k = (J_{k_1}, \ldots, J_{k_n})$ обслуживания требований множества $\mathcal{J}$, которая содержит параллелепипед оптимальности для той же перестановки. Параллелепипед оптимальности для перестановки π_k был исследован в [10–17]. В разделе 4 разработаны полиномиальные алгоритмы для определения области оптимальности для перестановки π_k и для определения объема области оптимальности. Доказаны необходимые и достаточные условия, при выполнении которых область оптимальности для перестановки обслуживания заданных требований является пустым множеством. В разделе 5 исследованы свойства перестановки π_k обслуживания требований множества $\mathcal{J}$, которая имеет максимальный объем области оптимальности.

2. Постановка задачи и обзор известных результатов

Множество требований $\mathcal{J} = \{J_1, \ldots, J_n\}$ необходимо обслужить на одном приборе. Точное значение длительности p_i обслуживания требования $J_i \in \mathcal{J}$ не известно на момент построения расписания обслуживания требований $\mathcal{J}$. Прерывания обслуживания требования запрещены. При реализации расписания обслуживания требований $\mathcal{J}$ длительность p_i обслуживания требования $J_i \in \mathcal{J}$ может принимать любое действительное значение из заданного отрезка $\left[p_i^L, p_i^U\right]$, где $p_i^U \geqslant p_i^L > 0$. Точное значение длительности $p_i \in \left[p_i^L, p_i^U\right]$ становится известным лишь в момент C_i завершения обслуживания требования J_i. Пусть R^n обозначает пространство n-мерных действительных векторов, а R_+^n – подмножество R^n всех неотрицательных n-мерных действительных векторов, $R_+^n \subset R^n$. В пространстве R^n множество всех векторов $(p_1, \ldots, p_n)$ допустимых длительностей обслуживания требований множества $\mathcal{J}$ представляет собой n-мерный параллелепипед, т.е. множество всех векторов $p = (p_1, \ldots, p_n) \in R_+^n$, удовлетворяющих следующей системе неравенств:

$$p_1^L \leqslant p_1 \leqslant p_1^U; \;\; \ldots; \;\; p_n^L \leqslant p_n \leqslant p_n^U.$$

Множество допустимых длительностей $(p_1, \ldots, p_n) = p \in R_+^n$ обслуживания требований множества $\mathcal{J}$ представим как декартово произведение отрезков $[p_i^L, p_i^U]$: $\times_{i=1}^n [p_i^L, p_i^U] := [p_1^L, p_1^U] \times \ldots \times [p_n^L, p_n^U] = T = \left\{p \in R_+^n : p_i^L \leqslant p_i \leqslant p_i^U, \; i \in \{1, \ldots, n\}\right\}$. Вектор $p \in T$ будем называть сценарием. Пусть множество $\Pi = \{\pi_1, \ldots, \pi_{n!}\}$ содержит все перестановки $\pi_k = (J_{k_1}, \ldots, J_{k_n})$ требований $\mathcal{J}$. Для фиксированных сценария $p \in T$ и перестановки $\pi_k \in \Pi$ пусть $C_i = C_i(\pi_k, p)$ обозначает момент завершения обслуживания требова-

ния $J_i \in \mathcal{J}$ в активном расписании [5, 18], однозначно определенном перестановкой π_k. Критерий $\sum C_i$ обозначает минимизацию суммы моментов завершения обслуживания требований $\mathcal{J}$:

$$\sum_{J_i \in \mathcal{J}} C_i(\pi_t, p) = \min_{\pi_k \in \Pi} \left\{ \sum_{J_i \in \mathcal{J}} C_i(\pi_k, p) \right\}. \tag{1}$$

Указанная в (1) перестановка $\pi_t = (J_{t_1}, \ldots, J_{t_n}) \in \Pi$ является оптимальной перестановкой обслуживания требований $\mathcal{J}$ для фиксированного сценария $p \in T$.

Поскольку число требований $n = |\mathcal{J}|$ известно на момент составления расписания π_t обслуживания требований $\mathcal{J}$, то критерий $\sum C_i$ означает также минимизацию среднего времени $\frac{\sum_{J_i \in \mathcal{J}} C_i(\pi_k, p)}{n}$ обслуживания требований множества $\mathcal{J}$.

Сформулированная неопределенная задача обозначается как $1|p_i^L \leqslant p_i \leqslant p_i^U| \sum C_i$, если использовать введенную в [19] трехпозиционную форму $\alpha|\beta|\gamma$ обозначения задач теории расписаний. Здесь и далее α обозначает тип обслуживающей системы, β – условия обслуживания требований, а γ – целевую функцию.

Если сценарий $p \in T$ известен к моменту построения расписания, иными словами, для каждого требования $J_i \in \mathcal{J}$ выполняется равенство $\left[p_i^L, p_i^U\right] = [p_i, p_i]$, то неопределенная задача $1|p_i^L \leqslant p_i \leqslant p_i^U| \sum C_i$ превращается в детерминированную задачу $1|| \sum C_i$. Обозначение $1|p| \sum C_i$ будем использовать для детерминированной задачи $1|| \sum C_i$, в которой зафиксирован сценарий $p \in T$ к моменту построения оптимального расписания. Как показано в [20], задачу $1|p| \sum C_i$ можно решить за время $O(n \log n)$ в силу следующего критерия оптимальности перестановки $\pi_k \in \Pi$.

Теорема 1. Перестановка $\pi_k = (J_{k_1}, \ldots, J_{k_n}) \in \Pi$ обслуживания требований множества $\mathcal{J}$ является оптимальной для задачи $1|p| \sum C_i$ тогда и только тогда, когда выполняются следующие неравенства:

$$p_{k_1} \leqslant \ldots \leqslant p_{k_n}. \tag{2}$$

Если $p_{k_u} < p_{k_v}$, то требование J_{k_u} предшествует требованию J_{k_v} в любой перестановке $\pi_k \in \Pi$, которая является оптимальной для задачи $1|p| \sum C_i$.

Поскольку в неопределенной задаче $1|p_i^L \leqslant p_i \leqslant p_i^U| \sum C_i$ сценарий $p \in T$ не фиксирован на момент построения перестановки $\pi_k \in \Pi$ обслуживания требований $\mathcal{J}$, то точное время C_i завершения обслуживания требования $J_i \in \mathcal{J}$ определить невозможно до момента завершения обслуживания требования J_i. Следовательно, значение целевой функции $\sum C_i$ для перестановки π_k обслуживания требований $\mathcal{J}$ остается неизвестным вплоть до момента завершения обслуживания всех требований множества $\mathcal{J}$, если для требований $J_i \in \mathcal{J}$ выполняются строгие неравенства $p_i^L < p_i^U$.

Для неопределенной задачи $\alpha|p_i^L \leqslant p_i \leqslant p_i^U|\gamma$, как правило, не существует расписания, которое оставалось бы оптимальным для всех сценариев из

множества T мощности, большей единицы. Поэтому в литературе по исследованию таких некорректных задач теории расписаний вводят дополнительные целевые функции и (или) используют те или иные предположения. В частности, использование стохастического метода для решения задачи $\alpha|p_i^L \leqslant p_i \leqslant p_i^U|\gamma$ основано на предположении о том, что все длительности обслуживания требований являются случайными величинами с известными законами распределения вероятностей [7, 18].

Если нет достаточной информации о распределении вероятностей случайных длительностей, то используют другие методы [5, 21]. Так, при использовании робастного метода [2, 22–24] лицо, принимающее решение, предпочитает исключить наихудший из возможных сценариев для искомого расписания, которое предлагается для реализации [4, 9, 23]. Для любой перестановки $\pi_k \in \Pi$ и любого сценария $p \in T$ разность $\gamma_p^k - \gamma_p^t = r(\pi_k, p)$ принято называть сожалением (regret) для перестановки π_k со значением целевой функции γ, равным γ_p^k для сценария p. Значение $Z(\pi_k) = \max\{r(\pi_k, p) : p \in T\}$ называется абсолютным сожалением в наихудшем случае (worst-case absolute regret), а значение $Z^*(\pi_k) = \max\left\{\frac{r(\pi_k,p)}{\gamma_p^t} : p \in T\right\}$ – относительным сожалением в наихудшем случае (worst-case relative regret).

Несмотря на то, что задача $1|p|\sum w_iC_i$ полиномиально разрешима [20] при любых весах $w_i > 0$, заданных для множества требований $J_i \in \mathcal{J}$, построение перестановки $\pi_t \in S$ с наименьшим значением $Z(\pi_t)$ (или перестановки с наименьшим значением $Z^*(\pi_t)$) для задачи $1|p_i^L \leqslant p_i \leqslant p_i^U|\sum C_i$ является NP-трудной задачей даже для двух допустимых сценариев [21, 24, 25]. В [3] разработан метод ветвей и границ для построения перестановки π_t с минимальным значением абсолютного сожаления $Z(\pi_t)$ для задачи $1|p_i^L \leqslant p_i \leqslant p_i^U|\sum w_iC_i$. Вычислительные эксперименты на компьютере показали, что предложенный метод позволяет строить такую перестановку π_t для задачи $1|p_i^L \leqslant p_i \leqslant p_i^U|\sum w_iC_i$ при условии, что число n заданных требований не превосходит 40.

Использование метода, основанного на нечетких множествах, позволяет строить расписания, которые являются оптимальными при нечетких длительностях обслуживания требований множества $\mathcal{J}$ на приборах заданного множества $\mathcal{M}$ [8, 9, 26]. Этот метод позволяет решать неопределенные задачи теории расписаний только при малых значениях числа n заданных требований.

В [1] было протестировано несколько эвристик для решения задачи $1|p_i^L \leqslant p_i \leqslant p_i^U|\sum w_iC_i$. В проведенных вычислительных экспериментах для $n \in \{100, 300, 400, 600, 800, 1000\}$ использовались различные законы распределения вероятностей для выбора фактических длительностей обслуживания требований. Вычислительные эксперименты позволили выделить наилучшую эвристику $U2$, которая по всем решенным тестовым примерам давала среднюю погрешность целевой функции $\sum w_iC_i$, равную 1,1% от значения целевой функции, полученного для фактических длительностей обслуживания требований.

Метод, основанный на устойчивости расписаний [6, 10–17] включает анализ устойчивости оптимальных расписаний к возможным вариациям дли-

тельностей обслуживания требований множества $\mathcal{J}$ и построение на основе такого анализа минимального доминирующего множества расписаний. Для любого сценария $p \in T$ минимальное доминирующее множество содержит хотя бы одно оптимальное расписание. В отличие от стохастического метода, робастного метода и метода, основанного на нечетких множествах, цель метода, основанного на устойчивости расписаний, заключается в поиске расписания, которое является оптимальным для максимально возможного числа допустимых сценариев из заданного множества T. В частности, если существует одноэлементное доминирующее множество $\{\pi_t\}$ для задачи $\alpha|p_i^L \leqslant p_i \leqslant p_i^U|\gamma$, то расписание π_t является оптимальным для задачи $\alpha|p|\gamma$ при всех сценариях $p \in T$, несмотря на неопределенность длительностей обслуживания требований множества $\mathcal{J}$.

В [10–12, 14, 15] метод, основанный на устойчивости расписаний, использовался для решения задачи $1|p_i^L \leqslant p_i \leqslant p_i^U| \sum w_i C_i$. В [15] доказан критерий существования одноэлементного доминирующего множества для задачи $1|p_i^L \leqslant p_i \leqslant p_i^U| \sum w_i C_i$. В [12] для $n \in \{50, 100, 500, 1000, 5000, 10000\}$ были случайным образом сгенерированы сложные примеры задачи $1|p_i^L \leqslant p_i \leqslant p_i^U| \sum w_i C_i$, которые были решены приближенно разработанным алгоритмом MAX-OPTBOX со средней погрешностью, равной 1,5%. В [10] примеры задачи $1|p_i^L \leqslant p_i \leqslant p_i^U| \sum C_i$ были случайным образом сгенерированы для $n \in \{10, 20, \ldots, 100, 200, \ldots, 1000, 2000, \ldots, 10000\}$ и решены приближенно разработанным алгоритмом 3 со средней погрешностью, равной 0,74%.

Алгоритм MAX-OPTBOX (алгоритм 3) строит перестановку $\pi_k \in \Pi$, для которой параллелепипед оптимальности имеет наибольший полупериметр (наибольший относительный полупериметр соответственно). Алгоритм 3 учитывает следующую особенность целевой функции $\sum C_i$: увеличение δ значения целевой функции в результате нарушения неравенства (2) из-за фактической длительности p_{k_i} требования J_{k_i} влечет увеличение значения целевой функции на величину $\delta(n - i + 1)$. Поэтому ошибка при выборе порядка обслуживания требования J_{k_i} важнее такой же по величине ошибки при выборе порядка обслуживания требования J_{k_j}, если $i < j$.

Определение параллелепипеда оптимальности приведено в [10, 12]. Пусть $M = (k_{i_1}, \ldots, k_{i_{|M|}})$ представляет собой упорядоченное подмножество множества $\{1, \ldots, n\}$, для которого выполняются следующие соотношения: $\{k_1, \ldots, k_{|M|}\} \subseteq \{1, \ldots, n\}$, $|M| \leqslant n$ и $k_{i_1} < \ldots < k_{i_{|M|}}$.

Определение 1. Максимальный по включению параллелепипед

$$\mathcal{OB}(\pi_k, T) = \left[l_{k_{i_1}}^{opt}, u_{k_{i_1}}^{opt}\right] \times \ldots \times \left[l_{k_{i_{|M|}}}^{opt}, u_{k_{i_{|M|}}}^{opt}\right] =: \times_{k_{i_r} \in M} \left[l_{k_{i_r}}^{opt}, u_{k_{i_r}}^{opt}\right] \subseteq T$$

называется параллелепипедом оптимальности для перестановки $\pi_k = (J_{k_{i_1}}, \ldots, J_{k_{i_n}}) \in \Pi$, *если перестановка* π_k, *будучи оптимальной для задачи* $1|p| \sum C_i$ *со сценарием* $p = (p_1, \ldots, p_n) \in T$, *остается оптимальной и для задачи* $1|p'| \sum C_i$ *с любым сценарием* $p' = (p'_1, \ldots, p'_n) \in [p_1, p_1] \times \ldots \times \ldots \times \left[p_{k_{i_r}-1}, p_{k_{i_r}-1}\right] \times \left[l_{k_{i_r}}^{opt}, u_{k_{i_r}}^{opt}\right] \times \left[p_{k_{i_r}+1}, p_{k_{i_r}+1}\right] \times \ldots \times [p_n, p_n] \in T$. *Если нет сценария* $p \in T$, *для которого перестановка* π_k *является оптимальной для задачи* $1|p| \sum C_i$, *то полагаем* $\mathcal{OB}(\pi_k, T) = \emptyset$.

Любая вариация $p'_{k_{i_r}}$ длительности $p_{k_{i_r}}$ обслуживания требования $J_{k_{i_r}} \in$ $\in \mathcal{J}$, принадлежащая максимальному (по включению) отрезку $\left[l^{opt}_{k_{i_r}}, u^{opt}_{k_{i_r}}\right]$, указанному в определении 1, гарантирует оптимальность перестановки $\pi_k \in$ $\in \Pi$ для любого сценария $p' = (p'_1, \ldots, p'_n)$, если для него выполняется включение $p'_{k_{i_r}} \in \left[l^{opt}_{k_{i_r}}, u^{opt}_{k_{i_r}}\right]$. Максимальный отрезок $\left[l^{opt}_{k_{i_r}}, u^{opt}_{k_{i_r}}\right]$ длины $u^{opt}_{k_{i_r}} - l^{opt}_{k_{i_r}} \geqslant 0$, $l^{opt}_{k_{i_r}} \leqslant u^{opt}_{k_{i_r}}$, указанный в определении 1, будем называть отрезком оптимальности для требования $J_{k_{i_r}} \in \mathcal{J}$ в перестановке π_k. Если не существует такого отрезка оптимальности $\left[l^{opt}_{k_{i_r}}, u^{opt}_{k_{i_r}}\right]$, $l^{opt}_{k_{i_r}} \leqslant u^{opt}_{k_{i_r}}$, для требования $J_{k_{i_r}} \in \mathcal{J}$, то будем говорить, что требование $J_{k_{i_r}}$ не имеет отрезка оптимальности в перестановке π_k.

3. Область оптимальности перестановки требований $\pi_k \in \Pi$

Определим область оптимальности $\mathcal{OR}(\pi_k, T)$ для перестановки $\pi_k \in$ $\in \Pi$, которая содержит параллелепипед оптимальности перестановки π_k: $\mathcal{OB}(\pi_k, T) \subseteq \mathcal{OR}(\pi_k, T)$.

Определение 2. Максимальное по включению замкнутое подмножество $\mathcal{OR}(\pi_k, T) \subseteq T$ множества R^n_+ называется областью оптимальности для перестановки $\pi_k = (J_{k_1}, \ldots, J_{k_n}) \in \Pi$ относительно T, если перестановка π_k является оптимальной для задачи $1|p|\sum C_i$ при любом сценарии $p = (p_1, \ldots, p_n) \in \mathcal{OR}(\pi_k, T)$. Если не существует сценария $p \in T$, для которого перестановка π_k является оптимальной для задачи $1|p|\sum C_i$, то полагаем $\mathcal{OR}(\pi_k, T) = \emptyset$.

В силу теоремы 1 будем различать три типа отрезков для каждого требования $J_{k_r} \in \mathcal{J}$ в фиксированной перестановке $\pi_k = (J_{k_1}, \ldots, J_{k_n}) \in \Pi$, а именно:

отрезок оптимальности $\left[l^{opt}_{k_r}, u^{opt}_{k_r}\right] \subseteq \left[p^L_{k_r}, p^U_{k_r}\right]$;

отрезок условной оптимальности $\left[l^{copt}_{k_r}, u^{copt}_{k_r}\right] \subseteq \left[p^L_{k_r}, p^U_{k_r}\right]$ и

отрезок неоптимальности $\left[l^{non}_{k_r}, u^{non}_{k_r}\right] \subseteq \left[p^L_{k_r}, p^U_{k_r}\right]$.

Отрезок оптимальности $\left[l^{opt}_{k_r}, u^{opt}_{k_r}\right]$ для требования J_{k_r} в перестановке π_k указан в определении 1 параллелепипеда оптимальности.

Отрезком неоптимальности для требования J_{k_r} в перестановке $\pi_k =$ $= (J_{k_1}, \ldots, J_{k_n}) \in \Pi$ будем называть максимальный по включению отрезок $\left[l^{non}_{k_r}, u^{non}_{k_r}\right] \subseteq \left[p^L_{k_r}, p^U_{k_r}\right]$, для которого перестановка $\pi_k = (J_{k_1}, \ldots, J_{k_n})$ не может быть оптимальной для задачи $1|p'|\sum C_i$ ни при каком сценарии $p' =$ $= (p'_1, \ldots, p'_n) \in T$ таком, что

$$(p'_1, \ldots, p'_n) \in \left\{\times_{i=1}^{r-1} \left[p^L_{k_i}, p^U_{k_i}\right]\right\} \times \left[l^{non}_{k_r}, u^{non}_{k_r}\right] \times \left\{\times_{i=r+1}^{n} \left[p^L_{k_i}, p^U_{k_i}\right]\right\}.$$

В силу необходимых и достаточных условий (2) оптимальности перестановки $\pi_k \in \Pi$ для задачи $1|p|\sum C_i$ для каждого отрезка неоптимальности $[l^{non}_{k_r}, u^{non}_{k_r}]$ либо существует требование $J_{k_v} \in \mathcal{J}$ такое, что $r < v$ и выполняют-

ся соотношения

$$p^U_{k_v} = l^{non}_{k_r} < u^{non}_{k_r} = p^U_{k_r}, \tag{3}$$

либо существует требование $J_{k_w} \in \mathcal{J}$ такое, что $w < r$ и выполняются соотношения

$$p^L_{k_r} = l^{non}_{k_r} < u^{non}_{k_r} = p^L_{k_w}. \tag{4}$$

Из определения 1 также следует, что открытый интервал неоптимальности $(l^{non}_{k_r}, u^{non}_{k_r})$ для требования J_{k_r} в перестановке $\pi_k = (J_{k_1}, \ldots, J_{k_n}) \in \Pi$ не может иметь общих точек с отрезком оптимальности $\left[l^{opt}_{k_r}, u^{opt}_{k_r}\right]$, т.е.

$$(l^{non}_{k_r}, u^{non}_{k_r}) \bigcap \left[l^{opt}_{k_r}, u^{opt}_{k_r}\right] = \emptyset. \tag{5}$$

Для демонстрации приведенных определений будем использовать пример 1 задачи $1|p^L_i \leqslant p_i \leqslant p^U_i| \sum C_i$ с 18 требованиями, $n = 18$. Отрезки $\left[p^L_i, p^U_i\right]$, определяющие допустимые длительности обслуживания требований $J_i \in \mathcal{J} = \{J_1, \ldots, J_{18}\}$, заданы в таблице. Отрезки $\left[p^L_i, p^U_i\right]$ представлены также на рис. 1 в системе прямоугольных координат для перестановки $\pi_1 = (J_1, \ldots, J_{18}) \in \Pi$. На рис. 1 ось абсцисс определяет отрезки $[p^L_i, p^U_i]$ допустимых длительностей обслуживания требований $J_i \in \mathcal{J}$. На оси ординат указаны требования J_i из множества $\mathcal{J}$.

Отрезком условной оптимальности для требования J_{k_r} в перестановке $\pi_k = (J_{k_1}, \ldots, J_{k_n}) \in \Pi$ называется максимальный по включению отрезок $\left[l^{copt}_{k_r}, u^{copt}_{k_r}\right] \subseteq \left[p^L_{k_r}, p^U_{k_r}\right]$ такой, что любая точка $p^*_{k_r} \in \left[l^{copt}_{k_r}, u^{copt}_{k_r}\right]$ не принадлежит открытому интервалу неоптимальности, $p^*_{k_r} \notin (l^{non}_{k_r}, u^{non}_{k_r})$, и при этом существует хотя бы одно требование $J_{k_d} \in \mathcal{J}$, $d \neq r$, для которого выполняется включение $p^*_{k_r} \in \left[p^L_{k_d}, p^U_{k_d}\right]$.

Из определения отрезка условной оптимальности следует, что существуют точки $p^*_{k_r} \in \left[l^{copt}_{k_r}, u^{copt}_{k_r}\right]$ такие, что перестановка $\pi_k \in \Pi$ является оптимальной для задачи $1|p'| \sum C_i$ со сценарием $p' = (p'_1, \ldots, p'_n) \in [p_{k_1}, p_{k_1}] \times \ldots$ $\ldots \times [p_{k_r-1}, p_{k_r-1}] \times [p^*_{k_r}, p^*_{k_r}] \times [p_{k_r+1}, p_{k_r+1}] \times \ldots \times [p_{k_n}, p_{k_n}]$, и при этом существуют другие точки $p^{**}_{k_r} \in \left[l^{copt}_{k_r}, u^{copt}_{k_r}\right]$ такие, что перестановка $\pi_k \in \Pi$ не является оптимальной для задачи $1|p''| \sum C_i$ со сценарием $p'' = (p''_1, \ldots, p''_n) \in$ $\in [p_{k_1}, p_{k_1}] \times \ldots \times [p_{k_r-1}, p_{k_r-1}] \times [p^{**}_{k_r}, p^{**}_{k_r}] \times [p_{k_r+1}, p_{k_r+1}] \times \ldots \times [p_{k_n}, p_{k_n}]$. На рис. 1 отрезки условной оптимальности для требований $J_i \in \mathcal{J}$ в перестановке $\pi_1 = (J_1, \ldots, J_{18})$ заштрихованы горизонтальными отрезками.

Исходные данные для примера 1 задачи $1|p^L_i \leqslant p_i \leqslant p^U_i| \sum C_i$

i	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
p^L_i	1	3	2	7	2	4	11	12	11	14	7	27	30	9	36	37	38	21
p^U_i	8	5	8	9	10	6	15	15	20	18	23	34	32	40	42	40	40	41

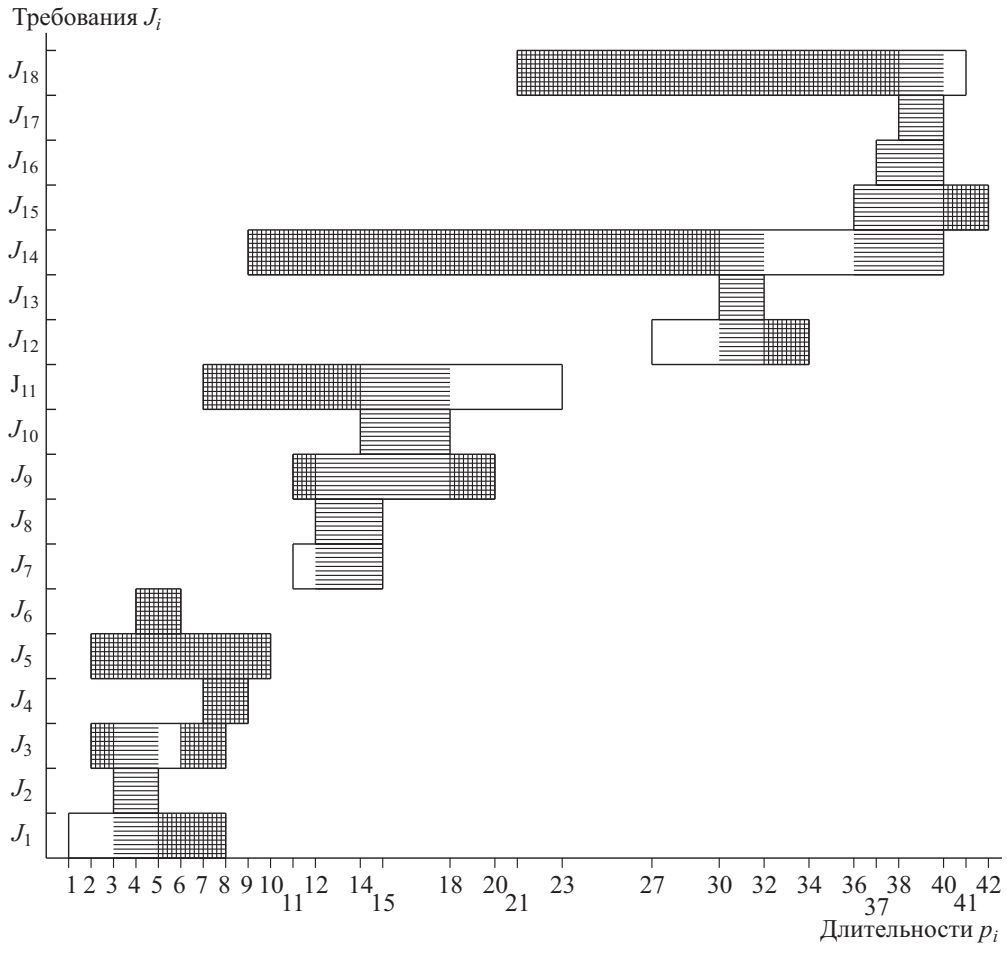


Рис. 1. Отрезки оптимальности, отрезки условной оптимальности (заштрихованы горизонтальными отрезками) и отрезки неоптимальности (заштрихованы горизонтальными и вертикальными отрезками) для требований $J_i \in \mathcal{J}$ в перестановке $\pi_1 = (J_1, \ldots, J_{18})$ для примера 1 задачи $1|p_i^L \leqslant p_i \leqslant p_i^U| \sum C_i$.

Отметим также, что в любой фиксированной перестановке $\pi_k = (J_{k_1}, \ldots$ $\ldots, J_{k_n}) \in \Pi$ отрезок $\left[l_{k_r}^{copt}, u_{k_r}^{copt}\right]$ условной оптимальности для требования J_{k_r} не имеет общих точек с открытым интервалом оптимальности $(l_{k_r}^{opt}, u_{k_r}^{opt})$ и с открытым интервалом неоптимальности $(l_{k_r}^{non}, u_{k_r}^{non})$. Иными словами, выполняются следующие равенства:

$$\left[l_{k_r}^{copt}, u_{k_r}^{copt}\right] \bigcap \left(l_{k_r}^{opt}, u_{k_r}^{opt}\right) = \emptyset, \tag{6}$$

$$\left[l_{k_r}^{copt}, u_{k_r}^{copt}\right] \bigcap \left(l_{k_r}^{non}, u_{k_r}^{non}\right) = \emptyset. \tag{7}$$

Если не существует отрезка $[l_{k_r}^{copt}, u_{k_r}^{copt}]$, $l_{k_r}^{copt} < u_{k_r}^{copt}$, условной оптимальности для требования $J_{k_r} \in \mathcal{J}$ в перестановке π_k, то будем говорить, что требование J_{k_r} не имеет условной оптимальности в перестановке π_k.

На рис. 1 для всех требований $J_i \in \mathcal{J}$ в перестановке $\pi_1 = (J_1, \ldots, J_{18})$ представлены отрезки оптимальности, отрезки условной оптимальности и отрезки неоптимальности. Отрезки неоптимальности для требований $J_i \in \mathcal{J}$ в перестановке π_1 зашрихованы дважды (горизонтальными и вертикальными отрезками).

Замечание 1. В силу теоремы 1 для каждого требования $J_i \in \mathcal{J}$ в перестановке $\pi_k \in \Pi$ может существовать не более одного отрезка оптимальности, не более двух отрезков условной оптимальности и не более двух отрезков неоптимальности.

Как показано на рис. 1, для требования J_3 в перестановке $\pi_1 \in \Pi$ имеется два отрезка неоптимальности $[2, 3]$ и $[6, 8]$, один отрезок $[3, 5]$ условной оптимальности и один отрезок $[5, 6]$ оптимальности. Для требования J_5 в перестановке π_1 имеется два отрезка неоптимальности $[2, 7]$ и $[6, 10]$ с непустым пересечением $[6, 7] = [2, 7] \bigcap [6, 10]$.

Из определений отрезков оптимальности, неоптимальности и условной оптимальности для требования J_{k_r}, $r \in \{1, \ldots, n\}$, в перестановке $\pi_k \in \Pi$ следует

Лемма 1. Отрезок $[p^L_{k_r}, p^U_{k_r}]$ длительностей обслуживания требования $J_{k_r} \in \mathcal{J}$ можно представить в виде объединения всех отрезков оптимальности, неоптимальности и условной оптимальности для требования J_{k_r}, $r \in \{1, \ldots, n\}$, в перестановке $\pi_k = (J_{k_1}, \ldots J_{k_n}) \in \Pi$. Для любого отрезка неоптимальности $[l^{non}_{k_r}, u^{non}_{k_r}]$ выполняется хотя бы одно из равенств $l^{non}_{k_r} = p^L_{k_r}$ или $u^{non}_{k_r} = p^U_{k_r}$.

Построение области оптимальности $\mathcal{OR}(\pi_k, T)$ перестановки $\pi_k = (J_{k_1}, \ldots$ $\ldots, J_{k_n}) \in \Pi$ можно свести к построению области оптимальности перестановки π_k для соответствующей задачи $1|\widehat{p}^L_i \leqslant p_i \leqslant \widehat{p}^U_i| \sum C_i$ с сокращенными отрезками допустимых длительностей $\left[\widehat{p}^L_i, \widehat{p}^U_i\right] \subseteq \left[p^L_i, p^U_i\right]$. Сокращенные отрезки $\left[\widehat{p}^L_{k_r}, \widehat{p}^U_{k_r}\right]$, $J_{k_r} \in \mathcal{J}$, для перестановки $\pi_k = (J_{k_1}, \ldots, J_{k_n})$ определяются по формулам

$$\widehat{p}^L_{k_r} = \max_{1 \leqslant j \leqslant r \leqslant n} p^L_{k_j}, \quad \widehat{p}^U_{k_r} = \min_{1 \leqslant r \leqslant j \leqslant n} p^U_{k_j} \tag{8}$$

для каждого требования $J_{k_r} \in \{J_{k_1}, \ldots, J_{k_n}\} = \mathcal{J}$. Множество сокращенных сценариев, определенных по формулам (8), обозначим как $\widehat{T} = \left[\widehat{p}^L_1, \widehat{p}^U_1\right] \times \ldots$ $\ldots \times \left[\widehat{p}^L_n, \widehat{p}^U_n\right]$.

Теорема 2. Область оптимальности $\mathcal{OR}(\pi_k, T)$ для перестановки $\pi_k = (J_{k_1}, \ldots, J_{k_n}) \in \Pi$ для задачи $1|p^L_i \leqslant p_i \leqslant p^U_i| \sum C_i$ совпадает с областью оптимальности $\mathcal{OR}(\pi_k, \widehat{T})$ для той же перестановки для задачи $1|\widehat{p}^L_i \leqslant p_i \leqslant \widehat{p}^U_i| \sum C_i$ с множеством $\widehat{T}$ допустимых сценариев.

Доказательство. Из необходимых и достаточных условий (2) оптимальности перестановки π_k для задачи $1|p| \sum C_i$ получаем, что из выполнения соотношений

$$p^L_{k_r} \leqslant p_{k_r} < \widehat{p}^L_{k_r} = \max_{1 \leqslant j \leqslant r \leqslant n} p^L_{k_j}$$

хотя бы для одной длительности p_{k_r} следует, что перестановка $\pi_k = (J_{k_1}, \ldots$ $\ldots, J_{k_r}, \ldots, J_{k_n})$ не может быть оптимальной для задачи $1|p|\sum C_i$ с каким-либо сценарием $p = (p_1, \ldots, p_n) \in \left\{\times_{i=1}^{r-1}\left[p_{k_i}^L, p_{k_i}^U\right]\right\} \times \left[l_{k_r}^{non}, u_{k_r}^{non}\right] \times$ $\times \left\{\times_{i=r+1}^{n}[p_{k_i}^L, p_{k_i}^U]\right\}$.

Аналогично, из выполнения соотношений

$$\min_{1 \leqslant r \leqslant j \leqslant n} p_{k_j}^U = \widehat{p}_{k_r}^U < p_{k_r} \leqslant p_{k_r}^U$$

хотя бы для одной длительности p_{k_r} следует, что перестановка $\pi_k =$ $= (J_{k_1}, \ldots, J_{k_r}, \ldots, J_{k_n})$ не может быть оптимальной для задачи $1|p|\sum C_i$ с каким-либо сценарием $p = (p_1, \ldots, p_n) \in \left\{\times_{i=1}^{r-1}\left[p_{k_i}^L, p_{k_i}^U\right]\right\} \times \left[l_{k_r}^{non}, u_{k_r}^{non}\right] \times$ $\times \left\{\times_{i=r+1}^{n}\left[p_{k_i}^L, p_{k_i}^U\right]\right\}$.

Таким образом, справедливо следующее утверждение: *множество всех сценариев $p \in T$, для которых перестановка π_k является оптимальной для задачи $1|p_i^L \leqslant p_i \leqslant p_i^U|\sum C_i$, содержится во множестве всех сценариев $p \in T$, для которых перестановка π_k является оптимальной для задачи $1|\widehat{p}_i^L \leqslant p_i \leqslant \widehat{p}_i^U|\sum C_i$ с множеством $\widehat{T}$ допустимых сценариев.*

Из включения $\widehat{T} \subseteq T$ следует обратное утверждение:

Утверждение. Множество всех сценариев $p \in T$, для которых перестановка π_k является оптимальной для задачи $1|\widehat{p}_i^L \leqslant p_i \leqslant \widehat{p}_i^U|\sum C_i$, содержится во множестве всех сценариев $p \in T$, для которых перестановка π_k является оптимальной для задачи $1|p_i^L \leqslant p_i \leqslant p_i^U|\sum C_i$.

Из доказанных утверждений следует, что для исходной задачи $1|p_i^L \leqslant$ $\leqslant p_i \leqslant p_i^U|\sum C_i$ и для задачи $1|\widehat{p}_i^L \leqslant p_i \leqslant \widehat{p}_i^U|\sum C_i$ с множеством сценариев $\widehat{T} = \left[\widehat{p}_1^L, \widehat{p}_1^U\times\right] \times \ldots \times \left[\widehat{p}_n^L, \widehat{p}_n^U\times\right]$ области оптимальности совпадают для любой фиксированной перестановки $\pi_k \in \Pi$: $\mathcal{OR}(\pi_k, T) = \mathcal{OR}(\pi_k, \widehat{T})$. Теорема 2 доказана.

Из определения 2 и теоремы 2 нетрудно получить следующее утверждение.

Лемма 2. Для задачи $1|\widehat{p}_i^L \leqslant p_i \leqslant \widehat{p}_i^U|\sum C_i$ с множеством сценариев $\widehat{T}$ открытый интервал оптимальности $\left(l_{k_r}^{opt}, u_{k_r}^{opt}\right)$ для требования J_{k_r} в перестановке $\pi_k \in \Pi$ не имеет общих точек с отрезком $\left[p_{k_d}^L, p_{k_d}^U\right]$ допустимых длительностей любого другого требования $J_{k_d} \in \mathcal{J}$, $d \neq r$, т.е. выполняется следующее равенство:

$$(l_{k_r}^{opt}, u_{k_r}^{opt}) \bigcap [p_{k_d}^L, p_{k_d}^U] = \emptyset. \tag{9}$$

Докажем небходимые и достаточные условия, при выполнении которых область оптимальности для перестановки $\pi_k \in \Pi$ является пустым множеством.

Теорема 3. Область оптимальности $\mathcal{OR}(\pi_k, T)$ для перестановки $\pi_k = (J_{k_1}, \ldots, J_{k_n}) \in \Pi$ является пустым множеством, $\mathcal{OR}(\pi_k, T) = \emptyset$, тогда и только тогда, когда существует хотя бы одно требование $J_{k_r} \in \mathcal{J}$, $r \in \{1, \ldots, n\}$, в перестановке $\pi_k = (J_{k_1}, \ldots, J_{k_n}) \in \Pi$, которое не имеет условной оптимальности и одновременно не имеет отрезка оптимальности.

Доказательство.

Достаточность. Пусть существует требование $J_{k_r} \in \mathcal{J}$ в перестановке $\pi_k = (J_{k_1}, \ldots, J_{k_n}) \in \Pi$, которое не имеет условной оптимальности и не имеет отрезка оптимальности. По лемме 1 получаем $\left[l_{k_r}^{non}, u_{k_r}^{non}\right] = \left[p_{k_r}^L, p_{k_r}^U\right] \neq \emptyset$. Следовательно, либо существует требование $J_{k_v} \in \mathcal{J}$ такое, что $r < v$ и выполняются соотношения (3), либо существует требование $J_{k_w} \in \mathcal{J}$ такое, что $w < r$ и выполняются соотношения (4). В первом случае неравенство $p_{k_v} < p_{k_r}$ выполняется для каждой допустимой длительности $p_{k_r} \in \left[p_{k_r}^L, p_{k_r}^U\right]$ обслуживания требования J_{k_r} и для каждой допустимой длительности $p_{k_v} \in \left[p_{k_v}^L, p_{k_v}^U\right]$ обслуживания требования J_{k_v}. Во втором случае неравенство $p_{k_w} > p_{k_r}$ выполняется для каждой допустимой длительности $p_{k_r} \in \left[p_{k_r}^L, p_{k_r}^U\right]$ обслуживания требования J_{k_r} и для каждой допустимой длительности $p_{k_w} \in \left[p_{k_w}^L, p_{k_w}^U\right]$ обслуживания требования J_{k_w}.

В силу теоремы 1 в обоих случаях перестановка π_k не может быть оптимальной для задачи $1|p|\sum C_i$ при каком-либо сценарии $p \in T$. Следовательно, согласно определению 2 область оптимальности для перестановки $\pi_k = (J_{k_1}, \ldots, J_{k_n}) \in \Pi$ является пустым множеством, $\mathcal{OR}(\pi_k, T) = \emptyset$. Достаточность доказана.

Необходимость. Докажем необходимость методом от противного. Пусть равенство $\mathcal{OR}(\pi_k, T) = \emptyset$ выполняется. Однако предположим, что не существует ни одного требования $J_{k_r} \in \mathcal{J}$, $r \in \{1, \ldots, n\}$, в перестановке $\pi_k = (J_{k_1}, \ldots, J_{k_n}) \in \Pi$, которое не имеет условной оптимальности и не имеет отрезка оптимальности.

Согласно определению 2 равенство $\mathcal{OR}(\pi_k, T) = \emptyset$ означает, что не существует ни одного сценария $p \in T$ такого, что перестановка π_k является оптимальной для задачи $1|p|\sum C_i$ со сценарием $p \in T$. Тем не менее покажем, как построить сценарий $p^* \in T$, который содержится в области оптимальности для перестановки π_k.

Если для требования J_{k_i} существует отрезок оптимальности $[l_{k_i}^{opt}, u_{k_i}^{opt}]$, $l_{k_i}^{opt} \leqslant u_{k_i}^{opt}$, в перестановке π_k, то существует хотя бы одна точка $p_{k_i}^* \in \left[l_{k_i}^{opt}, u_{k_i}^{opt}\right]$. Выберем такое значение $p_{k_i}^*$ в качестве длительности обслуживания требования J_{k_i}.

Если же не существует отрезка оптимальности для требования J_{k_j} в перестановке π_k, то согласно предположению существует отрезок условной оптимальности $\left[l_{k_j}^{copt}, u_{k_j}^{copt}\right]$ для требования J_{k_j}. Выберем значение $l_{k_j}^{copt}$ в качестве длительности обслуживания требования J_{k_j}: $p_{k_j}^* = l_{k_j}^{copt}$.

При таком выборе длительностей $p_{k_j}^*$ обслуживания всех требований $J_{k_j} \in \{J_{k_1}, \ldots, J_{k_n}\} = \mathcal{J}$ получаем сценарий $p^* = (p_{k_1}^*, \ldots, p_{k_n}^*) \in T$. Из равенств (6), (7) и леммы 2 с равенством (9) следует, что перестановка π_k является оптимальной для задачи $1|p^*|\sum C_i$. Следовательно, имеет место включение $p^* \in \mathcal{OR}(\pi_k, T)$, которое противоречит предполагаемому равенству $\mathcal{OR}(\pi_k, T) = \emptyset$. Полученное противоречие завершает доказательство теоремы 3.

Из теоремы 3 следует, что если $\mathcal{OR}(\pi_k, T) \neq \emptyset$, то в перестановке $\pi_k = (J_{k_1}, \ldots, J_{k_n}) \in \Pi$ для каждого требования $J_{k_r} \in \mathcal{J}$, $r \in \{1, \ldots, n\}$, существует хотя бы один отрезок оптимальности или отрезок условной оптимальности. Поэтому размерность непустой области оптимальности $\mathcal{OR}(\pi_k, T)$ равна $n = |\mathcal{J}|$. Поскольку из теоремы 3 следует и обратное утверждение, то получаем

Следствие 1. Размерность области оптимальности $\mathcal{OR}(\pi_k, T)$ равна $n = |\mathcal{J}|$ тогда и только тогда, когда $\mathcal{OR}(\pi_k, T) \neq \emptyset$.

На рис. 1 для отрезка неоптимальности $[l_4^{non}, u_4^{non}] = [7, 9]$ требования J_4 в перестановке $\pi_1 = (J_1, \ldots, J_{18})$ выполняются равенства $[l_4^{non}, u_4^{non}] = [7, 9] = [p_4^L, p_4^U]$, т.е. требование J_4 не имеет условной оптимальности и одновременно не имеет отрезка оптимальности в перестановке π_1. Из теоремы 3 следует, что область оптимальности для перестановки π_1 является пустым множеством, $\mathcal{OR}(\pi_1, T) = \emptyset$.

4. Построение области оптимальности и вычисление ее объема

В результате использования теоремы 3 разработан алгоритм 1 сложности $O(n)$ для проверки выполнения равенства $\mathcal{OR}(\pi_k, T) = \emptyset$ для фиксированной перестановки $\pi_k \in \Pi$.

Если алгоритм 1 устанавливает, что для перестановки π_k справедливо соотношение $\mathcal{OR}(\pi_k, T) \neq \emptyset$, то в соответствии с теоремой 2 алгоритм 1 строит по формулам (8) сокращенные отрезки $\left[\widehat{p}_i^L, \widehat{p}_i^U\right]$ допустимых длительностей обслуживания требований $J_i \in \mathcal{J}$. Тем самым определяются исходные данные для задачи $1|\widehat{p}_i^L \leqslant p_i \leqslant \widehat{p}_i^U| \sum C_i$ с множеством $\widehat{T}$ допустимых сценариев.

Алгоритм 1.

ВХОД: Отрезки $[p_i^L, p_i^U]$ длительностей обслуживания требований $J_i \in \mathcal{J}$; перестановка $\pi_k = (J_{k_1}, \ldots, J_{k_n}) \in \Pi$ обслуживания требований $\mathcal{J}$.

ВЫХОД: Отрезки $[\widehat{p}_i^L, \widehat{p}_i^U]$ для требований $J_i \in \mathcal{J}$, если установлено, что $\mathcal{OR}(\pi_k, T) \neq \emptyset$.

Шаг 1: $\widehat{p}_{k_1}^L = p_{k_1}^L$, $t_L = p_{k_1}^L$, $r = 2$;

Шаг 2: **IF** $p_{k_r}^U \geqslant t_L$ **THEN GOTO** шаг 3 **ELSE GOTO** шаг 5;

Шаг 3: **IF** $p_{k_r}^L > t_L$ **THEN** $t_L = p_{k_r}^L$, $\widehat{p}_{k_r}^L = t_L$, $r := r + 1$;
ELSE $\widehat{p}_{k_r}^L = t_L$, $r := r + 1$;
IF $r \leqslant n$ **THEN GOTO** шаг 2 **ELSE** $\widehat{p}_{k_n}^U = p_{k_n}^U$, $t_U = p_{k_n}^U$;

Шаг 4: **FOR** $r = n - 1$ **to** 1 **STEP** -1 **DO**
IF $p_{k_r}^U < t_U$ **THEN** $t_U = p_{k_r}^U$, $\widehat{p}_{k_r}^U = t_U$ **ELSE** $\widehat{p}_{k_r}^U = t_U$;
END FOR STOP.

Шаг 5: $\mathcal{OR}(\pi_k, T) = \emptyset$ **STOP**.

На шагах 1, 2 и 5 алгоритма 1 проверяется равенство $\mathcal{OR}(\pi_k, T) = \emptyset$. На шагах 2–4 строятся сокращенные отрезки допустимых длительностей обслу-

живания требований множества $\mathcal{J}$ для задачи $1|\widehat{p}_i^L \leqslant p_i \leqslant \widehat{p}_i^U|\sum C_i$. Согласно теореме 2 область оптимальности для перестановки $\pi_k \in \Pi$ обслуживания требований для задачи $1|p_i^L \leqslant p_i \leqslant p_i^U|\sum C_i$ совпадает с областью оптимальности для той же перестановки π_k обслуживания требований для задачи $1|\widehat{p}_i^L \leqslant p_i \leqslant \widehat{p}_i^U|\sum C_i$ с множеством $\widehat{T}$ сокращенных сценариев. Нетрудно убедиться в том, что для реализации алгоритма 1 требуется выполнить $O(n)$ элементарных операций.

4.1. Область оптимальности для перестановки $\pi_k \in \Pi$ в частных случаях

Построим область оптимальности $\mathcal{OR}(\pi_k, T)$ для двух частных случаев перестановки $\pi_k = (J_{k_1}, \ldots, J_{k_n}) \in \Pi$ и вычислим объем $Vol(\pi_k, T)$ области оптимальности $\mathcal{OR}(\pi_k, T)$. В первом случае область оптимальности $\mathcal{OR}(\pi_k, T)$ определяется только отрезками оптимальности всех требований $J_{k_r} \in \mathcal{J}$ (лемма 3). Во втором случае область оптимальности определяется только отрезками условной оптимальности всех требований $J_{k_r} \in \mathcal{J}$ (лемма 4).

Лемма 3. Если $\mathcal{OR}(\pi_k, T) \neq \emptyset$ и каждое требование $J_{k_r} \in \mathcal{J}$ не имеет условной оптимальности в перестановке $\pi_k = (J_{k_1}, \ldots, J_{k_n}) \in \Pi$, то область оптимальности для перестановки π_k совпадает с параллелепипедом оптимальности для той же перестановки:

$$\mathcal{OR}(\pi_k, T) = \times_{r=1}^{n} \left[l_{k_r}^{opt}, u_{k_r}^{opt} \right] = \mathcal{OB}(\pi_k, T). \tag{10}$$

Объем такой области оптимальности $\mathcal{OR}(\pi_k, T)$ равен

$$Vol(\pi_k, T) = \prod_{J_{k_r} \in \left\{ \mathcal{J} : l_{k_r}^{opt} < u_{k_r}^{opt} \right\}} (u_{k_r}^{opt} - l_{k_r}^{opt}). \tag{11}$$

Доказательство. В силу теоремы 2 вместо задачи $1|p_i^L \leqslant p_i \leqslant p_i^U|\sum C_i$ будем рассматривать задачу $1|\widehat{p}_i^L \leqslant p_i \leqslant \widehat{p}_i^U|\sum C_i$. Поскольку $\mathcal{OR}(\pi_k, T) \neq \emptyset$, то по теореме 3 не существует ни одного требования $J_{k_r} \in \mathcal{J}$, которое не имеет условной оптимальности и одновременно не имеет отрезка оптимальности в перестановке π_k.

Поскольку каждое требование $J_{k_r} \in \mathcal{J}$ не имеет условной оптимальности в перестановке π_k и для него существует не более одного отрезка оптимальности (замечание 1), то для каждого требования $J_{k_r} \in \mathcal{J}$ выполняется равенство $[\widehat{p}_{k_r}^L, \widehat{p}_{k_r}^U] = \left[l_{k_r}^{opt}, u_{k_r}^{opt} \right]$. Следовательно, в соответствии с определениями 1 и 2 область оптимальности $\mathcal{OR}(\pi_k, T)$ для перестановки π_k совпадает с параллелепипедом оптимальности для той же перестановки π_k и представляет собой n-мерный параллелепипед $\times_{r=1}^{n} \left[l_{k_r}^{opt}, u_{k_r}^{opt} \right] = \mathcal{OB}(\pi_k, T)$, объем которого равен $\prod_{J_{k_r} \in \{\mathcal{J} : l_{k_r}^{opt} < u_{k_r}^{opt}\}} (u_{k_r}^{opt} - l_{k_r}^{opt})$. Лемма 3 доказана.

На рис. 2 представлена перестановка $\pi_2 = (J_1, \ldots, J_{10})$ требований множества $\mathcal{J} = \{J_1, \ldots, J_{10}\}$ для примера 2 задачи $1|p_i^L \leqslant p_i \leqslant p_i^U|\sum C_i$. Поскольку

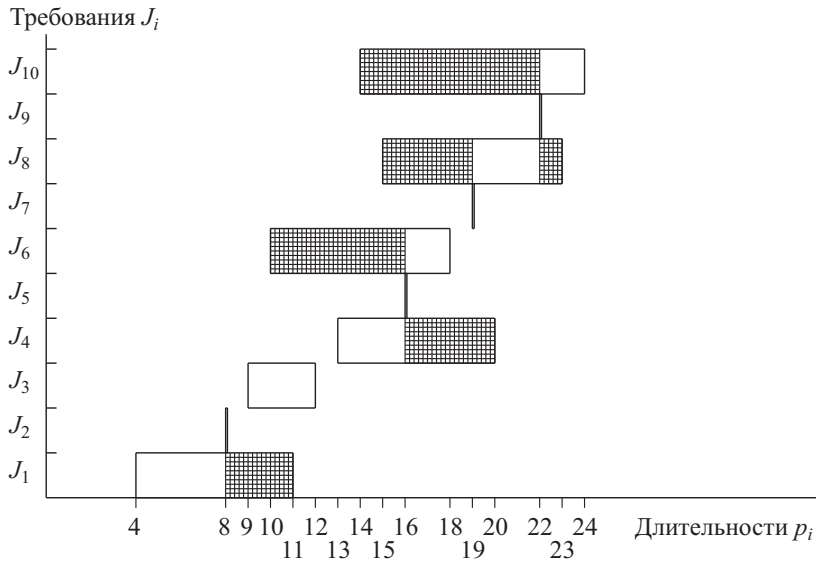


Рис. 2. Отрезки оптимальности и неоптимальности (заштрихованы) для требований $J_i \in \mathcal{J}$ в перестановке $\pi_2 = (J_1, \ldots, J_{10})$ для примера 2 задачи $1|p_i^L \leqslant \leqslant p_i \leqslant p_i^U| \sum C_i$.

перестановка π_2 удовлетворяет условию леммы 3, то область оптимальности $\mathcal{OR}(\pi_2, T)$ представляет собой следующий 10-мерный параллелепипед:

$$\mathcal{OR}(\pi_2, T) = \mathcal{OB}(\pi_2, T) = \left[l_1^{opt}, u_1^{opt}\right] \times \ldots \times \left[l_{10}^{opt}, u_{10}^{opt}\right] =$$
$$= [4, 8] \times [8, 8] \times [9, 12] \times [13, 16] \times [16, 16] \times [16, 18] \times$$
$$\times [19, 19] \times [19, 22] \times [22, 22] \times [22, 24],$$

объем которого вычисляется по формуле (11), а именно:

$$Vol(\pi_2, T) = \prod_{J_r \in \{\mathcal{J} : l_r^{opt} < u_r^{opt}\}} (u_r^{opt} - l_r^{opt}) =$$
$$= (8-4)(12-9)(16-13)(18-16)(22-19)(24-22) = 432.$$

В соответствии с теоремой 2 вместо задачи $1|p_i^L \leqslant p_i \leqslant p_i^U| \sum C_i$ будем рассматривать задачу $1|\widehat{p}_i^L \leqslant p_i \leqslant \widehat{p}_i^U| \sum C_i$ с множеством $\widehat{T}$ сокращенных сценариев.

Определение 3. Секцией перестановки $\pi_k \in \Pi$ *называется максимальная по включению перестановка* $s_v^{\pi_k} = (J_{k_v}, \ldots, J_{k_{v+m_v}})$, $1 \leqslant v \leqslant v + m_v \leqslant n$, *такая что для любого числа* $d \in \left(\widehat{p}_{k_v}^L, \widehat{p}_{k_{v+m_v}}^U\right)$ *существует требование* $J_{k_{v+i}}$, $i \in \{0, \ldots, m_v\}$, *для которого* $d \in \left(\widehat{p}_{k_{v+i}}^L, \widehat{p}_{k_{v+i}}^U\right)$. *Отрезок* $\left[\widehat{p}_{k_v}^L, \widehat{p}_{k_{v+m_v}}^U\right]$ *называется охватом секции* $s_v^{\pi_k}$.

Отметим, что множество $S(\pi_k) = \{s_v^{\pi_k}, \ldots, s_w^{\pi_k}\}$, $1 \leqslant v < \ldots < w \leqslant n$, всех секций каждой перестановки $\pi_k \in \Pi$ определено однозначно.

Замечание 2. Из определения 3 следует, что каждое требование $J_{k_i} \in \mathcal{J}$ либо содержится в единственной секции перестановки π_k, либо не содержится ни в одной секции перестановки π_k. Если существует хотя бы одно требование $J_{k_i} \in \mathcal{J}$, которое не содержится ни в одной секции перестановки π_k, то по теореме 3 получаем равенство $\mathcal{OR}(\pi_k, T) = \emptyset$.

Из замечания 2 и доказательства теоремы 3 получаем следующее утверждение.

Следствие 2. Для того чтобы область оптимальности перестановки $\pi_k = (J_{k_1}, \ldots, J_{k_n}) \in \Pi$ *не была пустым множеством,* $\mathcal{OR}(\pi_k, T) \neq \emptyset$*, необходимо и достаточно, чтобы выполнялось равенство* $\pi_k = (s_1^{\pi_k}, \ldots, s_w^{\pi_k})$.

Для перестановки $\pi_2 = (J_1, \ldots, J_{10})$ в примере 2, представленном на рис. 2, каждая секция состоит из единственного требования $s_1^{\pi_2} = (J_1)$, …, $s_{10}^{\pi_2} = (J_{10})$ и выполняется равенство $\pi_2 = (s_1^{\pi_2}, \ldots, s_{10}^{\pi_2})$. Секцию, состоящую из единственного требования, будем называть тривиальной.

Для доказательства леммы 4 разобьем охват $\left[\widehat{p}_{k_j}^L, \widehat{p}_{k_{j+m_j}}^U\right]$ каждой секции $s_j^{\pi_k} \in S(\pi_k)$ на максимальные (по включению) подынтервалы условной оптимальности

$$\left[\widehat{p}_{k_j}^L, \widehat{p}_{k_{j+m_j}}^U\right] = \left[l_1^j\left(s_j^{\pi_k}\right), u_1^j\left(s_j^{\pi_k}\right)\right) \bigcup \ldots \bigcup \left[l_i^j\left(s_j^{\pi_k}\right), u_i^j\left(s_j^{\pi_k}\right)\right) \bigcup \ldots \bigcup \left[l_{n(j)}^j\left(s_j^{\pi_k}\right), u_{n(j)}^j\left(s_j^{\pi_k}\right)\right], \tag{12}$$

которые отличаются один от другого тем, что для разных подмножеств $\mathcal{J}_i^j = \left\{J_{k_i}, \ldots, J_{k_{|\mathcal{J}_i^j|}}\right\}$ множества $\left\{J_{k_j}, \ldots, J_{k_{j+m_j}}\right\}$, $j \leqslant i \leqslant j + m_j$, выполняются включения $\left[l_i^j\left(s_j^{\pi_k}\right), u_i^j\left(s_j^{\pi_k}\right)\right] \subseteq \left[\widehat{p}_{k_r}^L, \widehat{p}_{k_r}^U\right]$ для всех требований $J_{k_r} \in \mathcal{J}_i^j$. Пусть $\widehat{J}_i^j = \left(J_{k_i}, \ldots, J_{k_{|\mathcal{J}_i^j|}}\right)$ обозначает перестановку требований множества $\mathcal{J}_i^j = \left\{J_{k_i}, \ldots, J_{k_{|\mathcal{J}_i^j|}}\right\}$.

Для иллюстрации введенных обозначений рассмотрим разбиения охватов секций $s_1^{\pi_2}$ и $s_8^{\pi_2}$ перестановки $\pi_2 = (J_1, \ldots, J_{10})$ в примере 3, представленном на рис. 3. Секция $s_1^{\pi_2}$ состоит из семи упорядоченных требований $s_1^{\pi_2} = (J_1, \ldots, J_7)$ и имеет охват $[\widehat{p}_1^L, \widehat{p}_7^U] = [4, 20]$. Получаем следующее разбиение (12): $[4, 20] = [4, 8) \cup [8, 9) \cup [9, 13) \cup [13, 16) \cup [16, 17) \cup [17, 18) \cup [18, 20]$ охвата $[\widehat{p}_1^L, \widehat{p}_7^U]$ на подынтервалы условной оптимальности. Здесь

$$\left[l_1^1(J_1, J_2), u_1^1(J_1, J_2)\right) = [4, 8); \quad \left[l_2^1(J_1, \ldots, J_4), u_2^1(J_1, \ldots, J_4)\right) = [8, 9);$$
$$\left[l_3^1(J_2, J_3, J_4), u_3^1(J_2, J_3, J_4)\right) = [9, 13); \quad \left[l_4^1(J_3, J_4), u_4^1(J_3, J_4)\right) = [13, 16);$$
$$\left[l_5^1(J_3, \ldots, J_6), u_5^1(J_3, \ldots, J_6)\right) = [16, 17);$$
$$\left[l_6^1(J_3, \ldots, J_7), u_6^1(J_3, \ldots, J_7)\right) = [17, 18];$$
$$\left[l_7^1(J_4, \ldots, J_7), u_7^1(J_4, \ldots, J_7)\right] = [18, 20].$$

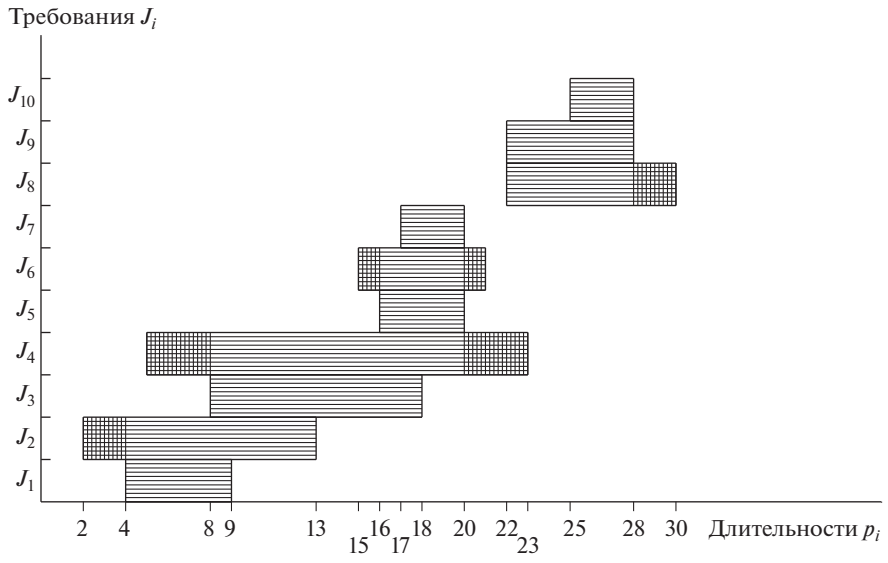


Рис. 3. Отрезки условной оптимальности (заштрихованы) и отрезки неоптимальности (заштрихованы дважды) для требований $J_i \in \mathcal{J}$ в перестановке $\pi_2 = (J_1, \ldots, J_{10})$ для примера 3 задачи $1|p_i^L \leqslant p_i \leqslant p_i^U| \sum C_i$.

Отметим равенства $\widehat{J}_1^1 = (J_1, J_2)$, $\widehat{J}_2^1 = (J_1, \ldots, J_4)$, $\widehat{J}_3^1 = (J_2, J_3, J_4)$, $\widehat{J}_4^1 = (J_3, J_4)$, $\widehat{J}_5^1 = (J_3, \ldots, J_6)$, $\widehat{J}_6^1 = (J_3, \ldots, J_7)$, $\widehat{J}_7^1 = (J_4, \ldots, J_7)$. Секция $s_8^{\pi_2}$ состоит из трех требований $s_8^{\pi_2} = (J_8, J_9, J_{10})$ и имеет охват $[\widehat{p}_8^L, \widehat{p}_{10}^U] = [22, 28]$. Получаем разбиение $[22,28] = [22,25) \cup [25,28]$ охвата $[22,28]$ на подынтервалы условной оптимальности $[l_1^2(J_8, J_9), u_1^2(J_8, J_9))$ и $[l_2^2(J_8, J_9, J_{10}), u_2^2(J_8, J_9, J_{10})]$. Отметим равенства $\widehat{J}_1^2 = (J_8, J_9)$ и $\widehat{J}_2^2 = (J_8, J_9, J_{10})$.

Поскольку любая секция $s_j^{\pi_k} \in S(\pi_k)$ перестановки π_k и любое упорядоченное множество $\widehat{J}_i^j$ требований множества $\mathcal{J}_i^j \subseteq \mathcal{J}$ сами являются перестановками соответствующего подмножества требований множества $\mathcal{J}$, то для них можно рассматривать области оптимальности для всех тех требований, из которых состоят эти перестановки. Для обозначения таких областей оптимальности будем использовать те же обозначения $\mathcal{OR}(s_j^{\pi_k}, T)$ и $\mathcal{OR}(\widehat{J}_i^j, T)$, что и для области оптимальности $\mathcal{OR}(\pi_k, T)$ для перестановки $\pi_k \in \Pi$ всего множества требований $\mathcal{J}$. Для обозначения объемов областей оптимальности $\mathcal{OR}(s_j^{\pi_k}, T)$ и $\mathcal{OR}(\widehat{J}_i^j, T)$ будем использовать обозначения $Vol(s_j^{\pi_k}, T)$ и $Vol(\widehat{J}_i^j, T)$ соответственно.

Доказательство следующей леммы 4 приведено в Приложении. Там же дано определение d-мерной пирамиды оптимальности $Pir^{opt}\widehat{J}_i^j$, представляющей собой область оптимальности для перестановки требований $\mathcal{J}_i^j \subseteq \mathcal{J}$, т.е. показано, что выполняются равенства $d = |\mathcal{J}_i^j|$ и $\mathcal{OR}(\widehat{J}_i^j, T) = Pir^{opt}\widehat{J}_i^j$.

Лемма 4. Если $\mathcal{OR}(\pi_k, T) \neq \emptyset$ и каждое требование $J_{k_r} \in \mathcal{J}$ не имеет отрезка оптимальности в перестановке $\pi_k = (J_{k_1}, \ldots, J_{k_n}) \in \Pi$, то область оптимальности $\mathcal{OR}(\pi_k, T)$ для перестановки π_k представляет собой декар-

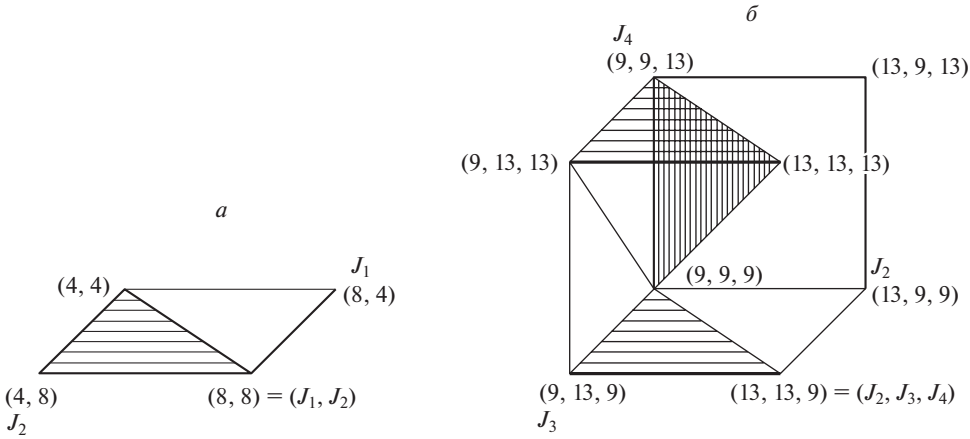


Рис. 4. (a) – Треугольник оптимальности $Pir^{opt}(J_1, J_2)$ с основанием $[(4,8),(8,8)]$ и высотой $[(4,8),(4,4)]$ для подынтервала условной оптимальности $[4,8)$ для перестановки $\pi_2 = (J_1, \ldots, J_{10})$ в примере 3. (b) – Пирамида оптимальности $Pir^{opt}(J_2, J_3, J_4)$ с основанием $[(9,9,13),(9,13,13),(13,13,13)]$ и высотой $[(9,9,13),(9,9,9)]$ для подынтервала условной оптимальности $[9,13)$ для перестановки π_2 в примере 3.

тово произведение $|S(\pi_k)|$ *областей оптимальности* $\mathcal{OR}(s_j^{\pi_k}, T)$ *всех секций* $S(\pi_k)$:

$$\mathcal{OR}(\pi_k, T) = \mathcal{OR}(s_1^{\pi_k}, T) \times \ldots \times \mathcal{OR}\left(s_j^{\pi_k}, T\right) \times \ldots \times \mathcal{OR}\left(s_{|S(\pi_k)|}^{\pi_k}, T\right), \tag{13}$$

где $\mathcal{OR}(s_j^{\pi_k}, T)$ *определяется как следующее объединение* d*-мерных пирамид оптимальности* $Pir^{opt}\widehat{J}_i^j$ *в пространстве* R^n, $d \in \left\{|\mathcal{J}_i^j|, \ldots, |\mathcal{J}_{n(j)}^j|\right\}$:

$$\mathcal{OR}(s_j^{\pi_k}, T) = \bigcup_{i=1}^{n(j)} \mathcal{OR}(\widehat{J}_i^j, T) = \bigcup_{i=1}^{n(j)} Pir^{opt}\widehat{J}_i^j. \tag{14}$$

Объем области оптимальности для перестановки π_k *определяется равенством*

$$Vol\,(\pi_k, T) = \prod_{j=1}^{|S(\pi_k)|} \sum_{i=1}^{n(j)} \frac{\left(u_i^j\left(s_j^{\pi_k}\right) - l_i^j\left(s_j^{\pi_k}\right)\right)^{|\mathcal{J}_i^j|}}{|\mathcal{J}_i^j|!}. \tag{15}$$

На рис. 3 представлена перестановка $\pi_2 = (J_1, \ldots, J_{10})$ требований множества $\mathcal{J} = \{J_1, \ldots, J_{10}\}$ для примера 3 задачи $1|p_i^L \leqslant p_i \leqslant p_i^U| \sum C_i$. По теореме 3 область оптимальности для перестановки π_2 не является пустым множеством, и поскольку перестановка π_2 удовлетворяет условию леммы 4, то объем

области оптимальности $\mathcal{OR}(\pi_2, T)$ вычисляется по формуле (15), а именно:

$$Vol(\pi_2, T) = \prod_{j=1}^{2} \sum_{i=1}^{n(j)} \frac{\left(u_i^j\left(s_j^{\pi_k}\right) - l_i^j\left(s_j^{\pi_k}\right)\right)^{|\mathcal{J}_i^j|}}{|\mathcal{J}_i^j|!} =$$

$$= \left[\frac{(8-4)^2}{2!} + \frac{(9-8)^4}{4!} + \frac{(13-9)^3}{3!} + \frac{(16-13)^2}{2!} + \frac{(17-16)^4}{4!} + \right.$$

$$\left. + \frac{(18-17)^5}{5!} + \frac{(20-18)^4}{4!}\right]\left[\frac{(25-22)^2}{2!} + \frac{(28-25)^3}{3!}\right] =$$

$$= \left[\frac{16}{2} + \frac{1}{24} + \frac{64}{6} + \frac{9}{2} + \frac{1}{24} + \frac{1}{120} + \frac{16}{24}\right]\left[\frac{9}{2} + \frac{27}{6}\right] = 210\frac{33}{40}.$$

На рис. 4,*a* представлен указанный в равенстве (14) треугольник $Pir^{opt}\widehat{J}_1^1 = Pir^{opt}(J_1, J_2)$ для подынтервала условной оптимальности $[4, 8)$, принадлежащий области оптимальности $\mathcal{OR}(\pi_2, T)$ для перестановки $\pi_2 = (J_1, \ldots, J_{10})$ в примере 3, а на рис. 4,*b* представлена указанная в (14) 3-мерная пирамида $Pir^{opt}\widehat{J}_3^1 = Pir^{opt}(J_2, J_3, J_4)$ для подынтервала условной оптимальности $[9, 13)$, принадлежащая той же области оптимальности $\mathcal{OR}(\pi_2, T)$.

4.2. Объем области оптимальности для перестановки $\pi_k \in \Pi$ в общем случае

Пусть $S^*(\pi_k)$ обозначает подмножество тривиальных секций множества $S(\pi_k)$.

Отметим, что перестановка $\pi_2 = (J_1, \ldots, J_{10})$ в примере 2 удовлетворяет условию леммы 3 и все секции множества $S(\pi_2)$ являются тривиальными: $S(\pi_2) = S^*(\pi_2)$.

Перестановка $\pi_2 = (J_1, \ldots, J_{10})$ в примере 3 удовлетворяет условию леммы 4, и множество $S^*(\pi_2)$ ее тривиальных секций – пустое множество: $S^*(\pi_2) = \emptyset$.

Теорема 4. Если $\mathcal{OR}(\pi_k, T) \neq \emptyset$, *то область оптимальности* $\mathcal{OR}(\pi_k, T)$ *представляет собой декартово произведение* (13) *областей оптимальности секций* $S(\pi_k)$ *такое, что*

$$\mathcal{OR}\left(s_j^{\pi_k}, T\right) = \mathcal{OB}\left(s_j^{\pi_k}, T\right) = \left[l_{k_r}^{opt}, u_{k_r}^{opt}\right]$$

для каждой тривиальной секции $s_j^{\pi_k} = (J_{k_r})$ *и*

$$\mathcal{OR}(s_j^{\pi_k}, T) = \bigcup_{i=1}^{n(j)} \mathcal{OR}(\widehat{J}_i^j, T) = \bigcup_{i=1}^{n(j)} Pir^{opt}\widehat{J}_i^j$$

для каждой нетривиальной секции $s_j^{\pi_k} \in S(\pi_k) \setminus S^*(\pi_k)$. *Объем области оптимальности равен*

(16) $$Vol(\pi_k, T) =$$

$$= \prod_{(J_{k_r})\in\left\{S^*(\pi_k):\, l_{k_r}^{opt}<u_{k_r}^{opt}\right\}} \left(u_{k_r}^{opt}-l_{k_r}^{opt}\right) \prod_{s_j^{\pi_k}\in S(\pi_k)\setminus S^*(\pi_k)} \sum_{i=1}^{n(j)} \frac{\left(u_i^j\left(s_j^{\pi_k}\right)-l_i^j\left(s_j^{\pi_k}\right)\right)^{|\mathcal{J}_i^j|}}{|\mathcal{J}_i^j|!}.$$

Доказательство. В силу теоремы 2 вместо задачи $1|p_i^L \leqslant p_i \leqslant p_i^U|\sum C_i$ будем рассматривать задачу $1|\widehat{p}_i^L \leqslant p_i \leqslant \widehat{p}_i^U|\sum C_i$. Поскольку $\mathcal{OR}(\pi_k, T) \neq \emptyset$, то размерность области оптимальности $\mathcal{OR}(\pi_k, T)$ равна n (следствие 1). Из теоремы 3 следует, что не существует ни одного требования $J_{k_r}\in\mathcal{J}$, которое не имеет условной оптимальности и одновременно не имеет отрезка оптимальности в перестановке π_k.

Поскольку $\mathcal{OR}(\pi_k, T) \neq \emptyset$, то из замечания 2 следует, что каждое требование $J_{k_i}\in\mathcal{J}$ содержится в единственной секции перестановки π_k. При этом согласно следствию 2 справедливо равенство $\pi_k = (s_1^{\pi_k}, \ldots, s_w^{\pi_k})$.

На основе перечисленных свойств перестановки π_k покажем, как доказать равенство (13) и равенство (16) в результате последовательного применения леммы 3 или леммы 4 в частных случаях, когда очередная рассматриваемая перестановка состоит из единственной секции $s_v^{\pi_k} \in S(\pi_k)$ перестановки π_k.

Вначале рассмотрим первую секцию $s_1^{\pi_k} = (J_{k_1}, \ldots, J_{k_{1+m_1}})$ перестановки π_k. Если секция $s_1^{\pi_k}$ является тривиальной, т.е. $s_1^{\pi_k} = (J_{k_1})$, то по лемме 3 получаем первый сомножитель $[l_{k_1}^{opt}, u_{k_1}^{opt}] = \mathcal{OR}(s_1^{\pi_k}, T)$ в искомом декартовом произведении (13) и первый сомножитель $(u_{k_1}^{opt} - l_{k_1}^{opt})$ в первом произведении равенства (16) при условии, что имеет место строгое неравенство $l_{k_1}^{opt} < u_{k_1}^{opt}$ (если $l_{k_1}^{opt} = u_{k_1}^{opt}$, то сомножитель $(u_{k_1}^{opt} - l_{k_1}^{opt}) = 0$ в равенство (16) не добавляется в соответствии с леммой 3).

Если же секция $s_1^{\pi_k}$ не является тривиальной, т.е.

$$s_1^{\pi_k} \in S(\pi_k)\setminus S^*(\pi_k),$$

то по лемме 4 получаем первый сомножитель

$$\mathcal{OR}(s_1^{\pi_k}, T) = \bigcup_{i=1}^{n(1)} \mathcal{OR}(\widehat{J}_i^1, T) = \bigcup_{i=1}^{n(1)} Pir^{opt}\widehat{J}_i^1$$

в декартовом произведении (13) и первый сомножитель

$$\sum_{i=1}^{n(1)} \frac{(u_i^1(s_1^{\pi_k}) - l_i^1(s_1^{\pi_k}))^{|\mathcal{J}_i^1|}}{|\mathcal{J}_i^1|!}$$

во втором произведении равенства (16). Здесь необходимо отметить, что в разбиение (12) секции $s_1^{\pi_k}$ на подынтервалы условной оптимальности могут входить и подынтервалы $[l_i^1(s_1^{\pi_k}), u_i^1(s_1^{\pi_k}))$, для которых выполняется равенство $|\mathcal{J}_i^1| = 1$ (такая возможность в условии леммы 4 не предусмотрена). Покажем, однако, что равенство

$$\mathcal{OR}(\widehat{J}_i^j, T) = \frac{(u_i^j(s_j^{\pi_k}) - l_i^j(s_j^{\pi_k}))^{|\mathcal{J}_i^j|}}{|\mathcal{J}_i^j|!},$$

содержащееся в (16), справедливо и для случая $|\mathcal{J}_i^j| = 1$. Действительно, если $|\mathcal{J}_i^j| = 1$, то

$$\frac{\left(u_i^j\left(s_j^{\pi_k}\right) - l_i^j\left(s_j^{\pi_k}\right)\right)^{|\mathcal{J}_i^j|}}{|\mathcal{J}_i^j|!} = \frac{\left(u_i^j\left(s_j^{\pi_k}\right) - l_i^j\left(s_j^{\pi_k}\right)\right)^{1}}{1!} = \left(u_i^j\left(s_j^{\pi_k}\right) - l_i^j\left(s_j^{\pi_k}\right)\right),$$

что и требуется для выполнения равенства (16).

Аналогично рассмотрим вторую секцию $s_2^{\pi_k} = (J_{k_{m_1+1}}, \ldots, J_{k_{m_1+1+m_2}})$ перестановки π_k. Применяя лемму 3, если секция $s_2^{\pi_k}$ тривиальная, или лемму 4, если секция $s_2^{\pi_k}$ не является тривиальной, дополним уже построенную часть декартова произведения (13) вторым сомножителем и дополним вторым сомножителем одно (соответствующее) произведение из двух произведений равенства (16).

Продолжив описанный процесс вплоть до рассмотрения последней секции $s_w^{\pi_k}$ перестановки π_k и добавления соответствующих сомножителей, получим оба равенства (13) и (16). Теорема 4 доказана.

Следующий алгоритм вычисления объема $Vol(\pi_k, T)$ области оптимальности $\mathcal{OR}(\pi_k, T) \neq \emptyset$ для перестановки $\pi_k \in \Pi$ основан на теоремах 2, 3 и 4.

Алгоритм 2.

ВХОД: Перестановка $\pi_k = (J_{k_1}, \ldots, J_{k_n}) \in \Pi$, для которой $\mathcal{OR}(\pi_k, T) \neq \emptyset$; отрезки $[\widehat{p}_i^L, \widehat{p}_i^U]$ длительностей обслуживания требований $J_i \in \mathcal{J}$.

ВЫХОД: Объем области оптимальности $\mathcal{OR}(\pi_k, T)$ для перестановки π_k.

Шаг 1: Определить множество секций
$S(\pi_k) = \left\{s_1^{\pi_k}, s_{1+m_1}^{\pi_k}, \ldots, s_{j+m_j}^{\pi_k}, \ldots, s_w^{\pi_k}\right\}$;

Шаг 2: $j = 1,\ Vol = 1,\ Vol^* = 1,\ Sum = 0$;

Шаг 3: **IF** секция $s_j^{\pi_k} = (J_{k_j}, \ldots, J_{k_{j+m_j}})$ тривиальная $s_j^{\pi_k} = (J_{k_j})$ **THEN GOTO** шаг 7;

Шаг 4: **ELSE** для секции $s_j^{\pi_k} = (J_{k_j}, \ldots, J_{k_{j+m_j}})$ построить разбиение (12) охвата $\left[\widehat{p}_{k_j}^L, \widehat{p}_{k_j+m_j}^U\right]$ на подынтервалы условной оптимальности:

$$\left[l_1^j\left(s_j^{\pi_k}\right), u_1^j\left(s_j^{\pi_k}\right)\right) \bigcup \ldots \bigcup \left[l_i^j\left(s_j^{\pi_k}\right), u_i^j\left(s_j^{\pi_k}\right)\right) \bigcup \ldots$$
$$\ldots \bigcup \left[l_{n(j)}^j\left(s_j^{\pi_k}\right), u_{n(j)}^j\left(s_j^{\pi_k}\right)\right];$$

Шаг 5: **FOR** $i = 1$ **to** $n(j)$ **DO** вычислить $OS = \frac{\left(u_i^j\left(s_j^{\pi_k}\right) - l_i^j\left(s_j^{\pi_k}\right)\right)^{|\mathcal{J}_i^j|}}{|\mathcal{J}_i^j|!}$;
$Sum := Sum + OS$ **END FOR**

Шаг 6: $Vol := Vol \cdot Sum$, $j := j + m_j$ **IF** $j \leqslant w$ **THEN GOTO** шаг 3; **ELSE GOTO** шаг 10;

Шаг 7: $j := j + m_j$, $OS^* = u_{k_j}^{opt} - l_{k_j}^{opt}$ **IF** $u_{k_j}^{opt} > l_{k_j}^{opt}$ **THEN GOTO** шаг 9; **ELSE IF** $j \leqslant w$ **THEN GOTO** шаг 3;

Шаг 8: **ELSE GOTO** шаг 10;

Шаг 9: $Vol^* := Vol^* \cdot OS^*$ **IF** $j \leqslant w$ **THEN GOTO** шаг 3 **ELSE**

Шаг 10: $Vol(\pi_k, T) = Vol \cdot Vol^*$ **STOP**.

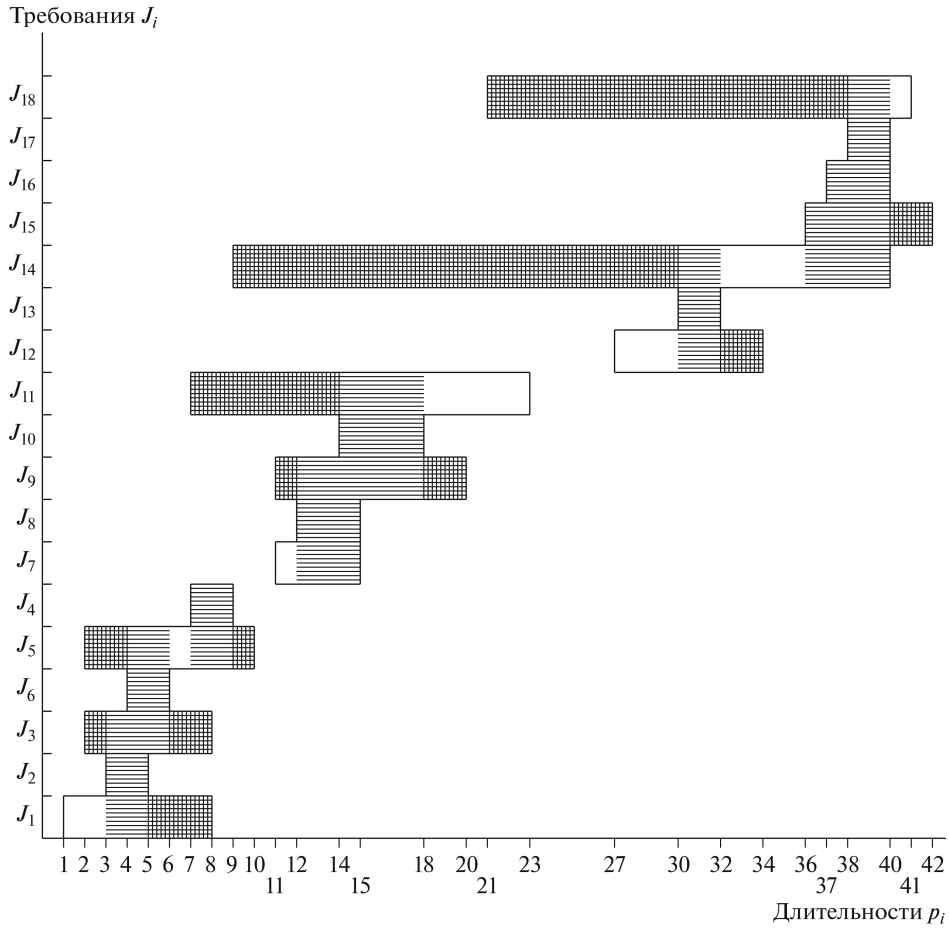


Рис. 5. Отрезки оптимальности, условной оптимальности (заштрихованы) и отрезки неоптимальности (заштрихованы дважды) для требований $J_i \in \mathcal{J} = \{J_1, \ldots, J_{18}\}$ в перестановке $\pi_3 = (J_1, \ldots, J_3, J_6, J_5, J_4, J_7, \ldots, J_{18})$ для примера 1.

Для реализации шага 1 требуется $O(n)$ операций. Выполнение шагов 3–6 требует $O(n^2)$ операций. Выполнение шагов 7–9 требует $O(n)$ операций. Всему алгоритму 2 требуется выполнить $O(n^2)$ операций для вычисления объема $Vol(\pi_k, T)$ области оптимальности $\mathcal{OR}(\pi_k, T)$ для фиксированной перестановки $\pi_k \in \Pi$.

На рис. 5 представлена перестановка $\pi_3 = (J_1, \ldots, J_3, J_6, J_5, J_4, J_7, \ldots, J_{18})$ требований множества $\mathcal{J} = \{J_1, \ldots, J_{18}\}$ для примера 3 задачи $1|p_i^L \leqslant p_i \leqslant p_i^U| \sum C_i$. По теореме 3 область оптимальности для перестановки π_3 не пустая, поэтому вычислим ее объем по формуле (16) из теоремы 4, учитывая равенство $S^*(\pi_3) = \emptyset$.

$$Vol(\pi_3, T) = \prod_{s_j^{\pi_3} \in S(\pi_3)} \sum_{i=1}^{n(j)} \frac{(u_i^j(s_j^{\pi_3}) - l_i^j(s_j^{\pi_3}))^{|\mathcal{J}_i^j|}}{|\mathcal{J}_i^j|!} =$$

$$= \left[\frac{(3-1)^1}{1!} + \frac{(4-3)^3}{3!} + \frac{(5-4)^5}{5!} + \frac{(6-5)^3}{3!} + \frac{(7-6)^1}{1!} + \frac{(9-7)^2}{2!}\right] \times$$
$$\times \left[\frac{(12-11)^1}{1!} + \frac{(14-12)^3}{3!} + \frac{(15-14)^5}{5!} + \frac{(18-15)^3}{3!} + \frac{(23-18)^1}{1!}\right] \times$$
$$\times \left[\frac{(30-27)^1}{1!} + \frac{(32-30)^3}{3!} + \frac{(36-32)^1}{1!} + \frac{(37-36)^2}{2!} + \frac{(38-37)^3}{3!} + \right.$$
$$\left. + \frac{(40-38)^5}{5!} + \frac{(41-40)^1}{1!}\right] = \left[\frac{2}{1} + \frac{1}{6} + \frac{1}{120} + \frac{1}{6} + \frac{1}{1} + \frac{4}{2}\right] \times$$
$$\times \left[\frac{1}{1} + \frac{8}{6} + \frac{1}{120} + \frac{27}{6} + \frac{5}{1}\right] \times \left[\frac{3}{1} + \frac{8}{6} + \frac{4}{1} + \frac{1}{2} + \frac{1}{6} + \frac{32}{120} + \frac{1}{1}\right] = 657{,}85.$$

5. Перестановка π_k с максимальной областью оптимальности

Если существует одноэлементное доминирующее множество $\{\pi_k\}$ для задачи $1|p_i^L \leqslant p_i \leqslant p_i^U|\sum C_i$, то перестановка π_k обслуживания требований множества $\mathcal{J}$ является оптимальной для задачи $1|p|\sum C_i$ при любом сценарии $p \in T$. В соответствии с определением 2 для такой перестановки π_k должно выполняться равенство $\mathcal{OR}(\pi_k, T) = T$.

5.1. Максимально возможная область оптимальности для заданных сценариев

Докажем необходимые и достаточные условия для существования перестановки $\pi_k \in \Pi$ с максимально возможной областью оптимальности для заданного множества сценариев T, т.е. докажем критерий существования перестановки π_k, для которой справедливо равенство $\mathcal{OR}(\pi_k, T) = T$.

Теорема 5. Область оптимальности для перестановки $\pi_k = (J_{k_1}, \ldots$ $\ldots, J_{k_n}) \in \Pi$ *является максимально возможной для заданного множества сценариев* T*, т.е.* $\mathcal{OR}(\pi_k, T) = T$*, тогда и только тогда, когда в перестановке* π_k *для каждого требования* $J_{k_r} \in \mathcal{J}$ *выполняется следующее равенство:*

$$\left[l_{k_s}^{opt}, u_{k_s}^{opt}\right] = \left[p_{k_s}^L, p_{k_s}^U\right]. \tag{17}$$

Доказательство.

Достаточность. Пусть равенство (17) выполняется для каждого требования $J_{k_s} \in \mathcal{J}$ в перестановке $\pi_k = (J_{k_1}, \ldots, J_{k_n})$. Тогда по определению 1 должны выполняться и равенства $\mathcal{OB}(\pi_k, T) = \times_{k_{i_r} \in M}\left[l_{k_{i_r}}^{opt}, u_{k_{i_r}}^{opt}\right] =$ $= \times_{k_{i_r} \in M}\left[p_{k_{i_r}}^L, p_{k_{i_r}}^U\right] = T$, такие что множество $M = (k_{i_1}, \ldots, k_{i_{|M|}})$, $k_{i_1} < \ldots$ $\ldots < k_{i_{|M|}}$, представляет собой упорядоченное множество $\{k_1, \ldots, k_n\} =$ $= \{1, \ldots, n\}$, для которого $n = |M|$. Из определения 1 следует, что перестановка π_k является оптимальной для задачи $1|p'|\sum C_i$ при любом сценарии $p' \in \mathcal{OB}(\pi_k, T) = T$. Согласно определению 2 получаем равенство $\mathcal{OR}(\pi_k, T) = T$. Достаточность доказана.

Необходимость. Пусть справедливо равенство $\mathcal{OR}(\pi_k, T) = T$. Однако предположим (от противного), что существует требование $J_{k_r} \in \mathcal{J}$ в перестановке $\pi_k = (J_{k_1}, \ldots, J_{k_n}) \in \Pi$ такое, что равенство (17) не выполняется.

Тогда в силу леммы 1 существует непустой отрезок неоптимальности $[l_{k_r}^{non}, u_{k_r}^{non}]$ или (и) существует непустой отрезок условной оптимальности $[l_{k_r}^{copt}, u_{k_r}^{copt}]$ для требования $J_{k_r} \in \mathcal{J}$ в перестановке $\pi_k = (J_{k_1}, \ldots, J_{k_n})$.

Если существует отрезок неоптимальности $[l_{k_r}^{non}, u_{k_r}^{non}]$, то выполняется равенство (5). Если же существует отрезок условной оптимальности $[l_{k_r}^{copt}, u_{k_r}^{copt}]$, то выполняется равенство (7). В обоих случаях существует сценарий $p^* \in (l_{k_r}^{non}, u_{k_r}^{non}) \bigcup (l_{k_r}^{copt}, u_{k_r}^{copt}) \subseteq T$ такой, что перестановка π_k не является оптимальной для задачи $1|p^*|\sum C_i$ со сценарием $p^* \in T$. Следовательно, в соответствии с определением 2 получаем соотношение $\mathcal{OR}(\pi_k, T) \neq T$, противоречащее предположению о том, что равенство $\mathcal{OR}(\pi_k, T) = T$ выполняется. Это противоречие завершает доказательство теоремы.

Из теоремы 5 и следствия 1 получаем следующее утверждение.

Следствие 3. Если для каждого требования $J_{k_r} \in \mathcal{J}$ в перестановке $\pi_k = (J_{k_1}, \ldots, J_{k_n})$ выполняется равенство (17)*, то область оптимальности $\mathcal{OR}(\pi_k, T)$ является n-мерным параллелепипедом $T \subset R_+^n$ с объемом $Vol(\pi_k, T) = \prod_{J_i \in \{\mathcal{J} : p_i^L < p_i^U\}} (p_i^U - p_i^L)$.*

Перестановка $\pi_k = (J_{k_1}, \ldots, J_{k_n})$, для которой равенства (17) выполняются для всех требований $J_{k_r} \in \mathcal{J}$, является оптимальной для задачи $1|p|\sum C_i$ с любым допустимым сценарием $p \in T$. Следовательно, множество $\{\pi_k\}$ является минимальным доминирующим множеством для задачи $1|p_i^L \leqslant p_i \leqslant p_i^U|\sum C_i$.

5.2. Как использовать перестановку с максимальной областью оптимальности

Оптимальная для задачи $1|p_i^L \leqslant p_i \leqslant p_i^U|\sum C_i$ перестановка π_k, критерий существования которой представлен в теореме 5 и следствии 3, встречается на практике довольно редко. Однако для конкретной задачи $1|p_i^L \leqslant p_i \leqslant p_i^U|\sum C_i$, возникающей на практике, как правило, одна перестановка должна быть выбрана из множества Π и затем реализована при обслуживании требований множества $\mathcal{J}$.

На основе доказанных в разделах 3–5.1 результатов можно рекомендовать выбирать для реализации такую перестановку π_t обслуживания требований множества $\mathcal{J}$, для которой объем области оптимальности $\mathcal{OR}(\pi_t, T)$ является максимальным среди всех перестановок множества Π. Если окажется, что фактический сценарий $p \in T$ обслуживания требований $\mathcal{J}$ будет принадлежать области оптимальности $\mathcal{OR}(\pi_t, T)$, то реализованная перестановка π_t будет оптимальной для фактического сценария обслуживания требований $\mathcal{J}$. Вообще говоря, чем больше объем области оптимальности $\mathcal{OR}(\pi_k, T)$, тем больше вероятность того, что перестановка π_k будет оптимальной для фактического сценария обслуживания требований $\mathcal{J}$. Поэтому важными задачами для дальнейших исследований представляются разработка эффективных алгоритмов построения перестановки π_t с

максимальным объемом $Vol^{\max}(\pi_t, T) = \max\{\, Vol^{\max}(\pi_k, T) : \pi_k \in \Pi\}$ области оптимальности $\mathcal{OR}(\pi_t, T)$ и тестирование таких алгоритмов на задачах $1|p_i^L \leqslant p_i \leqslant p_i^U| \sum C_i$ практической размерности.

В общем случае задачи $1|p_i^L \leqslant p_i \leqslant p_i^U| \sum C_i$ разность

$$1 - \frac{Vol^{\max}(\pi_k, T)}{\prod_{J_i \in \{\mathcal{J}: l_i^{opt} < u_i^{opt}\}} (p_i^U - p_i^L)} =: \mu$$

можно рассматривать как меру неопределенности (или меру сложности) задачи. В частности, если $\mu = 0$, то имеется гарантия того, что перестановка π_t с максимальным объемом $Vol^{\max}(\pi_t, T)$ области оптимальности $\mathcal{OR}(\pi_t, T)$ будет оптимальной для фактического сценария обслуживания требований множества $\mathcal{J}$, несмотря на неопределенность заданных сценариев T. Наоборот, если значение μ равно единице (или близко к единице), то вероятность того, что перестановка π_t будет оптимальной для фактического сценария обслуживания требований множества $\mathcal{J}$, равна нулю (близка к нулю). В таких случаях для решения неопределенной задачи $1|p_i^L \leqslant p_i \leqslant p_i^U| \sum C_i$ можно рекомендовать использование приближенных алгоритмов таких, как алгоритм U2, описанный в [1], или описанный в [10] алгоритм 3, который ориентирован на достижение наименьшей погрешности полученного решения.

6. Заключение

Задачи составления расписаний обслуживания требований с неопределенными числовыми данными на одном приборе возникают, например, при планировании рабочего времени работника на определенный период времени (рабочий день, неделю или месяц). Как правило, можно заранее оценить диапазоны возможных длительностей планируемых работ. Можно предполагать, что множество планируемых работ существенно не меняется в ходе реализации расписания. Критерий минимизации суммы моментов завершения обслуживания требований (среднего времени обслуживания требований) можно рассматривать как суммарный показатель эффективности выполнения работником заданного множества работ.

В [27] описан другой пример задачи $1|p_i^L \leqslant p_i \leqslant p_i^U| \sum C_i$, возникающей при поиске оптимального расписания доставки продукции от изготовителя в торговую сеть города при использовании одного транспортного средства. Время доставки продукции в магазин зависит от множества факторов таких, как автомобильные пробки, погодные условия, состояние транспортного средства и дорожного покрытия.

Задачи $1|p_i^L \leqslant p_i \leqslant p_i^U| \sum C_i$ могут возникать и в некоторых многостадийных обслуживающих системах, если один обслуживающий прибор является “узким местом” производственного процесса и для длительностей обслуживания требований на этом приборе известны только границы возможных значений.

Полученные в разделах 3–5 результаты и алгоритмы 1 и 2 можно использовать при построении перестановки $\pi_t \in \Pi$ обслуживания заданных требований с наибольшим объемом $Vol^{\max}(\pi_t, T)$ области оптимальности $\mathcal{OR}(\pi_t, T)$.

Использование перестановки π_t для обслуживания заданных требований позволяет повысить вероятность получения фактически оптимального расписания, несмотря на то, что законы распределения вероятностей случайных длительностей обслуживания требований не известны при построении расписания для задачи $1|p_i^L \leqslant p_i \leqslant p_i^U| \sum C_i$.

ПРИЛОЖЕНИЕ

При доказательстве леммы 4 будем рассматривать задачу

$$1|\widehat{p}_i^L \leqslant p_i \leqslant \widehat{p}_i^U| \sum C_i$$

вместо задачи $1|p_i^L \leqslant p_i \leqslant p_i^U| \sum C_i$ (теорема 2). Поскольку $\mathcal{OR}(\pi_k, T) \neq \emptyset$, то согласно теореме 3 не существует ни одного требования $J_{k_r} \in \mathcal{J}$, которое не имеет условной оптимальности и одновременно не имеет отрезка оптимальности в перестановке π_k. По условию леммы 4 каждое требование $J_{k_r} \in \mathcal{J}$ не имеет отрезка оптимальности в перестановке π_k.

Учитывая замечание 1 и лемму 1, получаем равенство

$$\left[\widehat{p}_{k_r}^L,\ \widehat{p}_{k_r}^U\right] = [l_{k_r}^{copt}, u_{k_r}^{copt}],$$

которое выполняется для каждого требования $J_{k_r} \in \mathcal{J}$. Из полученного равенства следует, что множество $S(\pi_k)$ секций перестановки π_k не содержит ни одной тривиальной секции. Построим разбиение (12) охватов $\left[\widehat{p}_{k_j}^L, \widehat{p}_{k_{j+m_j}}^U\right]$ всех секций $s_j^{\pi_k} \in S(\pi_k)$ на следующие подынтервалы условной оптимальности:

$$\left[l_1^j\left(s_j^{\pi_k}\right), u_1^j\left(s_j^{\pi_k}\right)\right) \bigcup \ldots \bigcup \left[l_i^j\left(s_j^{\pi_k}\right), u_i^j\left(s_j^{\pi_k}\right)\right) \bigcup \ldots$$
$$\ldots \bigcup \left[l_{n(j)}^j\left(s_j^{\pi_k}\right), u_{n(j)}^j\left(s_j^{\pi_k}\right)\right] = \left[\widehat{p}_{k_j}^L, \widehat{p}_{k_{j+m_j}}^U\right].$$

Далее методом математической индукции по мощности $|\mathcal{J}_i^j|$ множества $\mathcal{J}_i^j$ докажем, что для каждого подынтервала $\left[l_i^j\left(s_j^{\pi_k}\right), u_i^j\left(s_j^{\pi_k}\right)\right)$ условной оптимальности в построенном разбиении (12) выполняются следующие равенства:

$$\text{(П.1)} \qquad Vol\left(\widehat{J}_i^j, T\right) = \frac{\left(u_i^j\left(s_j^{\pi_k}\right) - l_i^j\left(s_j^{\pi_k}\right)\right)^{|\mathcal{J}_i^j|}}{|\mathcal{J}_i^j|!},$$

$$\text{(П.2)} \qquad \mathcal{OR}\left(\widehat{J}_i^j, T\right) = Pir^{opt}\widehat{J}_i^j = Pir^{opt}\left(J_{k_i}, \ldots, J_{k_{|\mathcal{J}_i^j|}}\right),$$

где основанием $|\mathcal{J}_i^j|$-мерной пирамиды $Pir^{opt}\widehat{J}_i^j = Pir^{opt}\left(J_{k_i}, \ldots, J_{k_{|\mathcal{J}_i^j|}}\right)$ является $(|\mathcal{J}_i^j| - 1)$-мерная пирамида, а длина высоты всех пирамид равна $\left(u_i^j\left(s_j^{\pi_k}\right) - l_i^j\left(s_j^{\pi_k}\right)\right)$.

Покажем вначале, что при $|\mathcal{J}_i^j| = 2$ пирамида $Pir^{opt}\widehat{J}_i^j = Pir^{opt}(J_{k_i}, J_{k_{i+1}})$ превращается в треугольник (как вырожденный случай пирамиды) с основанием $\left[l_i^j\left(s_j^{\pi_k}\right), u_i^j\left(s_j^{\pi_k}\right)\right]$ и высотой той же длины $\left(u_i^j\left(s_j^{\pi_k}\right) - l_i^j\left(s_j^{\pi_k}\right)\right)$, что и длина основания треугольника. Такой треугольник $Pir^{opt}\widehat{J}_1^1 = Pir^{opt}(J_1, J_2)$ представлен на рис. 4,*a* для подынтервала

$$[4, 8) = \left[l_1^1(J_1, J_2),\ u_1^1(J_1, J_2)\right)$$

условной оптимальности для перестановки $\pi_2 = (J_1, \ldots, J_{10}) \in \Pi$ для примера 3. Из теоремы 1 следует, что для того, чтобы порядок $(J_{k_i}, J_{k_{i+1}}) = \widehat{J}_i^j$ двух требований был оптимальным в перестановке $\pi_k \in \Pi$, необходимо и достаточно, чтобы длительности p_{k_i} и $p_{k_{i+1}}$ требований J_{k_i} и $J_{k_{i+1}}$ удовлетворяли неравенству $p_{k_i} \leqslant p_{k_{i+1}}$, а с учетом принадлежности допустимого сценария $(p_{k_1}, \ldots, p_{k_n})$ заданному множеству T допустимые длительности p_{k_i} и $p_{k_{i+1}}$ должны удовлетворять следующей системе неравенств:

$$\text{(П.3)} \qquad \begin{cases} p_{k_i} \leqslant p_{k_{i+1}}, \\ p_{k_i}^L \leqslant p_{k_i} \leqslant p_{k_i}^U, \\ p_{k_{i+1}}^L \leqslant p_{k_{i+1}} \leqslant p_{k_{i+1}}^U. \end{cases}$$

Система (П.3) определяет треугольник $Pir^{opt}\widehat{J}_i^j = Pir^{opt}(J_{k_i}, J_{k_{i+1}})$. Иными словами, системе (П.3) удовлетворяют все точки, принадлежащие треугольнику $Pir^{opt}\widehat{J}_i^j$, и только они. Таким образом, равенство (П.2) доказано для случая $|\mathcal{J}_i^j| = 2$. Поскольку площадь треугольника $Pir^{opt}\widehat{J}_i^j$ равна произведению основания на половину высоты: $\frac{(u_i^j(s_j^{\pi_k}) - l_i^j(s_j^{\pi_k}))^2}{2}$, то и равенство (П.1) доказано для случая $|\mathcal{J}_i^j| = 2$.

Рассмотрим следующее по мощности множество $\mathcal{J}_i^j$, т.е. $|\mathcal{J}_i^j| = 3$, и покажем, что в этом случае область оптимальности

$$\mathcal{OR}(\widehat{J}_i^j, T) = \mathcal{OR}((J_{k_i}, J_{k_{i+1}}, J_{k_{i+2}}), T)$$

представляет собой 3-мерную пирамиду $Pir^{opt}\widehat{J}_i^j$, длина высоты которой равна $\left(u_i^j\left(s_j^{\pi_k}\right) - l_i^j\left(s_j^{\pi_k}\right)\right)$ и основанием которой является треугольник с высотой той же длины $\left(u_i^j\left(s_j^{\pi_k}\right) - l_i^j\left(s_j^{\pi_k}\right)\right)$ и с основанием $\left[l_i^j\left(s_j^{\pi_k}\right), u_i^j\left(s_j^{\pi_k}\right)\right]$. Отметим, что такая пирамида оптимальности $Pir^{opt}\widehat{J}_1^1 = Pir^{opt}(J_2, J_3, J_4)$ представлена на рис. 4,*b* для подынтервала

$$[9, 13) = \left[l_3^1(J_2, J_3, J_4),\ u_3^1(J_2, J_3, J_4)\right)$$

условной оптимальности для перестановки $\pi_2 = (J_1, \ldots, J_{10})$ для примера 3. Из теоремы 1 следует, что для того, чтобы порядок $(J_{k_i}, J_{k_{i+1}}, J_{k_{i+2}}) = \widehat{J}_i^j$ трех требований был оптимальным в перестановке $\pi_k \in \Pi$, необходимо и достаточно, чтобы длительности p_{k_i}, $p_{k_{i+1}}$ и $p_{k_{i+2}}$ требований J_{k_i}, $J_{k_{i+1}}$ и $J_{k_{i+2}}$

удовлетворяли неравенствам $p_{k_i} \leqslant p_{k_{i+1}} \leqslant p_{k_{i+2}}$, а с учетом принадлежности допустимого сценария $(p_{k_1}, \ldots, p_{k_n})$ заданному множеству T длительности p_{k_i}, $p_{k_{i+1}}$ и $p_{k_{i+2}}$ должны удовлетворять следующей системе неравенств:

$$\text{(П.4)} \qquad \begin{cases} p_{k_i} \leqslant p_{k_{i+1}} \leqslant p_{k_{i+2}}, \\ p^L_{k_i} \leqslant p_{k_i} \leqslant p^U_{k_i}, \\ p^L_{k_{i+1}} \leqslant p_{k_{i+1}} \leqslant p^U_{k_{i+1}}, \\ p^L_{k_{i+2}} \leqslant p_{k_{i+2}} \leqslant p^U_{k_{i+2}}. \end{cases}$$

Система (П.4) определяет 3-мерную пирамиду

$$Pir^{opt}\widehat{J}_i^j = Pir^{opt}(J_{k_i},\ J_{k_{i+1}},\ J_{k_{i+2}}),$$

основанием которой является треугольник

$$\left[\left(p^L_{k_i},\ p^L_{k_{i+1}},\ p^U_{k_{i+2}}\right),\ \left(p^L_{k_i},\ p^U_{k_{i+1}},\ p^U_{k_{i+2}}\right),\ \left(p^U_{k_i},\ p^U_{k_{i+1}},\ p^U_{k_{i+2}}\right)\right],$$

а высотой – отрезок

$$\left[\left(p^L_{k_i},\ p^L_{k_{i+1}},\ p^U_{k_{i+2}}\right),\ \left(p^L_{k_i},\ p^L_{k_{i+1}},\ p^L_{k_{i+2}}\right)\right].$$

Следовательно, системе (П.4) удовлетворяют все точки пирамиды $Pir^{opt}\widehat{J}_i^j = Pir^{opt}(J_{k_i}, J_{k_{i+1}}, J_{k_{i+2}})$ и только они. Таким образом, равенство (П.2) доказано и для случая $|\mathcal{J}_i^j| = 3$. Поскольку объем 3-мерной пирамиды $Pir^{opt}\widehat{J}_i^j$ равен произведению площади основания, т.е. площади треугольника

$$\left[\left(p^L_{k_i}, p^L_{k_{i+1}}, p^U_{k_{i+2}}\right), \left(p^L_{k_i}, p^U_{k_{i+1}}, p^U_{k_{i+2}}\right), \left(p^U_{k_i}, p^U_{k_{i+1}}, p^U_{k_{i+2}}\right)\right]$$

на треть высоты:

$$\frac{\left(u_i^j\left(s_j^{\pi_k}\right) - l_i^j\left(s_j^{\pi_k}\right)\right)^2}{2}\,\frac{\left(u_i^j\left(s_j^{\pi_k}\right) - l_i^j\left(s_j^{\pi_k}\right)\right)}{3} =$$
$$= \frac{\left(u_i^j\left(s_j^{\pi_k}\right) - l_i^j\left(s_j^{\pi_k}\right)\right)^3}{3!} = Vol\left(\widehat{J}_i^j, T\right),$$

то равенство (П.1) доказано и для случая $|\mathcal{J}_i^j| = 3$.

Сделаем индуктивное предположение, т.е. предположим, что оба равенства (П.1) и (П.2) выполняются в случае $|\mathcal{J}_i^j| = d$, т.е. область оптимальности $\mathcal{OR}(\widehat{J}_i^j, T)$ представляет собой d-мерную пирамиду $Pir^{opt}\widehat{J}_i^j = Pir^{opt}(J_{k_i}, \ldots, J_{k_t})$, основанием которой является $(d-1)$-мерная пирамида, а $(u_i^j(s_j^{\pi_k}) - l_i^j(s_j^{\pi_k}))$ – это длина высоты каждой из этих пирамид. Покажем, что на основе индуктивного предположения можно получить равенства (П.1) и (П.2) для случая $|\mathcal{J}_i^j| = d + 1$.

Рассмотрим подынтервал $\left[l_i^j\left(s_j^{\pi_k}\right), u_i^j\left(s_j^{\pi_k}\right)\right)$ условной оптимальности, для которого

$$\widehat{J}_i^j = \left(J_{k_i}, \ldots, J_{k_{|\mathcal{J}_i^j|-1}}, J_{k_{|\mathcal{J}_i^j|}}\right) \quad \text{и} \quad |\mathcal{J}_i^j| = d+1.$$

Из индуктивного предположения следует, что для множества требований $\mathcal{J}_i^j \setminus \left\{J_{k_{|\mathcal{J}_i^j|}}\right\}$ оба равенства (П.1) и (П.2) выполняются, и область оптимальности $\mathcal{OR}\left(\left(J_{k_i}, \ldots, J_{k_{|\mathcal{J}_i^j|-1}}\right), T\right)$ представляет собой d-мерную пирамиду

$$Pir^{opt}\left(J_{k_i}, \ldots, J_{k_t}\right) = \mathcal{OR}\left(\left(J_{k_i}, \ldots, J_{k_{|\mathcal{J}_i^j|-1}}\right), T\right).$$

Следовательно, по теореме 1 для того, чтобы порядок обслуживания требований $\left(J_{k_i}, \ldots, J_{k_{|\mathcal{J}_i^j|-1}}\right)$ был оптимальным в перестановке $\pi_k \in \Pi$, необходимо и достаточно, чтобы длительности $p_{k_i}, \ldots, p_{k_{|\mathcal{J}_i^j|-1}}$ требований $J_{k_i}, \ldots, J_{k_{|\mathcal{J}_i^j|-1}}$ удовлетворяли следующей системе неравенств:

$$\text{(П.5)} \qquad \begin{cases} p_{k_i} \leqslant \ldots \leqslant p_{k_{|\mathcal{J}_i^j|-1}}, \\ p_{k_i}^L \leqslant p_{k_i} \leqslant p_{k_i}^U, \\ \ldots \\ p_{k_{|\mathcal{J}_i^j|-1}}^L \leqslant p_{k_{|\mathcal{J}_i^j|-1}} \leqslant p_{k_{|\mathcal{J}_i^j|-1}}^U. \end{cases}$$

Добавив к системе (П.5) неравенства

$$p_{k_{|\mathcal{J}_i^j|-1}} \leqslant p_{k_{|\mathcal{J}_i^j|}} \quad \text{и} \quad p_{k_{|\mathcal{J}_i^j|}}^L \leqslant p_{k_{|\mathcal{J}_i^j|}} \leqslant p_{k_{|\mathcal{J}_i^j|}}^U,$$

получим следующую систему неравенств:

$$\text{(П.6)} \qquad \begin{cases} p_{k_i} \leqslant \ldots \leqslant p_{k_{|\mathcal{J}_i^j|-1}}, \\ p_{k_{|\mathcal{J}_i^j|-1}} \leqslant p_{k_{|\mathcal{J}_i^j|}}, \\ p_{k_i}^L \leqslant p_{k_i} \leqslant p_{k_i}^U, \\ \ldots \\ p_{k_{|\mathcal{J}_i^j|-1}}^L \leqslant p_{k_{|\mathcal{J}_i^j|-1}} \leqslant p_{k_{|\mathcal{J}_i^j|-1}}^U, \\ p_{k_{|\mathcal{J}_i^j|}}^L \leqslant p_{k_{|\mathcal{J}_i^j|}} \leqslant p_{k_{|\mathcal{J}_i^j|}}^U, \end{cases}$$

которая определяет $(d+1)$-мерную пирамиду

$$Pir^{opt}\widehat{J}_i^j = Pir^{opt}\left(J_{k_i}, \ldots, J_{k_{|\mathcal{J}_i^j|}}\right),$$

основанием которой является d-мерная пирамида $Pir^{opt}\left(J_{k_i},\ldots,J_{k_{|\mathcal{J}_i^j|-1}}\right)$, а разность $\left(u_i^j\left(s_j^{\pi_k}\right)-l_i^j\left(s_j^{\pi_k}\right)\right)$ равна длине высоты каждой из двух пирамид.

Из сделанного предположения и теоремы 1 следует, что область оптимальности $\mathcal{OR}\left(\left(J_{k_i},\ldots,J_{k_{|\mathcal{J}_i^j|}}\right),T\right)$ представляет собой $(d+1)$-мерную пирамиду $Pir^{opt}\left(J_{k_i},\ldots,J_{k_{|\mathcal{J}_i^j|}}\right)=\mathcal{OR}\left(\left(J_{k_i},\ldots,J_{k_{|\mathcal{J}_i^j|}}\right),T\right)$. Следовательно, равенство (П.2) установлено и для случая $|\mathcal{J}_i^j|=d+1$. Поскольку объем $(d+1)$-мерной пирамиды $Pir^{opt}\widehat{J}_i^j$ равен произведению объема основания, т.е. объема d-мерной пирамиды $Pir^{opt}\left(J_{k_i},\ldots,J_{k_{|\mathcal{J}_i^j|-1}}\right)=\mathcal{OR}\left(\left(J_{k_i},\ldots,J_{k_{|\mathcal{J}_i^j|-1}}\right),T\right)$ на высоту, деленную на $(d+1)$:

$$\frac{\left(u_i^j\left(s_j^{\pi_k}\right)-l_i^j\left(s_j^{\pi_k}\right)\right)^d}{d!}\frac{\left(u_i^j\left(s_j^{\pi_k}\right)-l_i^j\left(s_j^{\pi_k}\right)\right)}{d+1}=$$
$$=\frac{(u_i^j(s_j^{\pi_k})-l_i^j(s_j^{\pi_k}))^{d+1}}{(d+1)!}=Vol(\widehat{J}_i^j,T),$$

то равенство (П.1) установлено и для случая $|\mathcal{J}_i^j|=d+1$. Таким образом, равенства (П.1) и (П.2) доказаны методом математической индукции.

Равенство (14) следует из равенства (П.2) и того, что при любом сценарии $p\in T$ длительность каждого требования $J_i\in\mathcal{J}$ принимает единственное значение p_i. Равенство (13) следует из замечания 2 и равенства (14). Равенство (15) следует из равенств (13), (14) и (П.1). Лемма 4 доказана.

СПИСОК ЛИТЕРАТУРЫ

1. *Allahverdi A., Aydilek H., Aydilek A.* Single machine scheduling problem with interval processing times to minimize mean weighted completion times // Comput. Oper. Res. 2014. V. 51. P. 200–207.
2. *Goren S., Sabuncuoglu I.* Robustness and stability measures for scheduling: single-machine environment // IIE Transact. 2008. V. 40. P. 66–83.
3. *Pereira J.* The robust (minmax regret) single machine scheduling with interval processing times and total weighted completion time objective // Comput. Oper. Res. 2016. V. 66. P. 141–152.
4. *Lu C.-C., Lin S.-W., Ying K.-C.* Robust scheduling on a single machine total flow time // Comput. Oper. Res. 2012. V. 39. P. 1682–1691.
5. *Sotskov Y.N., Werner F.* Sequencing and Scheduling with Inaccurate Data. N.Y., USA: Nova Science Publishers, Hauppauge, 2014.
6. *Braun O., Lai T.C., Schmidt G., Sotskov Y.N.* Stability of Johnson's schedule with respect to limited machine availability // Int. J. Product. Res. 2002. V. 40. No. 17. P. 4381–4400.
7. *Davis W., Jones A.* A real-time production scheduler for a stochastic manufacturing environment // Int. J. Product. Res. 1988. V. 1. No. 2. P. 101–112.

8. *Grabot B., Geneste L.* Dispatching rules in scheduling: a fuzzy approach // Int. J. Product. Res. 1994. V. 32. No. 4. P. 903–915.

9. *Kasperski A., Zielinski P.* Possibilistic minmax regret sequencing problems with fuzzy parameters // IEEE Transact. Fuzzy Syst. 2011. V. 19. P. 1072–1082.

10. *Sotskov Y.N., Egorova N.G.* Single machine scheduling problem with interval processing times and total completion time objective // Algorithms. 2018. V. 11. No. 66. P. 21–40.

11. *Sotskov Y.N., Lai T.-C.* Minimizing total weighted flow time under uncertainty using dominance and a stability box // Comput. Oper. Res. 2012. V. 39. No. 6. P. 1271–1289.

12. *Lai T.-C., Sotskov Y.N., Egorova N.G., Werner F.* The optimality box in uncertain data for minimising the sum of the weighted job completion times // Int. J. Product. Res. 2018. V. 56. No. 19. P. 6336–6362.

13. *Сотсков Ю.Н., Егорова Н.Г.* Многогранники устойчивости оптимальной перестановки обслуживания требований // АиТ. 2014. № 7. С. 136–154.

    Sotskov Y.N., Egorova N.G. Stability polyhedra of optimal permutation of jobs servicing // Autom. Remote Control. 2014. V. 75. No. 7. P. 1267–1282.

14. *Sotskov Y.N., Lai T.-C., Werner, F.* Measures of problem uncertainty for scheduling with interval processing times // OR Spectrum. 2013. V. 35. P. 659–689.

15. *Sotskov Y.N., Egorova N.G., Lai T.-C.* Minimizing total weighted flow time of a set of jobs with interval processing times // Math. Comput. Modell. 2009. V. 50. P. 556–573.

16. *Lai T.-C., Sotskov Y.N., Sotskova N., Werner F.* Optimal makespan scheduling with given bounds of processing times // Math. Comput. Modell. 1997. V. 26. No. 3. P. 67–86.

17. *Сотсков Ю.Н., Егорова Н.Г., Вернер Ф.* Минимизация суммарного взвешенного времени обслуживания требований с неопределенными данными: метод, основанный на устойчивости // АиТ. 2010. № 10. С. 26–49.

    Sotskov Y.N., Egorova N.G., Werner F. Minimizing total weighted completion time with uncertain data: a stability approach // Automat. Remote Control. 2010. V. 71. No. 10. P. 2038–2057.

18. *Pinedo M.* Scheduling: Theory, Algorithms and Systems. N.J., USA: Prentice-Hall, Englewood Cliffs, 2002.

19. *Graham R.E., Lawler E.L., Lenstra J.K., Rinnooy Kan A.H.G.* Optimization and approximation in deterministic sequencing and scheduling: a survey // Ann. Discret. Math. 1979. V. 5. P. 287–326.

20. *Smith W.* Various optimizers for single-stage production // Naval Res. Logist. Quarterly. 1956. V. 3. No. 1. P. 59–66.

21. *Kouvelis P., Yu G.* Robust Discrete Optimization and its Application. Boston, USA: Kluwer Academ. Publish., 1997.

22. *Burdett R.L., Kozan E.* Techniques to effectively buffer schedules in the face of uncertainties // Comput. Industr. Engineer. 2015. V. 87. P. 16–29.

23. *Kasperski A., Zielinski P.* A 2-approximation algorithm for interval data minmax regret sequencing problems with total flow time criterion // Oper. Res. Lett. 2008. V. 36. P. 343–344.

24. *Daniels R.L., Kouvelis P.* Robust scheduling to hedge against processing time uncertainty in single stage production // Management Sci. 1995. V. 41. No. 2. P. 363–376.

25. *Yang J., Yu G.* On the robust single machine scheduling problem // J. Combinat. Optim. 2002. V. 6. P. 17–33.

26. *Harikrishnan K.K., Ishii H.* Single machine batch scheduling problem with resource dependent setup and processing time in the presence of fuzzy due date // Fuzzy Optim. Decision Making. 2005. V. 4. P. 141–147.

27. *Sotskov Y.N., Egorova N.G.* Single-machine scheduling with uncertain durations for optimizing service logistics with one truck // EURO Mini-conf. Logist. Anal., June 18–19, 2018, Minsk, Belarus. 29 p.

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

Поступила в редакцию 03.07.2019

После доработки 14.10.2019

Принята к публикации 28.11.2019

© 2020 г. Я. ЗИНДЕР (yakov.zinder@uts.edu.au)
(Технологический университет, Сидней, Австралия),
А.А. ЛАЗАРЕВ, д-р физ.-мат. наук (jobmath@mail.ru)
(Институт проблем управления им. В.А. Трапезникова РАН, Москва;
Национальный исследовательский университет
"Высшая школа экономики", Москва),
Е.Г. МУСАТОВА, канд. физ.-мат. наук (nekolyap@mail.ru)
(Институт проблем управления им. В.А. Трапезникова РАН, Москва)

КОРРЕКТИРОВКА РАСПИСАНИЯ ДВИЖЕНИЯ НА ЧАСТИЧНО ЗАБЛОКИРОВАННОМ СЕГМЕНТЕ ЖЕЛЕЗНОЙ ДОРОГИ С РАЗЪЕЗДОМ[1]

Представлен полиномиальный алгоритм корректировки расписания движения поездов для случая, когда один из путей двухпутной железной дороги становится недоступным, оставшийся путь содержит разъезд, а все поезда делятся на две категории: приоритетные поезда, например пассажирские, и обычные поезда, к которым относятся большинство грузовых поездов. Представленный алгоритм минимизирует негативное влияние, оказываемое блокировкой пути, сначала для приоритетных поездов, а затем для обычных поездов на множестве всех расписаний, оптимальных для приоритетных поездов.

Ключевые слова: однопутная железная дорога, динамическое программирование, перепланирование, полиномиальный алгоритм.

DOI: 10.31857/S0005231020050062

1. Введение

Сбои, такие как аварии или повреждение пути, часто приводят к вре́менному закрытию одной колеи на двухпутном участке железной дороги. В таких ситуациях необходимо планировать движение поездов в обоих направлениях на оставшемся пути с целью минимизации влияния блокировки. Корректировка железнодорожных расписаний является областью активных исследований в течение нескольких десятилетий [1]. В статье добавляется к этим исследованиям полиномиальный алгоритм на основе динамического программирования для случая, когда оставшаяся колея имеет разъезд, а множество всех поездов разделено на две категории: приоритетные поезда, такие как пассажирские, и обычные, такие как большинство грузовых поездов.

Более подробно рассмотрим двухпутную железную дорогу между двумя точками A и B, где один железнодорожный путь заблокирован и движение

[1] Работа выполнена при частичной финансовой поддержке Российского научного фонда (проект № 17-19-01665).

всех поездов должно быть перепланировано на оставшемся пути. Оставшийся путь имеет разъезд, т.е. участок, позволяющий поезду разминуться со встречным поездом. В этом случае пропускающий поезд должен стоять в разъезде, в то время как проходящий поезд не может остановиться в разъезде. В разъезде одновременно может сделать остановку только один поезд. Все поезда имеют одинаковую постоянную скорость. Время, которое требуется поезду для прохождения отрезка дороги между точкой $s \in \{A, B\}$ и разъездом, обозначим через p_s. Не ограничивая общности, будем полагать, что $p_A \geqslant p_B$.

По соображениям безопасности два поезда не могут прибыть в разъезд одновременно. Пусть β — минимальный временно́й промежуток между двумя такими прибытиями. По той же причине для каждой конечной точки рассматриваемого отрезка пути два момента прибытия, два момента отправления, так же как и момент прибытия и момент отправления двух поездов, должны отличаться как минимум на величину β. При этом два поезда могут покинуть разъезд одновременно, если они движутся в разных направлениях. Предположение, что все эти требования безопасности имеют одинаковое минимальное время β, упрощает изложение материала, но не является существенным. Также будем полагать, что $\beta < p_B$.

Множество поездов состоит из двух категорий: приоритетные поезда и обычные поезда. Для каждого $s \in \{A, B\}$ пусть $\bar{s} = \{A, B\} \setminus \{s\}$. Обозначим через N_s^p и N_s^o множества приоритетных и обычных поездов, которые должны проходить путь в направлении от точки s к точке $\bar{s}$. Будем считать, что для каждого $s \in \{A, B\}$ $N_s^p \cup N_s^o \neq \emptyset$, поскольку в противном случае задача корректировки расписания не возникает.

Исходное расписание, построенное в предположении, что оба пути находятся в рабочем состоянии, выделяет каждому поезду $j \in N_s^\alpha$ временно́е окно $[r_j^{s,\alpha}, d_j^{s,\alpha}]$, в пределах которого поезд j должен пройти через рассматриваемый двухпутный участок железнодорожной сети. В соответствии с этим исходным расписанием один путь выделяется для всех поездов, приоритетных и обычных, идущих из A в B, а другой путь выделяется для всех поездов, приоритетных и обычных, движущихся в противоположном направлении, из B в A. Следовательно, для любых двух поездов $j \in N_s^\alpha$ и $j' \in N_s^{\alpha'}$, движущихся из s в $\bar{s}$, соответствующие временны́е окна удовлетворяют условию

$$r_j^{s,\alpha} \neq r_{j'}^{s,\alpha'} \quad \text{и} \quad r_j^{s,\alpha} > r_{j'}^{s,\alpha'} \text{ влечет } d_j^{s,\alpha} > d_{j'}^{s,\alpha'}. \tag{1}$$

В результате корректировки расписания для каждого поезда $j \in N_s^\alpha$ определяется момент времени $S_j^{s,\alpha}$, в который этот поезд должен войти на оставшийся путь и который будем называть моментом отправления поезда из точки s, и момент времени $C_j^{s,\alpha}$, в который поезд должен покинуть рассматриваемый путь и который будем называть временем прибытия в точку $\bar{s}$. Любое такое новое расписание должно удовлетворять следующему условию, которое диктует наличие исходных временны́х окон:

(s1) для любого $s \in \{A, B\}$, любого $\alpha \in \{p, o\}$ и любого поезда $j \in N_s^\alpha$ время отправления $S_j^{s,\alpha}$ этого поезда удовлетворяет неравенству $S_j^{s,\alpha} \geqslant r_j^{s,\alpha}$.

Требуется построить расписание, минимизирующее целевую функцию

$$\max_{s\in\{A,B\}} \max_{j\in N_s^p} [C_j^{s,p} - d_j^{s,p}] \tag{2}$$

и которое минимизирует целевую функцию

$$\sum_{s\in\{A,B\}} \sum_{j\in N_s^o} [C_j^{s,o} - r_j^{s,o}] \tag{3}$$

на множестве всех расписаний, оптимальных для (2).

Целевая функция (3) является лишь одним из возможных показателей влияния блокировки одного из путей на обычные поезда. Так, приведенная далее оптимизационная процедура может быть легко модифицирована для случая, когда вместо (3) используется

$$\max_{s\in\{A,B\}} \max_{j\in N_s^o} [C_j^{s,o} - d_j^{s,o}]. \tag{4}$$

Существует множество публикаций по планированию и корректировке расписания для однопутной железной дороги (ряд ссылок можно найти в [1, 2]). Среди них публикации [3–5] наиболее тесно связаны с данной статьей. Как и настоящая статья, [3] касается корректировки расписания в случае, когда первоначальное расписание полностью функционирующего двухпутного участка железнодорожной сети определяет временно́е окно для каждого поезда и рассматривает несколько категорий поездов. В отличие от данной статьи [3] предполагает отсутствие разъезда и отражает существование различных типов поездов путем введения весов в целевую функцию.

Другой близкой публикацией является [5], которая касается оптимизации упорядоченных целевых функций, но в отличие от данной статьи предполагает, что все поезда доступны одновременно (такая ситуация может возникнуть после полной блокировки рассматриваемого участка железной дороги). Кроме того, [5] предполагает, что на оставшемся пути нет разъезда.

И, наконец, [4] касается планирования на однопутной железной дороге, которая имеет разъезд, но аналогично [5] подразумевает, что все поезда имеются в наличии одновременно. Кроме того, в отличие от приведенной далее процедуры оптимизации, которая предназначена для двух категорий поездов и упорядоченных целевых функций, все алгоритмы, представленные в [4], были разработаны для случаев одной целевой функции и подразумевают, что все поезда имеют одинаковый тип. Несмотря на упомянутые существенные различия между [4] и данной статьей, некоторые результаты относительно структуры оптимальных расписаний, полученные в [4], могут быть непосредственно обобщены на случай, когда исходное расписание задает временно́е окно для каждого поезда. Кроме того, основная идея из [4], что существует вложенный Беллмановский процесс принятия решений, связанный с моментами отправления определенных поездов, остается в силе для данной статьи, хотя набор состояний и реализация общей схемы динамического программирования различны.

2. Экспрессы и неэкспрессы

Рассмотрим произвольное расписание для пути, который остался доступным после блокировки. Как и в [4], поезд, который не останавливается в разъезде, будет называться *экспрессом*, тогда как поезд, который делает остановку в разъезде, будет называться *неэкспрессом*. Понятия экспресса и неэкспресса не связаны с понятиями приоритетных и обычных поездов. Другими словами, любой приоритетный поезд, так же как и любой обычный поезд, может быть как экспрессом, так и неэкспрессом в зависимости от расписания. Множество всех экспрессов, которым уступает путь один и тот же неэкспресс, будем называть *пакетом*. Для любого момента времени t поезд $j \in N_s^\alpha$ будет называться *активным* в t, если

$$S_j^{s,\alpha} \leqslant t \leqslant C_j^{s,\alpha}.$$

Если поезда $j \in N_s^\alpha$ и $j' \in N_{\bar{s}}^{\alpha'}$, т.е. два поезда, движущиеся в противоположных направлениях, одновременно активны в некоторый момент времени, то они движутся навстречу друг другу в момент времени $\max[S_j^{s,\alpha}, S_{j'}^{\bar{s},\alpha'}]$. Следовательно, один из этих двух поездов должен быть экспрессом, а другой — неэкспрессом, который пропускает этот экспресс в разъезде.

Поскольку в любой момент времени в разъезде может стоять не более одного поезда, если поезд $j \in N_s^\alpha$ обгоняет поезд j', который движется по пути в том же направлении, что и j, т.е. из s в $\bar{s}$, то во временно́м интервале $[S_j^{s,\alpha}, C_j^{s,\alpha}]$ нет активных поездов, движущихся из $\bar{s}$ в s. Следовательно, вместо ожидания j поезд j' может покинуть разъезд по крайней мере в момент времени $S_j^{s,\alpha} + p_s - \beta$, что может только улучшить значение целевой функции, поскольку она неубывающая.

Поскольку обе целевые функции (2) и (3) являются неубывающими, приведенные выше рассуждения означают, что без ограничения общности достаточно рассмотреть только расписания, которые в дополнение к (s1) удовлетворяют следующим условиям:

(s2) для каждого $s \in \{A, B\}$ и каждого $\alpha \in \{p, o\}$, для любых двух поездов $j \in N_s^\alpha$ и $j' \in N_s^\alpha$ неравенство $r_j^{s,\alpha} > r_{j'}^{s,\alpha}$ влечет $S_j^{s,\alpha} > S_{j'}^{s,\alpha}$;

(s3) для любого неэкспресса существует как минимум один экспресс, который данный неэкспресс пропускает;

(s4) все экспрессы, которые пропускает один и тот же неэкспресс, движутся в направлении, противоположном направлению движения этого неэкспресса;

(s5) неэкспресс покидает разъезд одновременно с последним экспрессом, который он пропускает.

Два экспресса, движущиеся в противоположных направлениях, не могут быть активными одновременно. Более того, два экспресса, движущиеся в одном направлении, скажем, из s в $\bar{s}$, могут отправиться из s только в моменты времени, различающиеся не менее чем на β. Следовательно, все моменты отправления экспрессов различны.

Минимально возможная разница между двумя последовательными временами отправления экспрессов определяется несколькими факторами. Если, как и в [4], все поезда имеются в наличии одновременно, то этими факторами являются направления, в которых движутся экспрессы по рассматриваемому участку, и ситуации в разъезде, когда эти экспрессы его проходят. Все возможные комбинации этих факторов определены в [4] путем приписывания каждому экспрессу пары (s, b), называемой его типом. Здесь s указывает, что экспресс движется в направлении из s в $\bar{s}$, тогда как b отражает ситуацию в разъезде и принимает следующие значения:

- 0, если экспресс проходит через пустой разъезд;
- 1, если экспресс является частью пакета и не является последним в этом пакете;
- 2, если экспресс является последним в пакете.

Пусть $i \in N_s^\alpha$ и $i' \in N_{s'}^{\alpha'}$ — два последовательных экспресса типов (s, b) и (s', b') соответственно. Пусть время отправления экспресса i равно t. Предположим, что $b \neq 1$, $b' \neq 0$ и некоторый неэкспресс $g' \in N_{s'}^\gamma$ пропускает i'. Согласно [4] время отправления экспресса i определяет минимально возможное время отправления неэкспресса g':

$$\hat{\tau} = \begin{cases} t + p_A + p_B + \beta, & \text{если } s = s'; \\ t + \beta, & \text{если } s \neq s',\ b = 0; \\ t + 2p_s + \beta, & \text{если } s \neq s',\ b = 2. \end{cases} \tag{5}$$

Действительно, если $s = s'$, то g' движется из $\bar{s}$ в s. В силу того что $b \neq 1$, поезда g' и i не могут быть активными одновременно. Следовательно, g' может отправиться из $\bar{s}$ только через β после прибытия поезда i, который прибывает в $\bar{s}$ в момент $t + p_A + p_B$. Если $s \neq s'$, то оба поезда, i и g', движутся из s в $\bar{s}$. Тогда если $b = 0$, то времена отправления поездов g' и i могут отличаться только на β, в то время как при $b = 2$ поезд g' может покинуть точку s только через β после прибытия в точку s поезда, который пропускает i в разъезде. Поезд g' покидает разъезд одновременно с i, т.е. в момент времени $t + p_s$, и после этого ему требуется p_s временны́х единиц, чтобы достичь s.

В отличие от [4], где предполагается, что все поезда одновременно находятся в соответствующих конечных точках пути, данная статья посвящена задаче корректировки расписания, которая учитывает временны́е окна, определенные для каждого поезда исходным расписанием. Таким образом, существует еще одно ограничение на самое раннее возможное время отправления, наложенное исходным временны́м окном поезда. Следовательно, самое раннее возможное время отправления поезда g' из соответствующей конечной точки рассматриваемого участка пути равно

$$\tau = \max\left\{\hat{\tau}, r_{g'}^{\bar{s}',\gamma}\right\}. \tag{6}$$

Если временно́е окно для поезда i' не учитывать, то согласно [4] времена отправлений t и τ (если поезд g' существует) определяют следующее самое

раннее возможное время отправления поезда i':

$$(7)\quad \hat{t} = \begin{cases} t+\beta, & \text{если } s = s' \text{ и } b = 1; \\ t+\beta, & \text{если } s = s' \text{ и } b = b' = 0; \\ \max\{t+2p_s+\beta, \tau + p_{\bar{s}'} + \beta - p_{s'}\}, & \text{если } s = s',\ b = 2,\ b' \neq 0; \\ t+2p_s+\beta, & \text{если } s = s',\ b = 2,\ b' = 0; \\ \tau + p_{\bar{s}'} + \beta - p_{s'}, & \text{если } s = s',\ b = 0,\ b' \neq 0; \\ t+p_A+p_B+\beta, & \text{если } s \neq s',\ b' = 0; \\ \max\{t+p_A+p_B+\beta, \tau + p_{\bar{s}'} + \beta - p_{s'}\}, & \text{если } s \neq s',\ b' \neq 0. \end{cases}$$

Принимая во внимание ограничение, накладываемое временны́м окном для поезда i', его самое раннее возможное время отправления равно

$$(8)\qquad \max\left\{\hat{t}, r_{i'}^{s',\alpha'}\right\}.$$

Пусть экспресс i типа (s,b) имеет самое раннее время отправления среди всех экспрессов. Пусть g — поезд, который отправляется раньше i. Тогда g является неэкспрессом. Это в силу условий (s3) и (s4) означает, что g отправляется из $\bar{s}$ и пропускает i в разъезде. Следовательно, i — первый поезд, отправляющийся из s. Кроме того, поскольку только один неэкспресс может пропускать экспресс в разъезде, g — первый поезд, отправляющийся из $\bar{s}$.

Для каждого $s \in \{A,B\}$ и для каждого $\alpha \in \{p,o\}$ определим n_s^α как мощность множества N_s^α и пронумеруем все поезда $j \in N_s^\alpha$ в порядке убывания $r_j^{s,\alpha}$ или, что эквивалентно, в порядке убывания $d_j^{s,\alpha}$. Пусть первый экспресс в последовательности экспрессов имеет тип (s,b) и является поездом категории α, который разъезжается с неэкспрессом категории γ (если такой существует). Тогда, принимая во внимание условие (s2), время отправления этого первого экспресса равно

$$(9)\qquad t = \begin{cases} r_{n_s^\alpha}^{s,\alpha}, & \text{если } b = 0; \\ \max\left\{r_{n_s^\alpha}^{s,\alpha}, r_{n_{\bar{s}}^\gamma}^{\bar{s},\gamma} + p_{\bar{s}} + \beta - p_s\right\}, & \text{если } b \neq 0. \end{cases}$$

Используя те же рассуждения, что и выше, легко видеть, что если экспресс i типа (s,b) имеет самое позднее время отправления среди всех экспрессов и g — поезд, который отправляется позже i, то i — последний поезд, отправляющийся из s, g — последний поезд, отправляющийся из $\bar{s}$, и g пропускает поезд i в разъезде.

Пусть i и i' — два последовательных экспресса типов (s,b) и (s',b'), соответственно, таких, что время отправления i' больше времени отправления i. Как было показано выше, типы этих экспрессов вместе со временем отправления i полностью определяют самое раннее возможное время отправления i'. Кроме того, если некоторый неэкспресс g пропускает i и $b = 2$, то тип (s,b) и время отправления i полностью определяют время, когда g прибывает в s; и если некоторый неэкспресс g' пропускает i' и $b \neq 1$, то время отправления i

и типы этих двух экспрессов полностью определяют самое раннее время отправления g' из $\bar{s}'$. Более того, как было показано выше, тип первого экспресса полностью определяет самое раннее время отправления этого экспресса и самое раннее время отправления поезда, с которым этот экспресс расходится в разъезде, если такой поезд существует. Эти наблюдения позволяют построить искомое расписание, учитывая только времена отправления экспрессов и назначая каждому поезду самое раннее возможное время отправления из соответствующей точки пути.

3. Минимизация максимального временно́го смещения

В данном разделе показано, как найти оптимальное значение целевой функции (2). Целевая функция (2) включает в себя только одну категорию поездов, и, следовательно, эта оптимизационная задача сходна с задачей, рассмотренной в [4], с одним существенным отличием — рассматриваемая здесь задача корректировки расписания учитывает временны́е окна, назначенные поездам до возникновения блокировки. Это требует включения времени в определение состояния, что в свою очередь меняет реализацию общей структуры динамического программирования для решения задачи минимизации максимального временно́го смещения.

Рассмотрим произвольное расписание (напомним, что данный раздел касается только приоритетных поездов, а существование всех обычных поездов игнорируется) и произвольный экспресс в этом расписании. Пусть (s, b) — тип этого экспресса и t — время его отправления. Количество приоритетных поездов, которые отправляются из s в $\bar{s}$ в момент времени t или после этого момента, будем обозначать через k_s. Пусть $k_{\bar{s}}$ — количество приоритетных поездов, каждый из которых является либо поездом, который пропускает рассматриваемый экспресс в разъезде, либо поездом, который отправляется из $\bar{s}$ в s в момент времени t или позже, либо поездом, для которого выполнены оба условия. Отправлению рассматриваемого экспресса соответствует набор (t, k_A, k_B, s, b). Согласно терминологии динамического программирования этот набор будет называться *состоянием*. Легко видеть, что если отправлению экспресса соответствует состояние (t, k_A, k_B, s, b), то данный экспресс принадлежит множеству N_s^p и в силу (s2) и введенной нумерации поездов имеет номер k_s.

Состояние (t, k_A, k_B, s, b) позволяет вычислить

$$C_{k_s}^{s,p} - d_{k_s}^{s,p} = t + p_A + p_B - d_{k_s}^{s,p}$$

и в случае $b = 2$

$$C_{k_{\bar{s}}}^{\bar{s},p} - d_{k_{\bar{s}}}^{\bar{s},p} = t + 2p_s - d_{k_{\bar{s}}}^{\bar{s},p}.$$

Введем обозначение

$$L(t, k_A, k_B, s, b) = \begin{cases} \max\left\{t + p_A + p_B - d_{k_s}^{s,p}, t + 2p_s - d_{k_{\bar{s}}}^{\bar{s},p}\right\}, & \text{если } b = 2; \\ t + p_A + p_B - d_{k_s}^{s,p} & \text{иначе.} \end{cases}$$

Каждое расписание индуцирует последовательность состояний, в которой состояния перечислены в порядке возрастания соответствующего времени отправления. Рассмотрим все расписания, в которых последовательности состояний содержат (t, k_A, k_B, s, b). Пусть $F(t, k_A, k_B, s, b)$ — минимальное значение величины

$$\max_{s\in\{A,B\}} \max_{j\in\{1,\ldots,k_s\}} [C_j^{s,p} - d_j^{s,p}]$$

на множестве всех таких расписаний. Обозначим через $\Omega(t, k_A, k_B, s, b)$ множество всех состояний, таких что каждое из них следует сразу за (t, k_A, k_B, s, b) по крайней мере в одной из упомянутых выше последовательностей.

Согласно разделу 2 последний экспресс также является последним поездом, проходящим железнодорожный путь в соответствующем направлении. Более того, после отправления этого экспресса максимум один поезд может двигаться в противоположном направлении, и этот поезд должен пропускать данный экспресс. Следовательно, если состояние (t, k_A, k_B, s, b) соответствует отправлению последнего экспресса, то k_x в (t, k_A, k_B, s, b) равен

$$k_x = \begin{cases} 1 & \text{для } x = s; \\ 0 & \text{для } x = \bar{s} \text{ и } b \neq 2; \\ 1 & \text{для } x = \bar{s} \text{ и } b = 2. \end{cases} \tag{10}$$

Это означает, что

$$F(t, k_A, k_B, s, b) = L(t, k_A, k_B, s, b). \tag{11}$$

Если в некотором расписании, у которого последовательность состояний содержит (t, k_A, k_B, s, b), экспресс $k_s \in N_s^p$ не является последним экспрессом в расписании, то он не является последним экспрессом во всех расписаниях, чьи последовательности состояний содержат (t, k_A, k_B, s, b). Поэтому

$$F(t, k_A, k_B, s, b) = \tag{12}$$
$$= \max\left\{ L(t, k_A, k_B, s, b), \min_{(t', \hat{k}_A, \hat{k}_B, s', b')\in\Omega(t, k_A, k_B, s, b)} F(t', \hat{k}_A, \hat{k}_B, s', b') \right\}.$$

Как уже отмечалось в разделе 2, первый экспресс также является первым поездом, отправляющимся в соответствующем направлении, и время его отправления можно вычислить с помощью (9). Пусть X — множество всех состояний, соответствующих всем возможным вариантам выбора первого экспресса и его типа. Тогда оптимальное значение функции (2) может быть записано в виде

$$\min_{(t, n_A^p, n_B^p, s, b)\in X} F(t, n_A^p, n_B^p, s, b). \tag{13}$$

Таким образом, принимая во внимание (11), (12) и (13), оптимальное значение (2) может быть получено с помощью динамического программирования. Более подробная информация по реализации вычислений будет представлена в разделе 5.

4. Минимизация суммарного времени нахождения в системе

В данном разделе рассматривается задача минимизации (3) на множестве всех расписаний, оптимальных для (2). Пусть F^* — оптимальное значение для (2). Любое расписание оптимально для (2) тогда и только тогда, когда

$$C_i^{s,p} \leqslant d_i^{s,p} + F^* \quad \text{для любого } s \in \{A, B\} \text{ и любого } i \in N_s^p. \tag{14}$$

Другими словами, необходимо найти расписание, доставляющее наименьшее значение для (3) среди всех расписаний, удовлетворяющих условию (14). Следовательно, в данном разделе будут рассматриваться только расписания, для которых выполнено (14).

В отличие от раздела 3, который касался только приоритетных поездов, теперь рассматриваются все поезда. Таким образом, с отправлением каждого экспресса связано больше информации, что ведет к соответствующему изменению определения состояния. Теперь состояние — это набор $(t, k_A^p, k_A^o, k_B^p, k_B^o, s, b, \alpha, \gamma)$, где, как и в разделе 3, t и пара (s, b) — время отправления и тип соответствующего экспресса. Кроме того, α — категория экспресса и γ — категория поезда (если такой поезд существует), который пропускает данный экспресс в разъезде. Если такой поезд не существует, то γ принимает любое значение из $\{p, o\}$. Независимо от того, пропускает ли какой-либо поезд данный экспресс или нет, $k_{\bar{s}}^{\gamma}$ — это количество поездов в подмножестве множества $N_{\bar{s}}^{\gamma}$, которое состоит из всех поездов, прибывающих в s после t. Каждое другое значение k_x^u в $(t, k_A^p, k_A^o, k_B^p, k_B^o, s, b, \alpha, \gamma)$ — это количество поездов, которые отправляются из x в момент времени t или позже. В частности, в силу (s2) состояние $(t, k_A^p, k_A^o, k_B^p, k_B^o, s, b, \alpha, \gamma)$ соответствует отправлению поезда, принадлежащего множеству N_s^{α}, номер которого равен k_s^{α}.

Если $\alpha = o$, то состояние $(t, k_A^p, k_A^o, k_B^p, k_B^o, s, b, \alpha, \gamma)$ предоставляет информацию для вычисления

$$C_{k_s}^{s,o} - r_{k_s}^{s,o} = t + p_A + p_B - r_{k_s}^{s,o},$$

а если $\gamma = o$ и $b = 2$, то это состояние позволяет также вычислить

$$C_{k_{\bar{s}}}^{\bar{s},o} - r_{k_{\bar{s}}}^{\bar{s},o} = t + 2p_s - r_{k_{\bar{s}}}^{\bar{s},o}.$$

Следовательно, вклад в значение целевой функции (3) экспресса, отправлению которого соответствует состояние $(t, k_A^p, k_A^o, k_B^p, k_B^o, s, b, \alpha, \gamma)$, и поезда, который пропускает данный экспресс в разъезде (если такой поезд существует), равен

$$R(t, k_A^p, k_A^o, k_B^p, k_B^o, s, b, \alpha, \gamma) =$$

$$= \begin{cases} 2t + 3p_s + p_{\bar{s}} - r_{k_s}^{s,o} - r_{k_{\bar{s}}}^{\bar{s},o} & \text{для } \alpha = o, b = 2, \gamma = o; \\ t + p_A + p_B - r_{k_s}^{s,o} & \text{для } \alpha = o, b = 2, \gamma \neq o; \\ t + p_A + p_B - r_{k_s}^{s,o} & \text{для } \alpha = o, b \neq 2; \\ t + 2p_s - r_{k_{\bar{s}}}^{\bar{s},o} & \text{для } \alpha \neq o, b = 2, \gamma = o; \\ 0 & \text{иначе.} \end{cases} \tag{15}$$

Основываясь на одних и тех же концепциях динамического программирования, процедуры оптимизации для (2) и (3) имеют много общего, несмотря на несколько важных отличий — разные целевые функции; условие (14); и разные множества поездов. Кроме того, оптимизационный подход, описанный в этом разделе, применяется только после процедуры минимизации (2). Поэтому использование далее тех же обозначений Ω и F не вызовет путаницы, а скорее подчеркнет сходство и облегчит изложение в разделе 5.

Рассмотрим все расписания, в которых индуцированные последовательности состояний содержат некоторое состояние $(t, k_A^p, k_A^o, k_B^p, k_B^o, s, b, \alpha, \gamma)$. Напомним, что в этом разделе рассматриваются только расписания, удовлетворяющие (14). Как и прежде, в каждой индуцированной последовательности состояний состояния перечислены в порядке возрастания соответствующих времен отправления. Обозначим через $\Omega(t, k_A^p, k_A^o, k_B^p, k_B^o, s, b, \alpha, \gamma)$ множество всех состояний таких, что каждое из них сразу следует за $(t, k_A^p, k_A^o, k_B^p, k_B^o, s, b, \alpha, \gamma)$ по крайней мере в одной из упомянутых выше последовательностей. Если $k_A^o + k_B^o > 0$, то определим $F(t, k_A^p, k_A^o, k_B^p, k_B^o, s, b, \alpha, \gamma)$ как минимальное значение

$$\sum_{s\in\{A,B\}} \sum_{j\leqslant k_s^o} [C_j^{s,o} - r_j^{s,o}] \tag{16}$$

на множестве рассматриваемых расписаний. Если $k_A^o + k_B^o = 0$, то положим

$$F(t, k_A^p, k_A^o, k_B^p, k_B^o, s, b, \alpha, \gamma) = 0.$$

Далее для любых $\alpha \in \{p, o\}$ удобно использовать обозначение $\bar{\alpha} = \{p, o\} \setminus \{\alpha\}$. Если состояние $(t, k_A^p, k_A^o, k_B^p, k_B^o, s, b, \alpha, \gamma)$ соответствует отправлению последнего экспресса, то согласно разделу 2 k_x^u в $(t, k_A^p, k_A^o, k_B^p, k_B^o, s, b, \alpha, \gamma)$ определяется как

$$k_x^u = \begin{cases} 1 & \text{для } x = s \text{ и } u = \alpha; \\ 0 & \text{для } x = s \text{ и } u = \bar{\alpha}; \\ 0 & \text{для } x = \bar{s} \text{ и } u = \bar{\gamma}; \\ 0 & \text{для } x = \bar{s}, u = \gamma \text{ и } b \neq 2; \\ 1 & \text{для } x = \bar{s}, u = \gamma \text{ и } b = 2. \end{cases} \tag{17}$$

Это означает, что

$$F(t, k_A^p, k_A^o, k_B^p, k_B^o, s, b, \alpha, \gamma) = R(t, k_A^p, k_A^o, k_B^p, k_B^o, s, b, \alpha, \gamma). \tag{18}$$

Если экспресс, которому соответствует состояние $(t, k_A^p, k_A^o, k_B^p, k_B^o, s, b, \alpha, \gamma)$, не является последним экспрессом в расписании, то

$$F(t, k_A^p, k_A^o, k_B^p, k_B^o, s, b, \alpha, \gamma) = R(t, k_A^p, k_A^o, k_B^p, k_B^o, s, b, \alpha, \gamma) + \tag{19}$$

$$+ \min_{(t', \hat{k}_A^p, \hat{k}_A^o, \hat{k}_B^p, \hat{k}_B^o, s', b', \alpha', \gamma') \in \Omega(t, k_A^p, k_A^o, k_B^p, k_B^o, s, b, \alpha, \gamma)} F(t', \hat{k}_A^p, \hat{k}_A^o, \hat{k}_B^p, \hat{k}_B^o, s', b', \alpha', \gamma').$$

Обозначим через Y множество всех состояний, которые являются первыми по крайней мере в одной последовательности состояний, индуцированной расписанием, удовлетворяющим (14). Тогда минимальное значение (3) на множестве всех расписаний, оптимальных для (2), равно

$$\min_{(t,n_A^p,n_A^o,n_B^p,n_B^o,s,b,\alpha,\gamma)\in Y} F(t,n_A^p,n_A^o,n_B^p,n_B^o,s,b,\alpha,\gamma). \tag{20}$$

Заметим, что (13) и (20) существенно различны. Действительно, чтобы перечислить все состояния в X, достаточно знать только n_A^p, n_A^p, $r_{n_A^p}^{A,p}$ и $r_{n_B^p}^{B,p}$, в то время как перечисление всех состояний в Y может потребовать значительно более сложную вычислительную процедуру. Эта процедура представлена в разделе 5.

5. Динамическое программирование

Для приведенной ниже оптимизационной процедуры требуется множество T моментов времени, которое содержит все времена отправления экспрессов во всех расписаниях, в которых каждый экспресс отправляется в самый ранний возможный момент времени для экспресса этого типа. Такое множество описано ниже.

Принимая во внимание (5)–(9), легко видеть, что время отправления любого экспресса представимо в виде

$$t = r_i^{s,\alpha} + m_1 p_A + m_2 p_B + m_3 \beta \tag{21}$$

для некоторого $\alpha \in \{p, o\}$, $s \in \{A, B\}$, $i \in N_s^\alpha$, и некоторых целых чисел m_1, m_2 и m_3. Введем обозначение

$$n = n_A^p + n_B^p + n_A^o + n_B^o.$$

Отметим, что s и α в выражении (21) не обязательно совпадают с соответствующими параметрами рассматриваемого экспресса. Если для двух последовательных экспрессов α, s и i в выражении (21) остаются теми же самыми, то согласно (7) и (9) m_1 и m_2 не могут увеличиться больше чем на 3 и не могут уменьшиться больше чем на 1, тогда как m_3 не может увеличиться больше чем на 2. Таким образом, m_1 и m_2 не превосходят $3n$ и не меньше $(-n)$, а m_3 не превосходит $2n$. Следовательно, все возможные моменты отправления экспресса принадлежат множеству

$$\Big\{ t |\ t \geqslant 0,\ t = r_i^{s,\alpha} + m_1 p_A + m_2 p_B + m_3 \beta,$$
$$i \in N_s^\alpha,\ s \in \{A, B\},\ \alpha \in \{p, o\},\ m_1 \in \{-n, \ldots, 0, 1, \ldots, 3n\}, \tag{22}$$
$$m_2 \in \{-n, \ldots, 0, 1, \ldots, 3n\}, m_3 \in \{0, 1, \ldots, 2n\} \Big\}$$

мощности $O(n^4)$.

Рассмотрим задачу минимизации (2) и предположим, что $n_A^p > 0$ и $n_B^p > 0$, поскольку в противном случае минимизация (2) тривиальна. Поскольку данная задача включает только приоритетные поезда, то, следуя подходу, принятому в разделе 3, можно рассматривать только приоритетные поезда. Следовательно, вместо (22) можно использовать в качестве T его подмножество

$$\begin{gathered}\Big\{t|\ t \geqslant 0,\ t = r_i^{s,p} + m_1 p_A + m_2 p_B + m_3 \beta,\ i \in N_s^p,\ s \in \{A, B\},\\ m_1 \in \{-(n_A^p + n_B^p), \ldots, 0, 1, \ldots, 3(n_A^p + n_B^p)\},\\ m_2 \in \{-(n_A^p + n_B^p), \ldots, 0, 1, \ldots, 3(n_A^p + n_B^p)\},\\ m_3 \in \{0, 1, \ldots, 2(n_A^p + n_B^p)\}\Big\}.\end{gathered} \tag{23}$$

Оптимизационная процедура, представленная в данном разделе, получает оптимальное значение для (2) путем генерации последовательности множеств $V_1, \ldots, V_{n_A^p + n_B^p}$. Следовательно, количество множеств в этой последовательности равно количеству поездов, поскольку, как и в разделе 3, рассматриваются только приоритетные поезда. Каждое множество состоит из наборов (t, k_A, k_B, s, b), которые являются кандидатами на состояния в оптимальном для (2) расписании.

В каждом множестве V_k, $1 \leqslant k \leqslant n_A^p + n_B^p$, содержатся только кандидаты, удовлетворяющие условию

$$k_A^p + k_B^p = k.$$

Таким образом, ввиду (10) все кандидаты на состояние, соответствующее последнему экспрессу, могут быть только в множествах V_1 и V_2. Множество V_1 содержит только таких кандидатов. Более точно, это множество состоит из всех наборов вида $(t, 1, 0, A, 0)$, где $t \in T$ и $t \geqslant r_1^{A,p}$, и всех наборов вида $(t, 0, 1, B, 0)$, где $t \in T$ и $t \geqslant r_1^{B,p}$.

Множество V_2 содержит всех кандидатов на состояние, соответствующее последнему экспрессу и удовлетворяющее равенству $k_A + k_B = 2$, а также других кандидатов, удовлетворяющих этому равенству. Подмножество V_2 всех кандидатов на последнее состояние состоит из всех наборов типа $(t, 1, 1, A, 2)$, где $t \in T$ и $t \geqslant r_1^{A,p}$, и всех наборов типа $(t, 1, 1, B, 2)$, где $t \in T$ и $t \geqslant r_1^{B,p}$.

Множества $V_1, \ldots, V_{n_A^p + n_B^p}$ генерируются одно за другим в порядке возрастания их индексов. После включения всех кандидатов на состояние, соответствующее последнему экспрессу, анализируются наборы для каждого множества один за другим. Чтобы быть включенным в множество V_k, набор (t, k_A, k_B, s, b), для которого выполняется $k_A + k_B = k$, должен обладать несколькими свойствами. Эти свойства включают:

(p1) $k_s \leqslant n_s^{s,p}$ для $s \in \{A, B\}$;

(p2) $r_{k_s}^{s,p} \leqslant t$ для $t \in T$;

(p3) если $b \neq 0$, то $k_{\bar{s}} \geqslant 1$;

(p4) если $b = 1$, то $k_s \geqslant 2$;

(p5) если $k_A^p = n_A^p$ и $k_B^p = n_B^p$, то t вычисляется по (9).

Более того, рассматриваемый набор (t, k_A, k_B, s, b), удовлетворяющий свойствам (p1)–(p5), включается в множество $V_{k_A+k_B}$, если только существует набор $(t', \hat{k}_A, \hat{k}_B, s', b')$, который включен в одно из ранее сгенерированных множеств, и имеет t', которое можно вычислить по формулам (5)–(8); имеет $\hat{k}_A$ и $\hat{k}_B$, удовлетворяющие (24) и (25), приведённым ниже; и удовлетворяет условию (26), приведённому ниже:

$$\hat{k}_s = k_s - 1, \tag{24}$$

$$\hat{k}_{\bar{s}} = \begin{cases} k_{\bar{s}} - 1, & \text{если } b = 2, \\ k_{\bar{s}} & \text{иначе}, \end{cases} \tag{25}$$

$$\text{если } b = 1, \text{ то } s = s' \text{ и } b' \neq 0. \tag{26}$$

Для каждого набора (t, k_A, k_B, s, b), обладающего требуемыми свойствами и, следовательно, принадлежащего $V_{k_A+k_B}$, обозначим через $W(t, k_A, k_B, s, b)$ множество всех наборов $(t', \hat{k}_A, \hat{k}_B, s', b')$ в ранее сгенерированных множествах таких, что t' совпадает с моментом времени, вычисленным по формуле (8); $\hat{k}_A$ и $\hat{k}_B$ удовлетворяют (24) и (25); и выполняется (26). Если (t, k_A, k_B, s, b) выбирается в качестве состояния, то

$$W(t, k_A, k_B, s, b) = \Omega(t, k_A, k_B, s, b).$$

Тогда с учетом (13) оптимальное значение (2) равно

$$\min_{(t, n_A, n_B, s, b) \in V_{n_A^p + n_B^p}} f(t, k_A, k_B, s, b),$$

где f определяется подобно F в (11) и (12), т.е. для каждого кандидата (t, k_A, k_B, s, b) на состояние, соответствующее последнему экспрессу,

$$f(t, k_A, k_B, s, b) = L(t, k_A, k_B, s, b), \tag{27}$$

и для любого другого (t, k_A, k_B, s, b) в $V_2 \cup \ldots \cup V_{n_A^p + n_B^p}$

$$f(t, k_A, k_B, s, b) = \tag{28}$$
$$= \max \left\{ L(t, k_A, k_B, s, b), \min_{(t', \hat{k}_A, \hat{k}_B, s', b') \in W(t, k_A, k_B, s, b)} f(t', \hat{k}_A, \hat{k}_B, s', b') \right\}.$$

Учитывая мощность множества T (см. (23)), сложность данной оптимизационной процедуры составляет $O((n_A^p + n_B^p)^6)$.

Процедура построения расписания, доставляющего минимальное значение для (3) среди всех расписаний, оптимальных для (2), сходна с описанной процедурой минимизации (2) с одним важным отличием: чтобы гарантировать (14), каждый набор должен удовлетворять дополнительным условиям (см. (g6) и (g7) ниже). Согласно этой процедуре n множеств $V_1, \ldots, V_n$

генерируются одно за другим в порядке возрастания их индексов. Аналогично описанному выше множество V_k, $1 \leqslant k \leqslant n$, содержит только наборы $(t, k_A^p, k_A^o, k_B^p, k_B^o, s, b, \alpha, \gamma)$, удовлетворяющие условию

$$k_A^p + k_A^o + k_B^p + k_B^o = k.$$

Каждый набор в множествах $V_1, \ldots, V_n$ является кандидатом на состояние в искомом расписании. Чтобы быть включенным в множество, набор $(t, k_A^p, k_A^o, k_B^p, k_B^o, s, b, \alpha, \gamma)$ должен обладать несколькими свойствами. Первая группа свойств:

(g1) $k_s^\varepsilon \leqslant n_s^{s,\varepsilon}$ для $\varepsilon \in \{p, o\}$ и $s \in \{A, B\}$;

(g2) $r_{k_s}^{s,\alpha} \leqslant t$ и $t \in T$;

(g3) если $b \neq 0$, то $k_{\bar{s}}^\gamma \geqslant 1$;

(g4) если $b = 1$, то $k_s^\alpha + k_s^{\bar{\alpha}} \geqslant 2$;

(g5) если $k_A^p = n_A^p$, $k_A^o = n_A^o$, $k_B^p = n_B^p$, и $k_B^o = n_B^o$, то t вычисляется по (9).

Эти свойства аналогичны (p1)–(p5). Вторая группа свойств:

(g6) если $\alpha = p$, то $t + p_A + p_B \leqslant d_{k_s^p}^{s,p} + F^*$;

(g7) если $\gamma = p$ и $b = 2$, то $t + 2p_s \leqslant d_{k_{\bar{s}}^p}^{\bar{s},p} + F^*$

— гарантирует, что полученное расписание будет удовлетворять (14).

Для того чтобы $(t, k_A^p, k_A^o, k_B^p, k_B^o, s, b, \alpha, \gamma)$ был выбран в качестве кандидата на состояние, должен существовать уже выбранный набор $(t', \hat{k}_A^p, \hat{k}_A^o, \hat{k}_B^p, \hat{k}_B^o, s', b', \alpha', \gamma')$ такой, что

$$\hat{k}_s^\alpha = k_s^\alpha - 1, \quad \hat{k}_s^{\bar{\alpha}} = k_s^{\bar{\alpha}} \quad \text{и} \quad \hat{k}_{\bar{s}}^{\bar{\gamma}} = k_{\bar{s}}^{\bar{\gamma}}, \tag{29}$$

$$\hat{k}_{\bar{s}}^\gamma = \begin{cases} k_{\bar{s}}^\gamma - 1, & \text{если } b = 2, \\ k_{\bar{s}}^\gamma & \text{иначе.} \end{cases} \tag{30}$$

Равенства (29) и (30) выполняют ту же роль, что (24) и (25) ранее. Более точно, набор $(t, k_A^p, k_A^o, k_B^p, k_B^o, s, b, \alpha, \gamma)$, удовлетворяющий (g1)–(g7), включается в множество $V_{k_A^p + k_A^o + k_B^p + k_B^o}$, если только существует набор $(t', \hat{k}_A^p, \hat{k}_A^o, \hat{k}_B^p, \hat{k}_B^o, s', b', \alpha', \gamma')$, который уже включен в одно из ранее сгенерированных множеств и для которого t' совпадает с моментом времени, вычисленным по формулам (5)–(8); $\hat{k}_A^p$, $\hat{k}_A^o$, $\hat{k}_B^p$, $\hat{k}_B^o$ удовлетворяют (29) и (30); и выполняется (26). Обозначим через $W(t, k_A^p, k_A^o, k_B^p, k_B^o, s, b, \alpha, \gamma)$ множество всех таких наборов $(t', \hat{k}_A^p, \hat{k}_A^o, \hat{k}_B^p, \hat{k}_B^o, s', b', \alpha', \gamma')$. Тогда минимальное значение (3) на множестве всех расписаний, оптимальных для (2), может быть записано как

$$\min_{(t, k_A^p, k_A^o, k_B^p, k_B^o, s, b, \alpha, \gamma) \in V_n} f(t, k_A^p, k_A^o, k_B^p, k_B^o, s, b, \alpha, \gamma),$$

где f определяется следующим образом.

Для каждого кандидата $(t, k_A^p, k_A^o, k_B^p, k_B^o, s, b, \alpha, \gamma)$ на состояние, соответствующее последнему экспрессу,

$$f(t, k_A^p, k_A^o, k_B^p, k_B^o, s, b, \alpha, \gamma) = R(t, k_A^p, k_A^o, k_B^p, k_B^o, s, b, \alpha, \gamma).$$

Для каждого элемента $V_2 \cup \ldots \cup V_n$, который не является кандидатом на состояние, соответствующее последнему экспрессу,

$$f(t, k_A^p, k_A^o, k_B^p, k_B^o, s, b, \alpha, \gamma) = R(t, k_A^p, k_A^o, k_B^p, k_B^o, s, b, \alpha, \gamma) +$$
$$+ \min_{(t', \hat{k}_A^p, \hat{k}_A^o, \hat{k}_B^p, \hat{k}_B^o, s', b', \alpha', \gamma') \in W(t, k_A^p, k_A^o, k_B^p, k_B^o, s, b, \alpha, \gamma)} f(t', \hat{k}_A^p, \hat{k}_A^o, \hat{k}_B^p, \hat{k}_B^o, s', b', \alpha', \gamma').$$

Принимая во внимание мощность множества T (см. (22)), данная оптимизационная процедура имеет временну́ю сложность $O(n^8)$.

6. Заключение

Полиномиальные алгоритмы, представленные в статье, направлены на уменьшение прямого воздействия блокировки одного из путей на двухпутном участке дороги, измеряемого максимальным временны́м смещением приоритетных поездов и общим (а значит, и средним) временем нахождения в системе для обычных поездов. Направления дальнейшего обобщения представленного подхода могут включать минимизацию других мер влияния блокировки; уменьшение влияния блокировки на более широкие участки железнодорожной сети; планирование работ по техническому обслуживанию, требующих вре́менного закрытия некоторых сегментов железнодорожной сети, и разработка быстрых аппроксимационных алгоритмов.

СПИСОК ЛИТЕРАТУРЫ

1. *Cacchiani V., Huisman D., Kidd M., Kroon L., Toth P., Veelenturf L., Wagenaar J.* An Overview of Recovery Models and Algorithms for Real-Time Railway Rescheduling // Transport. Res. Part B. 2014. V. 63. P. 15–37.
2. *Lusby R., Larsen J., Ehrgott M., Ryan D.* Railway Track Allocation: Models and Methods // OR Spectrum. 2011. V. 33. No. 4. P. 843–883.
3. *Brucker P., Heitmann S., Knust S.* Scheduling Railway Traffic at a Construction Site // OR Spectrum. 2002. V. 24. No. 1. P. 19–30.
4. *Зиндер Я., Лазарев А.А., Мусатова Е.Г., Тарасов И.А.* Построение расписаний двухстороннего движения на однопутной железной дороге с разъездом // АиТ. 2018. № 3. С. 144–166.

   Zinder Y., Lazarev A.A., Musatova E.G., Taracov I.A. Scheduling the Two-Way Traffic on a Single-Track Railway With a Siding // Autom. Remote Control. 2018. V. 79. No. 3. P. 506–523.
5. *Zinder Y., Lazarev A., Musatova E.,Taracov I., Khusnullin N.* Two-Station Single Track Scheduling Problem // IFAC-PapersOnLine. 2016. V. 49. No. 12. P. 231–236.

Статья представлена к публикации членом редколлегии Ф.Т. Алескеровым.

Поступила в редакцию 02.07.2019

После доработки 15.10.2019

Принята к публикации 28.11.2019

© 2020 г. С.А. МАЛАХ (malahsveta@mail.ru),
В.В. СЕРВАХ, д-р физ.-мат. наук (svv_usa@rambler.ru)
(Институт математики им. С.Л. Соболева СО РАН, Омск)

МАКСИМИЗАЦИЯ УДЕЛЬНОЙ ПРИВЕДЕННОЙ ПРИБЫЛИ В СИСТЕМАХ УПРАВЛЕНИЯ ЗАПАСАМИ[1]

Исследуется модель максимизации прибыли коммерческой компании с учетом интенсивности продажи товаров, стоимости закупки, доставки, хранения и реализации, а также возможности альтернативного размещения свободного капитала. Показано, что функция прибыли в зависимости от периода завоза товаров имеет единственную точку максимума. Построена модель и разработаны алгоритмы решения задачи максимизации прибыли в многономенклатурных системах при ограниченном оборотном капитале.

Ключевые слова: исследование операций, управление запасами, максимизация прибыли, динамическое программирование.

DOI: 10.31857/S0005231020050074

1. Введение

Рассматривается задача оптимизации деятельности торговой компании, которая закупает товары на бирже и реализует их на розничном рынке. На предприятии построен специализированный склад, внедрены современные логистические технологии с соответствующими базами данных, постоянным мониторингом движения товаров, автоматизацией склада и погрузочно-разгрузочных работ. Новейшие технологии, эффективная организация собственного склада позволили существенно сократить издержки. Сложилась ситуация, когда классические модели, например [1–5], основанные на минимизации затрат, не всегда адекватно отражают ситуацию. В настоящее время с учетом высокой мобильности экономики, острой конкуренции и внедрения современных систем логистики возникла необходимость использования новых моделей, направленных, в первую очередь, на эффективное использование капитала и получение максимальной прибыли. Одной из первых в этом направлении была статья [6]. Из российских публикаций отметим [7–11] и недавно изданную монографию [12]. Такой подход особенно важен при оптимизации закупок товаров, для которых отношение объем/цена является малой величиной, например радиодеталей, медикаментов и др. [13, 14]. Затраты на доставку и хранение становятся незначительными, и большее значение приобретает скорость оборота денег.

[1] Работа выполнена при поддержке программы фундаментальных научных исследований государственных академий наук на 2013–2020 гг., п. I.5, проект № 0314-2019-0019 «Анализ и решение задач проектирования сложных систем методами дискретной оптимизации».

Другой проблемой является разработка алгоритма автоматической корректировки заявок при ограничении оборотного капитала. Автоматизированная система при формировании очередной заявки определяет оптимальные объемы закупки товаров. В некоторые моменты времени общая стоимость заказываемых товаров может оказаться больше имеющихся в наличии свободных денег. В такой ситуации приходится либо сокращать заказ, либо брать кредит. Кредит меняет стоимость денег, и тем самым меняются и оптимальные объемы закупок. Сокращение заказа также приводит к отклонению от оптимума. Возникает задача корректировки многономенклатурного заказа с учетом ограничений на размер оборотного капитала.

Большое влияние на представляемую работу оказали успехи ученых белорусской школы по комбинаторной оптимизации, созданной В.С. Танаевым. Работы [15, 16] и многие другие стали источником получения качественной систематизированной информации, базой для дальнейших исследований. Работы по логистике поставок [19, 20] выполнялись в рамках проектов INTAS в тесном сотрудничестве с белорусскими учеными. Настоящая публикация опирается на современные разработки, проводимые в объединенном институте проблем информатики Национальной академии наук Беларуси. В частности, отметим статьи, связанные с задачами максимизации прибыли [17, 18].

Структура статьи. В разделе 2 рассматривается модель управления запасами при наличии альтернативных возможностей использования капитала. Раздел 3 посвящен построению модели и разработке алгоритмов решения задачи оптимизации текущих закупок в многономенклатурных системах при ограниченном оборотном капитале, в том числе и при возможности использования кредитов. В раздел 4 рассматривается более общая модель с учетом особенностей ее реализации на практике.

2. Задача максимизации чистой приведенной прибыли

Рассматривается классическая модель управления запасами и ее развитие с учетом факторов современной экономики. Основное внимание уделяется эффективному использованию капитала при наличии альтернативных возможностей его использования.

Модель будем рассматривать в предположении, что имеется гарантированная возможность альтернативного безрискового размещения капитала под ставку r_0. Тогда деньги в разные моменты времени будут иметь различную ценность. Чтобы сравнивать поступления, полученные в разное время, используется следующий подход. Если в момент t_1 имеется некоторый капитал K_1, то его всегда можно разместить на рынке под текущую рыночную процентную ставку r_0. При таком размещении к моменту t_2 капитал увеличится до значения $K_1(1+r_0)^{t_2-t_1}$. И наоборот, если в момент t_2 необходим капитал K_2, то в момент t_1 достаточно иметь его в количестве $K_2/(1+r_0)^{t_2-t_1}$. Поэтому чтобы сравнить капитал K_1 в момент времени t_1 и K_2 в момент времени t_2, необходимо сравнить величины K_1 и $K_2/(1+r_0)^{t_2-t_1}$. Эта операция называется операцией приведения к моменту времени t_1 или дисконтированием.

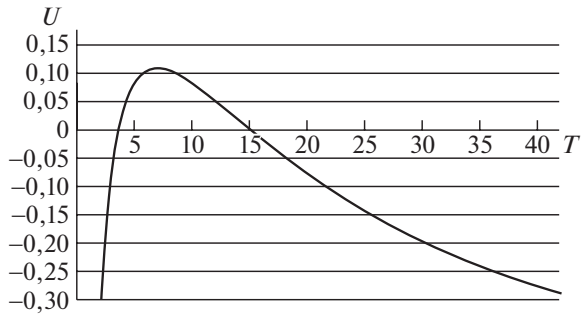


Рис. 1. Зависимость удельной прибыли от T.

Процесс закупки и реализации товара будем рассматривать, как в классической модели: закупаем товар в объеме v по цене β и продаем его с интенсивностью λ по цене c. Затраты на доставку задаются функцией $\alpha + \beta v$, где α — постоянные издержки, включающие стоимость заявки и доставки продукции. Стоимость хранения единицы товара в единицу времени обозначим через $c_{\text{хр}}$.

Время реализации закупленного товара составит $T = \frac{v}{\lambda}$. Интенсивность поступления денег от продажи равна $c\lambda$. С учетом дисконтирования интенсивность поступления денег выражается функцией $\frac{c\lambda}{(1+r_0)^t}$. За период реализации товара $[0, T]$ суммарные поступления, дисконтированные к начальному моменту времени, будут равны

$$Q(T) = \int\limits_0^T \frac{c\lambda}{(1+r_0)^t} dt = \frac{c\lambda}{ln(1+r_0)} \frac{(1+r_0)^{\mathrm{T}} - 1}{(1+r_0)^{\mathrm{T}}},$$

а затраты на хранение товара с учетом дисконтирования составят

$$Z(T) = \int\limits_0^T \frac{(T\lambda - t\lambda)c_{\text{хр}}}{(1+r_0)^t} dt = \frac{c_{\text{хр}}\lambda}{ln^2(1+r_0)} \left(T \ln(1+r_0) + \frac{1}{(1+r_0)^{\mathrm{T}}} - 1 \right).$$

Таким образом, за указанный период $[0, T]$ чистая приведенная прибыль будет получена в размере

$$Q(T) - (\alpha + \beta v) - Z(T).$$

В задаче требуется максимизировать удельную прибыль, которая выражается следующей функцией:

$$U(T) = \frac{Q(T) - (\alpha + \beta T\lambda) - Z(T)}{T} \to \max.$$

На графике (рис. 1) приведен пример функции $U(T)$ в зависимости от периода завоза товара при входных данных $\lambda = 1$, $\alpha = 2$, $\beta = 0{,}25$, $c = 1$, $r_0 = 0{,}1$, $c_{\text{хр}} = 0{,}03$. Оптимальное значение удельной чистой привиденной прибыли равно 0,109 и достигается при $T = 7{,}02$.

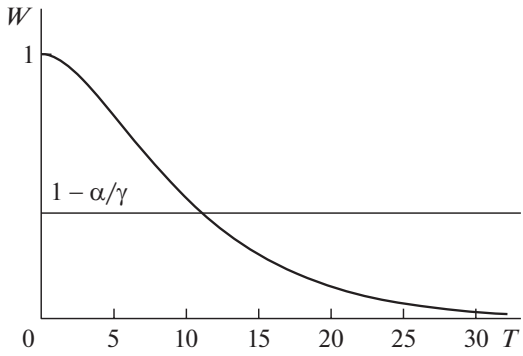


Рис. 2. Изменение функции $W(T)$ в зависимости от T.

Исследуем экстремальные значения функции $U(T)$. Обозначим

$$\gamma = \frac{c\lambda}{ln(1+r_0)} + \frac{c_{\text{хр}}\lambda}{ln^2(1+r_0)} > 0, \quad \delta = \frac{1}{1+r_0}, \quad 0 < \delta < 1.$$

Тогда

$$U(T) = \frac{1}{T}(\gamma(1-\delta^{\text{T}}) - \alpha - \beta T\lambda) - \text{const},$$

где $\text{const} = \frac{c_{\text{хр}}\lambda}{ln^2(1+r_0)}$,

$$\lim_{T\to+0} U(T) = -\infty, \quad \lim_{T\to+\infty} U(T) = -\beta\lambda.$$

Исследуем нули производной

$$U'(T) = \frac{-\gamma\delta^{\text{T}}\ln\delta T - \gamma(1-\delta^{\text{T}}) + \alpha}{T^2} = 0.$$

Получаем следующее уравнение:

$$\delta^{\text{T}}(1 - T\ln\delta) = 1 - \frac{\alpha}{\gamma}.$$

Решим его графически. Обозначим $W(T) = \delta^{\text{T}}(1 - T\ln\delta)$.

$$W(0) = \delta^0 = 1;$$

$$W'(T) = \delta^{\text{T}}\ln\delta - \delta^{\text{T}}T\ln^2\delta - \delta^{\text{T}}\ln\delta = -\delta^{\text{T}}T\ln^2\delta < 0;$$

$$\lim_{T\to\infty} W(T) = 0.$$

Следовательно, функция $W(T)$ всегда положительна и монотонно убывает. График ее изображен на рис. 2. Графиком правой части уравнения является прямая, параллельная оси Ox.

При $\alpha < \gamma$ графики пересекаются и существует единственное решение уравнения $\delta^{\text{T}}(1 - T\ln\delta) = 1 - \frac{\alpha}{\gamma}$. Очевидно, что это точка максимума. Тем самым доказана следующая

Теорема 1. При $\alpha < \frac{c\lambda}{ln(1+r_0)} + \frac{c_{\text{хр}}\lambda}{ln^2(1+r_0)}$ *функция прибыли имеет единственную точку максимума. При* $\alpha \geqslant \frac{c\lambda}{ln(1+r_0)} + \frac{c_{\text{хр}}\lambda}{ln^2(1+r_0)}$ *функция прибыли монотонно возрастает на всем промежутке* $(0, \infty)$ *и при этом значение прибыли всегда отрицательно и меньше величины* $-\beta\lambda - \frac{c_{\text{хр}}\lambda}{\ln(1+r_0)}$.

Таким образом, при определенных условиях существует такой период завоза продукции, при котором вложенные деньги используются максимально эффективно, принося наибольшую прибыль. Точка максимума находится как единственное решение уравнения $\delta^{\mathrm{T}}(1 + T\ln\delta) = 1 - \frac{\alpha}{\gamma}$.

3. Задача с ограничением на оборотный капитал

Описанный выше подход является математической основой моделей, реализованных на практике в сетевых торговых структурах. В многономенклатурных системах определяется, какие товары завозить и в каком объеме с учетом текущего спроса, цен закупки и продажи, стоимости доставки и хранения. Критерием является максимизация удельной приведенной прибыли. Используемая модель позволяет учитывать различные дополнительные ограничения по страховым запасам товаров, временным лагам и т.д. Сроки и объемы заказов рассчитываются автоматически программой-роботом без участия работников фирмы.

Сложности возникают, когда при формировании заявки общая стоимость заказываемых товаров может оказаться больше имеющихся в наличии свободных денег. Во-первых, стоимость заявки колеблется, так как в разные моменты времени завозятся разные виды и объемы товаров. Во-вторых, размер оборотного капитала может сократиться, например в период уплаты налогов или при внешнем отвлечении оборотных средств. В такой ситуации приходится либо сокращать заказ, либо брать кредит. Сокращение заказа приводит к дополнительным издержкам. Кредит уменьшает прибыль из-за выплаты процентов. Ниже построена модель и разработан алгоритм решения задачи минимизации издержек в случае ограничения на объем оборотного капитала.

Опишем параметры задачи:

N — количество видов товара;

α_i — стоимость доставки одной партии товара i;

β_i — цена закупки единицы продукции товара i;

$\alpha_i + \beta_i v_i$ — стоимость заказа и доставки партии товара i объемом v_i;

λ_i — интенсивность продажи товара i;

c_i — цена продажи единицы товара i;

$c_i^{\text{хр}}$ — стоимость хранения единицы товара i в единицу времени;

r_0 — ставка альтернативного безрискового ликвидного размещения капитала.

В текущий момент времени для каждого товара i с нулевым остатком на складе требуется найти период завоза T_i, при котором удельная приведенная прибыль $U_i(T_i)$, равная

$$\frac{1}{T_i}\Big(\frac{c_i\lambda_i}{ln(1+r_0)}\frac{(1+r_0)^{T_i}-1}{(1+r_0)^{T_i}} - \beta_i T_i\lambda_i - \\ -\frac{c_i^{\text{xp}}\lambda_i}{ln^2(1+r_0)}\left(T_i\ln(1+r_0)+\frac{1}{(1+r_0)^{T_i}}-1\right)\Big),$$

будет максимальна.

Отметим, что параметры задачи постоянны только на определенном временном промежутке. При следующем заказе товаров используются новые, актуальные на момент расчета значения параметров. Сделаем еще некоторые естественные допущения, которые не влияют на общность рассматриваемой модели. Заказы невозможно делать в любой момент времени. Дискретность планирования во времени естественна для экономических задач. Если выбрать подходящую единицу измерения, то можно рассматривать только целочисленные моменты времени. Временные лаги при отгрузке и доставке товара учитывать не будем, так как это не влияет на суть задачи.

При этих условиях достаточно рассматривать только такие объемы завоза, чтобы товар заканчивался в целочисленные моменты времени, и новый завоз делать, когда остаток товара будет равен нулю. Тогда и значения T_i должны быть целыми. Действительно, если T_i не является целым, то в момент времени $\lfloor T_i\rfloor$ все равно придется делать очередной завоз, так как остатка $(T_i-\lfloor T_i\rfloor)\lambda_i$ до следующего завоза не хватит. Целочисленный оптимум $U_i(T_i)$ обозначим через T_i^*. Соответствующее значение объема завозимого товара равно $v_i^*=T_i^*\lambda_i$.

Очередная заявка формируется в текущий момент $t\in Z^+$. Закупаем только те товары, которые к этому моменту заканчиваются. Без ограничения общности для удобства изложения перенумеруем их числами от 1 до n. Считываем из базы данных текущие параметры по этим позициям, далее по каждому товару $i=1,\ldots,n$ находим целочисленные оптимумы T_i^* и соответствующие значения завозимых объемов $v_i^*=T_i^*\lambda_i$. Для их покупки потребуется

$$\sum_{i=1}^{n}(\alpha_i+\beta_i v_i^*)$$

единиц капитала. При наличии необходимой суммы заявка оплачивается и товар поставляется.

3.1. Задача сокращения заявки

Проблемы возникают, когда текущего капитала K не хватает для оплаты заявки целиком. На практике рассматриваются три варианта: сокращение заявки, банковский краткосрочный кредит, товарный кредит. Последние два варианта отличаются организационно, однако с позиций максимизации чистой приведенной прибыли между ними различий нет, часть дохода приходится отдавать. Рассмотрим сначала задачу сокращения заявки, а потом обобщим ее на случай возможности использования кредитов.

Введем переменную $x_i\leqslant v_i^*$ — объем заказа товара $i=1,\ldots,n$. Как было упомянуто выше, T_i является целым числом, и, значит, $x_i\in\{\lambda_i,2\lambda_i,3\lambda_i,\ldots$

$\ldots, v_i^*\}$. Здесь $x_i \geqslant \lambda_i$, так как спрос должен быть удовлетворен. Сокращение заказа до уровня x_i приводит к потере прибыли в размере $H_i(x_i) = U_i\left(\frac{v_i^*}{\lambda_i}\right) - -U_i\left(\frac{x_i}{\lambda_i}\right)$. Отметим, что на множестве $\{\lambda_i, 2\lambda_i, 3\lambda_i, \ldots, v_i^*\}$ функция $H_i(x_i)$ монотонно убывает.

Таким образам, если $K < \sum\limits_{i=1}^{n} (\alpha_i + \beta_i v_i^*)$, то денег на оплату текущей заявки не хватает и закупки необходимо сократить. При этом желательно, чтобы потери прибыли были минимальны. Получаем следующую модель:

$$\sum_{i=1}^{n} H_i(x_i) \to \min,$$
$$\sum_{i=1}^{n} (\alpha_i + \beta_i x_i) \leqslant K,$$
$$x_i \in \{\lambda_i, 2\lambda_i, 3\lambda_i, \ldots, v_i^*\},\ i = 1, \ldots, n.$$

Заметим, что в данной постановке $K \geqslant \sum\limits_{i=1}^{n} (\alpha_i + \beta_i \lambda_i)$, так как до следующего завоза спрос должен быть удовлетворен. Величину $\sum\limits_{i=1}^{n} (\alpha_i + \beta_i \lambda_i)$ обозначим через $K_{\min}$.

Для решения данной задачи используем схему динамического программирования [21]. Обозначим через $\varphi(m, k)$ оптимальное значение целевой функции при текущем капитале k и подмножестве товаров $\{1, \ldots, m\}$, где $m = = 1, \ldots, n$, $k = 1, \ldots, K$. Закупка товара m в количестве x_m допустима, если $x_m \in \{\lambda_m, 2\lambda_m, 3\lambda_m, \ldots, v_m^*\}$ и $\sum\limits_{i=1}^{m-1} (\alpha_i + \beta_i \lambda_i) + \alpha_m + \beta_m x_m \leqslant k$. Множество допустимых значений x_m обозначим через $P(m, k)$.

Для вычисления $\varphi(m, k)$ перебираем все значения $x_m \in P(m, k)$. Если товар m закупаем в объеме x_m, то получаем подзадачу

$$\sum_{i=1}^{m-1} H_i(x_i) + H_m(x_m) \to \min,$$
$$\sum_{i=1}^{m-1} (\alpha_i + \beta_i x_i) \leqslant k - (\alpha_m + \beta_m x_m),$$
$$x_i \in \{\lambda_i, 2\lambda_i, 3\lambda_i, \ldots, v_i^*\}, i = 1, \ldots, m-1,$$

оптимум для которой равен $\varphi(m-1, k - \alpha_m - \beta_m x_m)$. Таким образом имеем рекуррентное соотношение

$$\varphi(m, k) = \min_{x_m \in P(m,k)} \{H_m(x_m) + \varphi(m-1, k - \alpha_m - \beta_m x_m)\}.$$

Для нахождения $\varphi(n, k)$ данное рекуррентное соотношение необходимо реализовать в двойном цикле $m = 2, \ldots, n$, $k = 1, \ldots, K$ при начальных

условиях $\varphi(1,k) = H_1(x_1)$, где $x_1 = \min\{\lfloor \frac{k-\alpha_1}{\beta_1\lambda_1} \rfloor \lambda_1, v_1^*\}$. Восстановление оптимального решения осуществляется обратным ходом по стандартной схеме. Заметим, что реализация алгоритма требует целочисленности величин $\alpha_m + \beta_m x_m$, что может быть обеспечено подбором единиц измерения капитала.

Трудоемкость алгоритма псевдополиномиально зависит от длины записи входных данных и составляет $O\left(K \sum\limits_{m=1}^{n} T_m^*\right)$ операций, где K — имеющийся капитал в выбранных единицах измерения, T_m^* — оптимальный период завоза товара m.

3.2. Формирование заявки при возможности использования кредитов

Обобщим выписанную модель на случай возможности использования кредитов. Ставка по кредиту r известна. Переменной D будем обозначать размер кредита. Как и ранее, $K_{\min} = \sum\limits_{i=1}^{n} (\alpha_i + \beta_i\lambda_i)$ — это минимально необходимый объем оборотных средств. Введем также величину $K_{\max} = \sum\limits_{i=1}^{n} (\alpha_i + \beta_i\lambda_i T_i^*)$ — размер средств, достаточный для полного обеспечения заявки.

В случае, если минимально необходимый размер оборотных средств $K_{\min}$ больше наличного капитала K, размер кредита не может быть меньше величины $K_{\min} - K$. Максимальное значение кредита D не превосходит $K_{\max} - K$.

Приведенные издержки на использование кредита составят $\frac{D(1+r)}{1+r_0} - D$. Размер доступного капитала будет равен $K + D$. Получаем следующую модель:

$$\sum_{i=1}^{n} H_i(x_i) + \frac{D(1+r)}{1+r_0} - D \to \min,$$

$$\sum_{i=1}^{n} (\alpha_i + \beta_i x_i) \leqslant K + D,$$

$$\max\{0, K_{\min} - K\} \leqslant D \leqslant K_{\max} - K,$$
$$x_i \in \{\lambda_i, 2\lambda_i, 3\lambda_i, \ldots, v_i^*\}.$$

Если переменную D зафиксировать, то величина $\frac{D(1+r)}{1+r_0} - D$ будет константой и получим задачу, описанную в предыдущем параграфе, с начальным капиталом $K + D$. Ее решаем описанным там же алгоритмом и находим все $\varphi(m,k)$ для $m = 1, \ldots, n$ и $k = 1, \ldots, K_{\max}$. После этого остается перебрать все целые значения $D \in [\max\{0, K_{\min} - K\}, K_{\max} - K]$ и найти минимум

$$\min_{D \in [\max\{0, K_{\min}-K\}, K_{\max}-K]} \left\{ \varphi(n, K+D) - D + \frac{D(1+r)}{1+r_0} \right\}.$$

Трудоемкость алгоритма составит $O\left(K_{\max}\sum\limits_{i=1}^{n}T_i^*\right)$ операций.

Расчеты на реальных данных показали актуальность модели с кредитами. Часто оптимум по D достигается внутри интервала

$$\left[\max\{0, K_{\min}-K\},\ K_{\max}-K\right],$$

т.е. заявка все-таки сокращается, но ее часть оплачивается за счет кредита. Аналогичный результат был получен для задачи календарного планирования инвестиционных проектов, когда за счет кредитов выполнялась только часть работ проекта [22].

4. Формирование заявки с учетом неравномерности потребления

В рассмотренных моделях в момент формирования заявки предполагалось, что интенсивность потребления каждого товара не меняется в течение всего периода его реализации. При следующем формировании заявки будет использоваться другое актуальное значение λ_i и остатков на складе. При этом значение λ_i могло измениться. Таким образом, функция интенсивности потребления хоть и являлась кусочно-постоянной по времени, но в рассматриваемый период была константой. В реальности процесс реализации товара может оказаться неравномерным. Поэтому наиболее эффективно вместо константы λ_i использовать функцию $\lambda_i(t)$, $t=1,\ldots,T$, где за $t=0$ берем текущий момент времени очередного заказа. Отметим, что в рассматриваемой практической задаче величины $\lambda_i(t)$ на период планирования T достаточно хорошо прогнозируются. Это обусловлено и спецификой задачи, и небольшим горизонтом планирования. Несмотря на это, в модели необходимо учесть, что реальный спрос может отклониться от заданного. Поэтому необходим некоторый механизм, который устраняет данную проблему. Как вариант можно формировать страховой запас, но это приводит к снижению прибыли и дополнительной оптимизации. Получаем две противоположные тенденции. С одной стороны, нежелательно возникновения дефицита товара на складе, так как это приводит к недополученной прибыли. С другой стороны, лишние запасы на складе уменьшают прибыль. Задачи такого типа исследуются в многочисленных публикациях, например [23]. В данной задаче ситуация проще. Так как в каждый целочисленный момент ситуация на складе отслеживается, то в момент T_i-1 можно сравнить остаток на складе и значение $\lambda_i(T_i)$. Если остаток меньше, то очередной заказ товара делаем в момент T_i-1. Пусть в момент очередного заказа на складе имеется остаток $R_i\geqslant 0$, которого не хватает для покрытия спроса на ближайший единичный период времени.

При этих условиях объем заказа товара i на период T_i составит $\sum\limits_{t=1}^{T_i}\lambda_i(t)-$ $-R_i$, а начальный объем на складе $\sum\limits_{t=1}^{T_i}\lambda_i(t)$. Приведенный доход будет равен $Q(T_i)=\sum\limits_{t=1}^{T_i}\frac{c_i\lambda_i(t)}{(1+r_0)^t}$. Здесь можно учесть и изменения цен продажи, используя

вместо c_i заданные значения $c_i(t)$:

$$Q_i(T_i) = \sum_{t=1}^{T_i} \frac{c_i(t)\lambda_i(t)}{(1+r_0)^t}.$$

Величины α_i и β_i учитываются только в момент $t = 0$. Поэтому нет смысла их варьировать.

Выпишем затраты на хранение $Z_i(T_i)$. Остаток товара i на складе на момент $t = 1, \ldots, T_i$ составит $\sum\limits_{\tau=1}^{T_i} \lambda_i(\tau) - \sum\limits_{\tau=1}^{t} \lambda_i(\tau) = \sum\limits_{\tau=t+1}^{T_i} \lambda_i(\tau)$. Средний остаток на интервале $[t-1, t]$, за который приходится платить, равен $\sum\limits_{\tau=t+1}^{T_i} \lambda_i(\tau) + \frac{\lambda_i(t)}{2}$. Можно также учесть изменение удельной стоимости хранения товаров во времени. В результате получаем следующую формулу:

$$Z_i(T_i) = \sum_{t=1}^{T_i} \frac{\left(\sum\limits_{\tau=t+1}^{T_i} \lambda_i(\tau) + \frac{\lambda_i(t)}{2} \right) c_i^{\text{xp}}(t)}{(1+r_0)^t}.$$

Функция удельной приведенной прибыли будет равна

$$U_i(T_i) = \frac{1}{T_i} \left(Q_i(T_i) - \alpha_i - \beta_i \sum_{t=1}^{T_i} \lambda_i(t) - Z_i(T_i) \right).$$

Оптимум T_i^* находим простым перебором для $T_i = 1, 2, \ldots,$ до тех пор пока функция $U_i(T_i)$ не начнет убывать.

Уточним математическую модель задачи. Пусть, как и ранее, $\{1, \ldots, n\}$ — это множество номеров товаров, заказываемых в момент $t = 0$. Оптимальный объем товара i в заявке равен $\sum\limits_{t=1}^{T_i^*} \lambda_i(t) - R_i$. Тогда стоимость оптимальной заявки составит

$$K_{\max} = \sum_{i=1}^{n} \left(\alpha_i + \beta_i \left(\sum_{t=1}^{T_i^*} \lambda_i(t) - R_i \right) \right).$$

Средства, необходимые для удовлетворения минимального спроса, будут равны

$$K_{\min} = \sum_{i=1}^{n} (\alpha_i + \beta_i(\lambda_i(1) - R_i)).$$

Переменная x_i может принимать значения $\sum\limits_{t=1}^{T_i} \lambda_i(t) - R_i,\ T_i = 1, \ldots, T_i^*$. Функция потерь $H_i(x_i)$ определяется как отклонение от оптимального значения прибыли $U_i(T_i^*) - U_i(T_i)$. Таким образом, итоговая модель сокращения

заявки и оптимизации кредитных заимствований принимает вид:

$$\sum_{i=1}^{n} H_i(x_i) + \frac{D(1+r)}{1+r_0} - D \to \min,$$

$$\sum_{i=1}^{n} (\alpha_i + \beta_i x_i) \leqslant K + D,$$

$$\max\{0, K_{\min} - K\} \leqslant D \leqslant K_{\max} - K,$$

$$x_i \in \left\{ \sum_{t=1}^{T_i} \lambda_i(t) - R_i \mid T_i = 1, \ldots, T_i^* \right\}.$$

Для решения сформулированной задачи используем описанный выше алгоритм динамического программирования. Алгоритм реализован и внедрен в логистической системе управления закупками одного из дистрибьюторов фармацевтического рынка. Общее число позиций в номенклатуре товаров составляет более 40 тысяч. Ежедневный заказ включает несколько тысяч наименований товаров. Оптовая отгрузка осуществляется через кросс-докинговый склад. База данных о ценах, спросе и других параметрах обновляется ежедневно. Информация об остатках на складе компании отслеживается в реальном режиме времени. Расчет заявки происходит в конце каждого дня с учетом текущих параметров системы.

5. Заключение

В работе исследована задача максимизации прибыли с учетом альтернативного использования капитала. Доказана теорема о единственности точки максимума функции прибыли. Построена модель задачи формирования заявки максимизации прибыли с учетом ограничения оборотного капитала, выявлены и обоснованы ее свойства, предложен и реализован алгоритм решения задачи, основанный на схеме динамического программирования. Предложен и реализован алгоритм решения задачи формирования заявки при возможности использования кредитов. Рассмотрен случай формирования заявки с учетом неравномерности потребления. Разработанный подход используется при оперативном управлении поставками в компании, торгующей медикаментами.

СПИСОК ЛИТЕРАТУРЫ

1. *Wilson R.H.* A Scientific Routine for Stock Control // Harvard Business Rev. 1934. No. 13. P. 116–128.
2. *Букан Дж., Кенигсберг Э.* Научное управление запасами. М.: Наука, 1967.
3. *Первозванский А.А.* Математические модели в управлении производством. М.: Наука. Главная редакция физ.-мат. лит. 1975.
4. *Рыжиков Ю.И.* Теория очередей и управление запасами. СПб.: Питер, 2001.
5. *Хедли Дж., Уайтин Т.* Анализ систем управления запасами. М.: Наука, 1969.

6. *Chandra M.J., Bahner M.L.* The effects of inflation and the time value of money on some inventory systems // Int. J. Product. Res. 1985. V. 23. No. 4. P.723–729.

7. *Бродецкий Г.Л.* Модели оптимизации систем управления запасами с учетом временной стоимости денег при ограничениях на размер капитала // Логистика и управление цепями поставок. 2007. № 2. С. 70–88.

8. *Бродецкий Г.Л.* Многономенклатурное управление запасами: новый подход к оптимизации решений // Логистика сегодня. 2014. № 1. С. 34–45.

9. *Brodetskiy G.L.* The new approach to inventory optimization // Int. J. Logist. Syst. Management (IJLSM). 2015. V. 22. No. 3. P. 251–266.

10. *Лукинский В.В.* Актуальные проблемы формирования теории управления запасами. СПб.: СПбГИЭУ, 2008.

11. *Лукинский В.В.* Управление запасами в цепях поставок: в 2 ч. Ч. 2: учебник и практикум для бакалавриата и магистратуры. М.: Изд-во Юрайт, 2017.

12. *Бродецкий Г.Л., Герами В.Д., Колик А.В., Шидловский И.Г.* Управление запасами: многофакторная оптимизация процесса поставок. М.: Изд-во Юрайт, 2019.

13. *Бурлакова Н.И., Сервах В.В.* Максимизация чистой приведенной прибыли в задаче управления запасами // Сб. научных тр. VIII Междунар. школы-симпозиума “Анализ, управление, моделирование, развитие”. Симферополь: ТНУ им. В.И. Вернадского, 2014. С. 61–62.

14. *Бурлакова Н.И., Полянцева И.А., Сервах В.В.* Оптимизация закупок с учетом альтернативного использования капитала // Тез. докл. XVI Байкал. междунар. школы-семинара “Методы оптимизации и их приложения”, Иркутск, ИСЭМ СО РАН, 2014. 32 с.

15. *Танаев В.С., Гордон В.С., Шафранский Я.М.* Теория расписаний. Одностадийные системы. М.: Наука, 1984.

16. *Танаев В.С., Сотсков Ю.Н., Струсевич В.А.* Теория расписаний. Многостадийные системы. М.: Наука, 1989.

17. *Braun, O., Sotskov, Yu.N.* Scheduling personal finances via integer programming // J. Math. Modell. Algorithm., 2013. V. 12. No. 2. P. 179–199.

18. *Голами О., Сотсков Ю.Н, Вернер Ф. Затюпо О.С.* Эвристические алгоритмы для максимизации дохода и количества требований, обслуживаемых на параллельных приборах // АиТ. 2019. № 2. С. 125–151.

    Gholami O., Sotskov Y.N., Werner F., Zatsiupo A.S. Heuristic Algorithms to Maximize Revenue and the Number of Jobs Processed on Parallel Machines // Autom. Remote Control. 2019. Vo. 80. No. 2. P. 297–316.

19. *Chauhan S.S., Eremeev A.V., Romanova A.A., Servakh V.V., Woeginger G.J.* Approximation of the supply scheduling problem // Oper. Res. Lett. 2005. V. 33. No. 3. P. 249–254.

20. *Eremeev A.V., Romanova A.A., Chauhan S.S., Servakh V.V.* Approximate Solution of the Supply Management Problem // J. Appl Industr. Math. 2007. V. 1. No. 4. С. 1–9.

21. *Гимади Э.Х., Глебов Н.И.* Математические модели и методы принятия решений. Уч. пос. / Новосибирск: Изд-во Новосиб. гос. ун-та. 162с.

22. *Мартынова Е.А., Сервах В.В.* О задаче календарного планирования проектов с использованием кредитов // АиТ. 2012. № 3. С. 107–116.

    Martynova E.A., Servakh V.V. On Scheduling Credited Projects // Autom. Remote Control. 2012. V. 73. No. 3. P. 508–516.

23. *Singha K., Buddhakulsomsiri J., Parthanadee P.* Mathematical Model of (R,Q) Inventory Policy under Limited Storage Space for Continuous and Periodic Review Policies with Backlog and Lost Sales // Math. Probl. Engineer. December 2017. P. 1–9.

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

Поступила в редакцию 17.07.2019

После доработки 15.10.2019

Принята к публикации 28.11.2019

© 2020 г. Е.Р. ГАФАРОВ, д-р физ.-мат. наук (axel73@mail.ru)
(Институт проблем управления им. В.А. Трапезникова РАН, Москва),
А.А. ЛАЗАРЕВ, д-р физ.-мат. наук (jobmath@mail.ru)
(Институт проблем управления им. В.А. Трапезникова РАН, Москва),
Ф. ВЕРНЕР, профессор (frank.werner@mathematik.uni-magdeburg.de)
(Университет Отто-фон-Герике, Магдебург, Германия)

МИНИМИЗАЦИЯ СУММАРНОГО ВЗВЕШЕННОГО ЗАПАЗДЫВАНИЯ НА ОДНОМ ПРИБОРЕ С РАВНЫМИ ПРОДОЛЖИТЕЛЬНОСТЯМИ ОБСЛУЖИВАНИЯ ТРЕБОВАНИЙ[1]

Рассматривается задача теории расписаний, в которой необходимо минимизировать суммарное взвешенное запаздывание на одном приборе с равными продолжительностями обслуживания требований и неодновременным поступлением требований на обслуживание. Эта задача упомянута как минимальная, статус вычислительной сложности которой неизвестен: http://www2.informatik.uni-osnabrueck.de/knust/class/dateien/classes/ein_ma/ein_ma. Последние результаты по данной задаче опубликованы в 2000 и 2005 гг., а именно, алгоритмы решения частных случаев задачи. В данной статье представлены некоторые свойства задачи и пути дальнейших исследований.

Ключевые слова: теория расписаний, один прибор, суммарное взвешенное запаздывание.

DOI: 10.31857/S0005231020050086

1. Введение

Формулировка задачи: дано множество $N = \{1, \ldots, n\}$ из n требований, которые должны быть обслужены на одном приборе. Прерывание обслуживания требований не допускается. Одновременно прибор может обслуживать не более одного требования. Прибор готов к обслуживанию требований с момента времени 0. Для каждого требования $j \in N$ заданы фиксированное время обслуживания $p_j = p \in Z^+$, директивный срок $d_j \in Z^+$, время поступления $r_j \in Z^+$ (т.е. наиболее раннее время начала обслуживания) и вес $w_j \in Z^+$.

Будем называть активным такое расписание, при котором ни одно требование не может быть перемещено на более ранний срок обслуживания без нарушения ограничений задачи. В данной статье рассматриваются только активные расписания.

Расписание π однозначно определяется перестановкой требований из множества N. Пусть $C_j(\pi)$ – время завершения обслуживания требования j при расписании π. Если $C_j(\pi) > d_j$, тогда требование j запаздывает и принимается $U_j = 1$, иначе $U_j = 0$. Если $C_j(\pi) \leqslant d_j$, тогда требование j не запаздывает.

[1] Исследование выполнено при частичной поддержке гранта Российского научного фонда (проект № 17-19-01665).

Пусть $T_j(\pi) = \max\{0, C_j(\pi) - d_j\}$ — запаздывание требования j при расписании π. Определим $S_j = C_j - p$ как время начала обслуживания требования j при расписании π.

В задаче минимизации суммарного взвешенного запаздывания на одном приборе с равными продолжительностями обслуживания требований и неодновременным поступлением требований на обслуживание необходимо построить оптимальное расписание π^*, при котором достигается минимум функции

$$\sum_{j=1}^{n} f_j(C_j) = \sum_{j=1}^{n} w_j T_j.$$

Обозначим данную задачу как $1|r_j, p_j = p|\sum w_j T_j$ в соответствии с обозначением $\alpha|\beta|\gamma$ задач теории расписаний, представленным в работе [1], где α описывает характеристики приборов, β описывает характеристики требований и ограничения, а γ — целевую функцию. Если $w_j = 1$, $j \in N$, то этот частный случай обозначим как $1|r_j,\ p_j = p|\sum T_j$. В задаче минимизации взвешенного числа запаздывающих требований необходимо минимизировать значение функции

$$\sum_{j=1}^{n} f_j(C_j) = \sum_{j=1}^{n} w_j U_j.$$

Обозначим эту задачу как $1|r_j, p_j = p|\sum w_j U_j$.

В [2, 3] предложены полиномиальные алгоритмы решения задач $1|r_j, p_j = p|\sum T_j$ и $1|r_j, p_j = p|\sum w_j U_j$. Оба алгоритма основаны на следующих свойствах:

- известно множество времен начала обслуживания требований

$$\Theta = \{t : t = r_j + kp, j \in N, k = 0, 1, \ldots, n-1\}.$$

  Мощность множества $|\Theta| \leqslant n^2$;
- известны отношения доминирования между требованиями. Существует оптимальное расписание, при котором требование j обслуживается после всех требований i, где $r_i \leqslant S_j$ и $d_i \leqslant d_j$.

Частный случай, задача $1|r_j, p_j = p|\sum w_j T_j$, с общим директивным сроком может быть решен с помощью алгоритмов [3]. Для данной задачи существует оптимальное расписание, при котором требование j обслуживается после всех требований i, где $r_i \leqslant S_j$ и $w_i \geqslant w_j$.

Частный случай задачи $1|r_j, p_j = p|\sum w_j T_j$, при котором все значения r_j кратны p, может быть сведен к задаче о назначениях и решен за полиномиальное время.

В [4] авторы рассматривают одноприборную задачу теории расписаний — минимизацию суммарного взвешенного штрафа за отклонение от директивного срока $1|p_j = p|\sum(\alpha_j E_j + \beta_j T_j)$, где $E_j(\pi) = \max\{0, d_j - C_j(\pi)\}$. Авторы представили постановку задачи как задачи целочисленного линейного программирования (ЦЛП), предложили схему ее релаксации, для которой построен полиномиальный алгоритм решения. В задаче ЦЛП определены переменные $x_{j,t}$ для каждого требования j и возможного времени завершения

обслуживания требования t, где $x_{j,t} = 1$, если обслуживание требования j заканчивается в момент времени t, и $x_{j,t} = 0$ иначе. В релаксированной задаче $x_{j,t} \in [0, 1]$. Авторы представили полиномиальный алгоритм преобразования решения релаксированной задачи в целочисленное решение исходной задачи с тем же значением целевой функции.

В [5] авторы представили аналогичную модель ЦЛП и аналогичную релаксацию для задачи $1|r_j, p_j = p|\sum w_j T_j$. Авторы утверждают, что если в задаче нет двух требований i и j, для которых $d_i < d_j$ и $w_i < w_j$, то построенный полиномиальный алгоритм преобразует решение релаксированной задачи в оптимальное целочисленное решение исходной задачи. К сожалению, в работе не представлен подобный алгоритм, а дана ссылка на алгоритм из [4], хотя в отличие от задачи $1|r_j, p_j = p|\sum w_j T_j$ в задаче $1|p_j = p|\sum(\alpha_j E_j + \beta_j T_j)$ требования поступают на обслуживание одновременно, а множество времен окончания обслуживания требований относятся к другому интервалу.

Обзор результатов по задачам для параллельных машин с одинаковыми продолжительностями обслуживания требований представлен в [6].

В разделе 2 представлены найденные свойства задачи. В разделе 3 рассмотрены подходы к ее решению. В разделе 4 представлены алгоритмы решения частных случаев, а в разделе 5 — результаты исследования вычислительной сложности. Некоторые задачи максимизации рассмотрены в разделе 6.

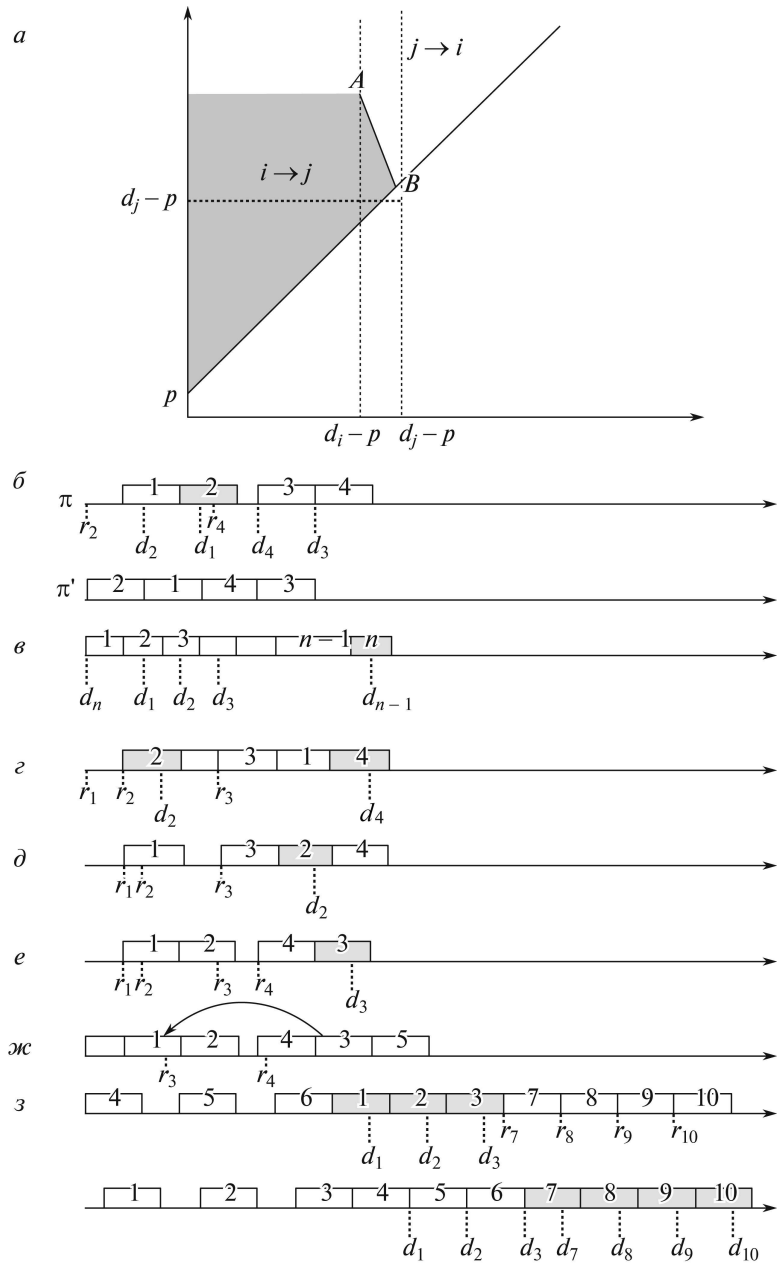
2. Свойства задачи $1|r_j, p_j = p|\sum w_j T_j$

Рассмотрим отношения предшествования между двумя требованиями i и j в оптимальном расписании.

Свойство 1. Пусть t_1 — наиболее ранний момент времени из моментов начала обслуживания требований $\{i, j\}$ и t_2 — поздний момент времени, где $t_2 \geqslant t_1 + p$. Тогда выполняются следующие отношения предшествования.

1. *Если $w_i \geqslant w_j$, $d_i \leqslant d_j$ и $t_1 \geqslant r_i$, тогда требование i обслуживается раньше требования j.*
2. *Если $w_i < w_j$, $d_i \leqslant d_j$ и $t_1 \geqslant \max\{r_i, r_j\}$, то*
   (а) *если $t_2 + p \leqslant d_j$ (требование j не запаздывает, если оно обслуживается начиная со времени t_2), тогда требование i обслуживается раньше требования j;*
   (б) *если $t_1 > d_j - p$ (оба требования запаздывают), тогда требование j обслуживается раньше требования i;*
   (в) *пусть $d_i - p < t_1 \leqslant d_j - p$ и пусть при расписании $\pi = (\pi_1, i, \pi_2, j, \pi_3)$ имеем $S_i(\pi) = t_1$ и $S_j(\pi) = t_2$, а при расписании $\pi' = (\pi_1, j, \pi_2, i, \pi_3)$ имеем $S_i(\pi') = t_2$ и $S_j(\pi') = t_1$. Тогда требование i обслуживается раньше требования j, если выполняется*

$$\sum w_j T_j(\pi) < \sum w_j T_j(\pi') \Longleftrightarrow w_i(t_2 - t_1) - w_j(t_2 + p - d_j) > 0$$
$$\Longleftrightarrow t_2 < \frac{w_j(d_j - p) - w_i t_1}{w_j - w_i};$$



Примеры.

(г) *пусть* $t_1 + p \leqslant d_i \leqslant d_j < t_2 + p$. *Тогда требование* i *обслуживается раньше требования* j, *если выполняется*

$$\sum w_j T_j(\pi) < \sum w_j T_j(\pi') \iff w_i(t_2 + p - d_i) > w_j(t_2 + p - d_j)$$
$$\iff t_2 < \frac{w_j(d_j - p) - w_i(d_i - p)}{w_j - w_i}.$$

Эти отношения предшествования изображены на рисунке, а, где начало координат — точка $(r_{\max}, r_{\max})$, $r_{\max} = \max\{r_i, r_j\}$, *точка* A *есть*

$$A = \left(d_i - p; \frac{w_j(d_j - p) - w_i(d_i - p)}{w_j - w_i} \right)$$

и точка B *есть точка пересечения двух линий*

$$t_2 = t_1 + p \qquad \text{и} \qquad t_2 = \frac{w_j(d_j - p) - w_i t_1}{w_j - w_i}.$$

Точки ниже линии $t_2 = t_1 + p$ *не рассматриваются.*

Отметим, что если $t_1 < r_j$, то сложно установить подобные отношения предшествования, так как при расписании π' требования, которые обслуживаются в интервале времени $[t_1 + p, t_2)$, могут быть смещены в обслуживании на более поздний срок.

В алгоритме динамического программирования, представленного в [3], рассматриваются интервалы $[s, e]$, $s, e \in \Theta$, в которых обслуживаются требования подмножества $N' = \{1, \ldots, k\}$, $k \leqslant n$, т.е. не более k требований в каждом из интервалов. Трудоемкость алгоритма динамического программирования полиномиально зависит от количества таких интервалов и равна $O(n^4)$ операций, так как $|\Theta| = O(n^2)$. Возникает вопрос, как много точек e необходимо рассмотреть для фиксированного значения s, т.е. существует ли пример, в котором нужно рассмотреть порядка $O(n^2)$ точек e?

Свойство 2. Для фиксированных s *и* k *существует порядка* $O(k^2)$ *точек* $e \in \Theta$.

Доказательство. Существует пример, в котором

$$\frac{k}{4} = \left\lceil \frac{k}{4} \right\rceil \qquad \text{и} \qquad p \gg k.$$

Для первых $\frac{k}{4}$ требований пусть

$$r_1 = s, \quad r_2 = r_1 + 2p, \quad r_3 = r_2 + 2p, \quad \ldots,$$

т.е.

$$r_i = s + 2p(i-1), \quad i = 1, \ldots, \frac{k}{4}.$$

Для следующих $\frac{k}{4}$ требований пусть

$$r_i = r_{i-\frac{k}{4}+1} + 1, \quad i = \frac{k}{4}, \frac{k}{4} + 1, \ldots, \frac{k}{2}.$$

Для следующих $\frac{k}{2}$ требований пусть

$$r_i = \frac{k}{2}p + 1 + \left(i - \frac{k}{2} \right), \quad i = \frac{k}{2} + 1, \ldots, k.$$

При активном расписании от $\frac{k}{4}$ до $\frac{k}{2}$ активных требований обслуживаются в начале расписания перед временем $r_{\frac{k}{2}+1}$, а от $\frac{k}{2}$ до $\frac{3k}{4}$ требований обслуживаются начиная с момента времени r_i, $i \in [\frac{k}{2}+1, k]$, без простоя прибора.

Тогда для данного примера существует $O(k^2)$ активных расписаний и $O(k^2)$ возможных моментов времени окончания обслуживания e.

Свойство 3. Пусть

$$T = \left\{ t : t \in \Theta \bigcap [t_1, t_1 + p] \right\},$$

т.е. T есть подмножество точек, входящих в интервал длины p. Тогда $|T| = O(n)$.

3. Алгоритмы решения задачи

Рассматриваемая задача может быть решена с помощью алгоритма динамического программирования и функциональных равенств, представленных в [7].

Пусть дано расписание, при котором первые k обслуживаемых требований составляют подмножество $N \setminus J$, а оставшиеся $n-k$ требований составляют подмножество J. Пусть функция $F(J,t)$ задает минимальное суммарное взвешенное запаздывание требований из подмножества J при условии, что обслуживание этих требований начинается не ранее момента времени $t \in \Theta$. В алгоритме динамического программирования необходимо вычислить значение $F(N,0)$ и найти соответствующее расписание. Функциональные равенства, предложенные в [7], имеют следующий вид:

$$F(J,t) = \min_{i \in J} \left\{ w_i \max\{0, t+p-d_i\} + F(J \setminus \{i\}, t+p) \right\}$$

и $F(\emptyset, t) = 0, \forall t \in \Theta$.

В алгоритме динамического программирования вычисления производятся согласно данным функциональным равенствам. Трудоемкость алгоритма — $O(n^3 2^n)$ операций.

Альтернативный алгоритм решения имеет такую же вычислительную сложность. В этом алгоритме рассматриваются 2^n подмножеств $N' \subseteq N$, где

$$|r_{j_1} - r_{j_2}| \geqslant p, \quad j_1, j_2 \in N'.$$

Для каждого подмножества N' примем $S_j = r_j$, $j \in N'$, и для каждого требования $i \in N \setminus N'$ возможные моменты времени начала обслуживания требований принадлежат множеству Θ', являющемуся подмножеством из $n - |N'|$ наименьших значений множества

$$\left\{ t : t = r_j + kp, j \in N', k = 1, \ldots, n - |N'| \right\}.$$

Тогда данная задача может быть сведена к задаче о назначениях, которая может быть решена за время $O((n-|N'|)^3)$.

Алгоритм, представленный в [3] для частного случая с равными весами требований, может быть использован как эвристика для решения исходной задачи. Несложно представить пример исходной задачи, для которого алгоритм из [3] не вернет оптимальное решение. Пусть в данном алгоритме зафиксировано отношение предшествования, где обслуживание требования j предшествует обслуживанию всех требований i, таких, что $r_i \leqslant S_j$ и $w_i \geqslant w_j$. Тогда несложно представить пример, где при оптимальном расписании требование j предшествует требованию i и

$$C_j - d_j = 1, \qquad C_i - d_i = 1, \qquad C_j < C_i.$$

3.1. Алгоритмы локального поиска

Локальный поиск — это эвристический метод решения задач комбинаторной оптимизации. В этом методе из исходного решения с помощью локальных модификаций строится альтернативное допустимое решение. Эта модификация повторяется до тех пор, пока не найден локальный оптимум или достигнут лимит выполненных операций.

В данной работе рассмотрим следующие модификации расписания. Обозначим активное расписание следующей последовательностью обслуживания требований $\pi = (\pi_1, i, \pi_2, j, \pi_3)$:

- **левый сдвиг**. Обслужить требование j сразу после частичного выполнения частичного расписания π_1, т.е. имеем модифицированное расписание $\pi' = (\pi_1, j, i, \pi_2, \pi_3)$;
- **правый сдвиг**. Обслужить требование i сразу после частичного выполнения частичного расписания π_2, т.е. имеем модифицированное расписание $\pi' = (\pi_1, \pi_2, i, j, \pi_3)$;
- **парная перестановка**. Поменять местами требования i и j в последовательности обслуживания требований, т.е. $\pi' = (\pi_1, j, \pi_2, i, \pi_3)$.

Возникает вопрос: “Можно ли с помощью данных модификаций найти оптимальное расписание, причем при каждой модификации значение целевой функции не должно увеличиваться.”

Свойство 4. Существуют пример задачи и расписание, при котором каждый левый сдвиг увеличивает значение целевой функции, но применение нескольких левых сдвигов уменьшает значение целевой функции.

Доказательство. Рассмотрим пример, где

$$r_1 = 2,\ r_2 = 0,\ r_3 = 9,\ r_4 = 6,\ p = 3,\ d_1 = 6,\ d_2 = 3,\ d_3 = 12,\ d_4 = 9,$$
$$w_1 = w_3 = 100,\ w_2 = w_4 = 1$$

(см. рисунок,b). Пусть исходное расписание $\pi = (1, 2, 3, 4)$. При левом сдвиге требования 2 на позицию перед требованием 1 значение целевой функции увеличивается. Аналогично при левом сдвиге требования 4 на позицию перед требованием 3 значение целевой функции увеличивается. Но при применении обоих левых сдвигов будет получено расписание $\pi' = (2, 1, 4, 3)$, при котором значение целевой функции уменьшается. При расписании π' суммарное взвешенное запаздывание равно 0, а при расписании π имеем суммарное взвешенное запаздывание, равное 11.

Следствие 1. Пусть π — расписание, при котором каждый левый сдвиг увеличивает значение целевой функции. Тогда относительная погрешность расписания π может быть сколь угодно большой.

Аналогичное доказательство можно представить для следующего свойства.

Свойство 5. Существуют пример задачи и расписание, при котором каждый правый сдвиг увеличивает значение целевой функции, но применение нескольких правых сдвигов уменьшает значение целевой функции.

Свойство 6. Пусть π — некоторое расписание и $N' = \{j_1, \ldots, j_k\}$ — подмножество требований, которые должны быть сдвинуты влево, и пусть порядок обслуживания требований из N' фиксирован, т.е. обслуживание требования j_1 предшествует обслуживанию требования j_2, требование j_2 предшествует требованию j_3 и т.д. Тогда оптимальная модификация расписания путем левого сдвига каждого требования из N' может быть найдена за $O(n^8)$ операций.

Доказательство. Оптимальная модификация может быть найдена с помощью следующего алгоритма динамического программирования. На каждой стадии $l = k,\ k-1, \ldots, 1$ для каждого требования $j_l \in N'$ необходимо рассмотреть x_l позиций в перестановке, в которой порядок обслуживания требований из $N \setminus N'$ остается неизменным и все возможные моменты начала обслуживания $t_l \in \Theta$. Таким образом, состояние системы в алгоритме динамического программирования определяется вектором (x_l, t_l). На каждой стадии необходимо вычислить функцию $F_l(x_l, t_l)$, являющуюся суммарным взвешенным запаздыванием требований, обслуживаемых при активном расписании с момента t_l согласно последовательности, начинающейся с требования j_l. Эта функция используется на следующей стадии для требования j_{l-1}. На каждой стадии необходимо рассмотреть $O(n^{1+2})$ состояний системы. Для каждого состояния системы (x_l, t_l) необходимо рассмотреть состояния системы, полученные на предыдущей стадии, чтобы за $O(n)$ операций вычислить значение функции $F_l(x_l, t_l)$. Таким образом, трудоемкость алгоритма динамического программирования составляет $O(n^8)$ операций.

Свойство 7. Пусть для расписания $(1, 2, 3)$ две попарные перестановки $1 \Longleftrightarrow 2$ и $2 \Longleftrightarrow 3$ приводят к увеличению значения целевой функции. Тогда существует пример, для которого попарная перестановка $1 \Longleftrightarrow 3$ приводит к уменьшению значения целевой функции.

Параметры данного примера:

$$r_1 = r_2 = r_3 = 0,\ p = 3,\ d_1 = 5,\ d_2 = 7,\ d_3 = 8,\ w_1 = 1,\ w_2 = w_3 = 5.$$

При расписании $(1, 2, 3)$ имеем $\sum w_j T_j = 5$, а при расписании $(3, 2, 1)$ имеем $\sum w_j T_j = 4$.

Следствие 2. При локальном поиске с помощью попарной перестановки необходимо рассмотреть $O(n^2)$ пар требований.

Свойство 8. Пусть для расписания $(1, \ldots, n-1, n)$ любая попарная перестановка приводит к увеличению значения целевой функции. То-

гда существует пример, в котором левый сдвиг приводит к расписанию $(n, 1, \ldots, n-1)$ *с меньшим значением целевой функции.*

Доказательство. См. рисунок, *в*. Рассмотрим следующий пример:

$$(1)\qquad \begin{cases} n > 3, & \\ r_i = 0, & i = 1\ldots, n, \\ p = 2, & \\ w_i = pw_n - 1, & i = 1\ldots, n-2, \\ w_{n-1} = pw_n + 1, & i = 1\ldots, n-2, \\ d_n = 0, & \\ d_i = (i+1)p - 1, & i = 1\ldots, n-1. \end{cases}$$

При расписании $\pi = (1, \ldots, n-1, n)$ любая попарная перестановка приводит к увеличению значения целевой функции. Но при расписании $\pi' = (n, 1, \ldots$ $\ldots, n-1)$ имеем

$$F(\pi') = F(\pi) + \sum_{i=1}^{n-1} w_n - p(n-1)w_n = \\ = F(\pi) + p(n-1)w_n - (n-2) + 1 - p(n-1)w_n < F(\pi).$$

Свойство 9. Существуют пример и расписание, при котором любой левый сдвиг приводит к увеличению значения целевой функции, но возможен правый сдвиг, который приводит к уменьшению значения целевой функции.

3.2. Релаксация задачи

Рассмотрим релаксацию задачи, при которой все значения r_i округлены таким образом, что расстояние между ними кратно p. Например,

$$r'_j = \left\lfloor \frac{r_j}{p} \right\rfloor p, \qquad d'_j = d_j - (r_j - r'_j).$$

Такой модифицированный пример I' может быть сведен к задаче о назначениях и решен за полиномиальное время. Пусть π — оптимальное расписание для исходного примера I и π' — оптимальное расписание для модифицированного примера I' с округленными параметрами.

Рассмотрим вопросы:

- какова относительная погрешность

$$\frac{F(\pi) - F(\pi')}{F(\pi)}$$

  для примера I?
- какова разница значений $F(\pi)$ для примера I и $F'(\pi')$ для примера I', где F' — значение целевой функции для примера с округленными параметрами?

Свойство 10. Существует пример, для которого значение

$$\frac{F(\pi)}{F'(\pi')}$$

может быть сколь угодно большим.

Рассмотрим следующий пример, где $n = 2$, $p = 3$, $r_1 = 1$, $r_2 = 3$, $d_1 = 4$, $d_2 = 6$, $w_1 = w_2 = 1$. При расписании $\pi = \pi' = (1, 2)$ имеем $F(\pi) = 1$, а также $F'(\pi') = 0$, где $r'_1 = 0$, $d'_1 = 3$, $r'_2 = r_2$, $d'_2 = d_2$.

Свойство 11. Существует пример, для которого

$$\frac{F(\pi) - F(\pi')}{F(\pi)}$$

может быть сколь угодно большим.

Рассмотрим следующий пример:

$$(2) \qquad \begin{cases} p = 4, \\ r_n = 0, \\ r_i = \dfrac{p}{2}, & i = 1 \ldots, n-1, \\ d_i = r_i + 2p - 1, & i = 1 \ldots, n-1, \\ d_n = p, \\ w_n = 1, \\ w_i = np, & i = 1 \ldots, n-1. \end{cases}$$

Расписание $\pi' = (1, \ldots, n-1, n)$ является оптимальным для примера I', при котором запаздывает только требование n. Для примера I при этом расписании также запаздывает только требование n. При расписании $\pi = (n, 1, \ldots, n-1)$, оптимальном для примера I, ни одно из требований не запаздывает.

Можно рассмотреть другую релаксацию, при которой значения r'_i вычисляются таким образом, чтобы минимизировать значение $\sum_{j=1}^{n} |r_j - r'_j|$. Для такой релаксации свойства 10 и 11 также верны. Чтобы доказать это, необходимо рассмотреть n дополнительных требований, для которых $r_j = np$, $j = n+1, \ldots, 2n$.

4. Частные случаи задачи $1|r_j, p_j = p| \sum w_j T_j$

Представим алгоритмы решения двух частных случаев задачи. Как было сказано выше, частный случай задачи с общим директивным сроком может быть решен за полиномиальное время.

Рассмотрим частный случай, в котором разница максимального и минимального директивных сроков составляет $d_{\max} - d_{\min} \leqslant p$. Этот частный случай может быть решен с помощью следующего алгоритма.

Выберем два пограничных требования j_1 и j_2, которые будут обслужены при активном расписании одно за другим начиная с момента времени S_{j_1}

так, что никакое другое требование не может быть обслужено в интервале $[d_{\min}, d_{\max}]$, т.е.

$$S_{j_1} \leqslant d_{\min}, \quad C_{j_2} \geqslant d_{\max}.$$

Тогда подзадача со множеством требований $N \setminus \{j_1, j_2\}$ может быть решена модификацией алгоритма, представленного в [3], за $O(n^7)$ операций, где требование j обслуживается после обслуживания всех требований i, для которых $r_i \leqslant S_j$ и $w_i \geqslant w_j$. В модифицированном алгоритме рассматриваются только интервалы $[s, e]$, которые не пересекаются с интервалом $[S_{j_1}, C_{j_2}]$.

Имеется $O(n^2)$ пар пограничных требований j_1 и j_2, $O(n)$ возможных моментов начала обслуживания $S_{j_1} \in \Theta$ согласно свойству 3 и единственное значение S_{j_2} для выбранного S_{j_1}, так как рассматриваются только активные расписания. Тогда данный частный случай может быть решен за $O(n^{2+1+7})$ операций.

Далее рассмотрим частный случай с n требованиями, где

$$\begin{cases} w_1 \leqslant w_2 \leqslant \ldots \leqslant w_{n-1}, \\ d_1 \geqslant d_2 \geqslant \ldots \geqslant d_{n-1}. \end{cases} \tag{3}$$

Для решения данного частного случая необходимо выбрать время начала обслуживания $S_n \in \Theta$ и затем решить оставшуюся подзадачу с помощью модифицированного алгоритма из [3]. То есть данный частный случай может быть решен за $O(n^{2+7})$ операций.

5. Результаты численных экспериментов

В разделе представлены результаты численных экспериментов, проведенных с целью выяснить количество возможных точек в Θ, если для k требований время начала обслуживания фиксировано.

При этом был рассмотрен следующий частный случай задачи:

$$\begin{cases} w_1 \leqslant w_2 \leqslant \ldots \leqslant w_n, \\ d_1 \leqslant d_2 \leqslant \ldots \leqslant d_n \end{cases} \tag{4}$$

с $n = 10$.

Числовые параметры примеров были сгенерированы следующим образом. Для каждого значения $p \in \{5, 10, 15, 20, 25, 30\}$ были рассмотрены 10 примеров, для которых значения $r_j \in [0, (n-2)p]$, $d \in [0, (n-1)p]$, $w \in [1, 120]$ были определены случайным образом согласно равномерному распределению. Для каждого примера был использован алгоритм ветвей и границ с подпрограммой ветвления (j, Π), где j — требование, для которого в подпрограмме необходимо определить время начала обслуживания, и Π — частично построенное расписание, при котором моменты начала обслуживания S_i, $i = 1, \ldots, j-1$, уже определены. В данной подпрограмме вычисляется множество Θ_j согласно Π и значению r_j. Пусть $S_{j'} < S_{j''}$, $j', j'' \leqslant j-1$, и нет других требований

$j''' \leqslant j-1$, для которых $S_{j'} < S_{j'''} < S_{j''}$. Тогда определим множества:

$$\Omega_1 = \{S_{j'} + kp, k = 1, \ldots, n-j : S_{j'} + kp + p \leqslant S_{j''}, S_{j'} + kp \geqslant r_j\},$$
$$\Omega_2 = \{r_i + kp, k = 1, \ldots, n-j, i \geqslant j, r_j \leqslant r_i + kp \leqslant S_{j''} - p, r_i > S_{j'} + p\} \text{ и}$$
$$\Omega_3 = \{S_{j''} - kp, k = 1, \ldots, n-j : S_{j''} - kp - p \geqslant S_{j'}, S_{j''} - kp \geqslant r_j\}.$$

Пусть t — максимальное значение из множества Ω_2, где $(S_{j''} - t)$ кратно значению p. Если такое t существует, тогда из множества Ω_2 удаляются все значения, которые больше t, и из множества Ω_3 удаляются все значения, которые меньше t. Тогда множество Θ_j состоит из элементов множеств $\Omega_1, \Omega_2, \Omega_3$ и значения r_j, если $S_{j'} + p < r_j < S_{j''} - p$.

В подпрограмме ветвления рассматриваются все возможные моменты времени $t \in \Theta_j$. Пусть $TWT(\Pi)$ — суммарное взвешенное запаздывание требований, обслуживаемых при частичном расписании Π. Если

$$TWT(\Pi) + w_j \max\{t + p - d_j, 0\} \geqslant UB,$$

тогда t исключается из рассмотрения, где верхняя оценка UB — лучшее (минимальное) найденное на данный момент значение целевой функции.

Представим результат численного эксперимента для одного примера.

$$p = 10, r_j \in \{72, 58, 29, 56, 49, 58, 55, 70, 57, 72\},$$
$$d_j \in \{66, 71, 73, 75, 76, 76, 82, 85, 88, 88\},$$
$$w_j = \{22, 30, 42, 56, 68, 75, 75, 94, 103, 111\}.$$

Для данного примера в алгоритме ветвей и границ общее количество ветвей к рассмотрению 33 346 541, на каждом шаге (уровне дерева поиска) $j = 1, \ldots, n$ рассмотрено

$$\{58; 1\,926; 43\,059; 588\,149; 3\,820\,730; 9\,432\,283; 14\,004\,668; 4\,636\,929; 818\,539; 200\}$$

точек. При использовании свойства 1 количество ветвей можно сократить до 5 303 348, и на каждом шаге $j = 1, \ldots, n$ рассмотрено

$$\{58; 1\,666; 26\,343; 233\,762; 962\,857; 1\,457\,811; 1\,979\,532; 531\,006; 110\ 311; 2\}$$

точек.

В соответствии с результатами численных экспериментов свойство 1 позволяет сократить количество точек в Θ_j на 47%. В таблице представлены результаты численного эксперимента для примеров, где $p \in \{5, 10, 15, 20, 25, 30\}$. В колонках 1–3, представлены времена поступления, директивные сроки и веса для 10 требований. В колонке 4 представлено расписание (время начала обслуживания требований). В колонке 5 дано оптимальное значение целевой функции TWT для примера. В колонках 7 и 8 представлено количество ветвей в алгоритме ветвей и границ без использования свойства 1 — $TNN1$ — и с использованием свойства 1 — $TNN2$. В колонке 8 представлен процент рассмотренных ветвей при использовании свойства 1.

Заметим, что трудоемкость алгоритма ветвей и границ экспоненциальна, что не позволяет решать примеры размерности $n = 20$ за 60 минут на ПЭВМ Intel Core 2 Duo CPU P8600 2,4 GHz, 4 GB RAM.

Таблица. Результаты для $p \in \{5, 10, 15, 20, 25, 30\}$

r_j	d_j	w_j	Расписание	TWT	$TNN1$	$TNN2$	%
1	2	3	4	5	6	7	8
$p = 5$							
21, 0, 35, 31, 26, 11, 10, 25, 11, 14	8, 26, 35, 35, 38, 40, 42, 43, 44, 44	6, 13, 20, 20, 58, 59, 84, 90, 110, 116	50, 0, 40, 45, 30, 15, 10, 25, 20, 35	782	199344	107496	54
17, 36, 8, 34, 20, 25, 16, 39, 6, 23	19, 19, 28, 32, 32, 37, 38, 39, 43, 44	14, 21, 23, 46, 50, 60, 67, 89, 103, 107	51, 46, 11, 36, 21, 26, 16, 41, 6, 31	2227	620833	298294	48
16, 26, 10, 5, 25, 29, 11, 19, 31, 14	30, 31, 33, 34, 36, 39, 39, 43, 44, 44	9, 15, 44, 50, 52, 64, 77, 89, 91, 113	50, 45, 10, 5, 25, 30, 15, 20, 40, 35	601	196973	112646	57
35, 35, 37, 2, 22, 9, 16, 8, 35, 21	16, 22, 23, 31, 35, 35, 40, 41, 42, 44	22, 27, 31, 54, 65, 77, 84, 105, 105, 108	50, 45, 40, 2, 23, 13, 18, 8, 35, 28	2296	205273	80800	39
7, 18, 10, 31, 4, 7, 4, 20, 19, 0	11, 20, 30, 30, 33, 34, 35, 37, 41, 44	9, 9, 45, 48, 50, 62, 64, 85, 87, 117	41, 46, 10, 31, 5, 15, 20, 25, 36, 0	882	264585	207667	78
$p = 10$							
7, 65, 56, 9, 17, 39, 77, 16, 64, 79	35, 50, 56, 70, 74, 81, 86, 87, 87, 89	39, 46, 54, 55, 64, 71, 84, 97, 104, 104	7, 97, 57, 17, 27, 47, 77, 37, 67, 87	4132	100035	54247	54
53, 63, 4, 23, 26, 27, 18, 10, 49, 28	36, 36, 42, 55, 55, 68, 75, 81, 85, 88	1, 24, 26, 27, 27, 46, 58, 79, 95, 107	94, 84, 4, 24, 34, 44, 54, 14, 64, 74	1460	592976	416717	70
32, 56, 10, 4, 49, 78, 57, 42, 40, 23	41, 51, 67, 72, 83, 84, 84, 85, 87, 89	11, 28, 29, 34, 41, 48, 56, 61, 77, 85	34, 56, 14, 4, 96, 86, 66, 44, 76, 24	1972	414289	316671	76
24, 76, 5, 36, 21, 57, 70, 36, 59, 52	27, 35, 48, 69, 71, 73, 76, 79, 83, 88	9, 23, 26, 34, 71, 72, 102, 105, 113, 114	107, 97, 5, 36, 21, 57, 87, 46, 67, 77	4608	6476173	3329563	51
65, 2, 8, 3, 29, 34, 37, 33, 47, 57	16, 46, 47, 62, 64, 79, 79, 83, 83, 87	25, 31, 32, 49, 79, 90, 94, 100, 101, 108	92, 2, 12, 22, 32, 42, 52, 62, 72, 82	2690	849744	367675	43
$p = 15$							
97, 105, 22, 98, 42, 32, 114, 47, 79, 86	67, 89, 103, 112, 114, 116, 118, 130, 131, 134	5, 36, 36, 47, 47, 48, 53, 103, 112, 116	158, 143, 22, 98, 52, 37, 128, 67, 82, 113	4386	1709603	1166960	68
93, 31, 60, 55, 28, 90, 51, 29, 106, 85	50, 68, 68, 76, 86, 89, 105, 119, 122, 129	12, 13, 17, 27, 50, 52, 60, 63, 63, 90	166, 151, 136, 58, 28, 90, 73, 43, 106, 121	5719	4417446	1328448	30
54, 37, 9, 104, 93, 64, 35, 26, 20, 61	41, 109, 118, 123, 126, 127, 128, 129, 130, 133	6, 11, 20, 35, 45, 56, 59, 63, 91, 109	144, 129, 9, 114, 99, 69, 39, 54, 24, 84	1303	386234	318040	82

Таблица. (продолжение)

1	2	3	4	5	6	7	8
19, 58, 44, 22, 101, 44, 61, 27, 81, 119	65, 70, 79, 80, 88, 90, 102, 112, 122, 134	2, 18, 29, 43, 50, 56, 67, 81, 95, 110	157, 142, 52, 22, 112, 67, 82, 37, 97, 127	4610	1950750	843607	43
18, 51, 73, 14, 57, 50, 72, 57, 56 119	54, 73, 86, 89, 95, 105, 110, 116, 132, 134	1, 13, 20, 35, 44, 68, 85, 95, 96, 111	18, 155, 140, 33, 65, 50, 80, 95, 110, 125	3307	3689263	1928262	52
$p = 20$							
71, 25, 58, 69, 93, 122, 72, 123, 95, 70	102, 116, 125, 140, 142, 151, 164, 166, 166, 175	3, 6, 8, 10, 43, 51, 67, 87, 93, 113	218, 25, 58, 198, 98, 178, 78, 138, 118, 158	3924	29709961	6809732	23
83, 67, 52, 154, 92, 60, 38, 49, 65, 80	73, 103, 103, 117, 146, 156, 166, 171, 172, 175	18, 18, 47, 58, 67, 69, 77, 81, 105, 118	198, 218, 58, 178, 98, 78, 38, 118, 138, 158	10092	1849948	1203732	65
101, 41, 18, 116, 101, 42, 59, 138, 104, 140	86, 111, 117, 123, 137, 158, 166, 169, 170, 176	2, 19, 23, 24, 25, 27, 52, 59, 110, 110	201, 41, 18, 181, 101, 61, 81, 141, 121, 161	2692	2074218	884291	43
55, 84, 135, 56, 36, 98, 60, 5, 28, 43	83, 98, 106, 106, 111, 112, 130, 149, 165, 170	2, 22, 34, 37, 52, 54, 63, 72, 82, 113	188, 128, 168, 68, 48, 108, 88, 5, 28, 148	5002	758835	471049	62
145, 95, 25, 148, 40, 28, 84, 135, 46, 52	87, 87, 93, 122, 153, 155, 160, 166, 171, 178	18, 29, 36, 38, 42, 54, 75, 84, 90, 96	205, 105, 25, 185, 45, 65, 85, 145, 125, 165	7412	1288063	579338	45
$p = 25$							
5, 93, 118, 175, 140, 129, 22, 63, 20, 74	121, 129, 146, 150, 167, 168, 169, 173, 178, 211	26, 31, 32, 51, 52, 56, 84, 89, 96, 106	5, 105, 230, 205, 155, 130, 30, 80, 55, 180	8275	46808	29428	63
193, 142, 193, 194, 168, 175, 107, 186, 164, 59	152, 197, 200, 200, 218, 219, 220, 222, 222, 224	25, 33, 44, 46, 50, 55, 64, 71, 84, 97	317, 142, 292, 267, 242, 217, 107, 192, 167, 59	17845	2103722	444158	21
176, 177, 167, 25, 48, 113, 65, 80, 66, 170	100, 120, 136, 142, 152, 186, 201, 202, 205, 221	5, 17, 26, 28, 58, 81, 100, 106, 109, 118	250, 225, 175, 25, 50, 125, 75, 100, 150, 200	5221	973995	807576	83
137, 41, 14, 164, 146, 186, 74, 36, 119, 74	83, 92, 127, 132, 167, 174, 196, 208, 217, 220	5, 11, 31, 50, 58, 66, 80, 83, 114, 116	246, 41, 14, 171, 146, 221, 91, 66, 119, 196	9240	513204	278014	54

Таблица. (окончание)

1	2	3	4	5	6	7	8
138, 79, 40, 50, 109, 2, 163, 156, 132, 190	72, 105, 110, 151, 153, 169, 200, 204, 208, 221	4, 48, 53, 57, 59, 60, 60, 74, 80, 118	240, 90, 40, 65, 115, 2, 215, 165, 140, 190	3652	217856	173628	80
$p = 30$							
55, 1, 219, 107, 32, 190, 92, 17, 80, 130	119, 129, 143, 150, 180, 184, 198, 253, 255, 263	7, 9, 36, 40, 53, 84, 87, 102, 112, 112	280, 1, 250, 121, 61, 190, 151, 31, 91, 220	9333	641049	342412	53
46, 239, 103, 164, 226, 50, 209, 83, 117, 51	117, 129, 163, 166, 208, 210, 245, 264, 268, 269	19, 28, 52, 53, 79, 81, 84, 106, 110, 114	46, 316, 106, 166, 286, 76, 226, 136, 196, 256	19060	494327	325578	66
139, 13, 155, 191, 31, 182, 175, 158, 53, 101	173, 191, 192, 208, 214, 221, 227, 241, 246, 257	15, 16, 17, 31, 44, 45, 61, 80, 84, 101	139, 13, 289, 259, 43, 229, 199, 169, 73, 103	6502	620830	377214	61
191, 75, 230, 187, 26, 51, 77, 152, 148, 45	112, 119, 154, 220, 222, 230, 232, 242, 246, 266	9, 13, 26, 26, 58, 68, 88, 99, 106, 110	298, 86, 268, 208, 26, 56, 116, 178, 148, 238	6376	536269	269281	50
79, 132, 172, 203, 126, 81, 179, 178, 183, 113	166, 178, 205, 210, 219, 241, 242, 247, 254, 255	2, 13, 20, 27, 66, 83, 95, 98, 114, 119	358, 328, 298, 268, 143, 81, 208, 178, 238, 113	9216	22248262	7730427	35

6. Вычислительная сложность задачи $1|r_j, p_j = p|\sum w_j T_j$

Насколько известно, при доказательстве NP-трудности задачи теории расписаний с классическими ограничениями p_j, w_j, d_j, r_j, D_j и целевыми функциями $C_{\max}, L_{\max}, \sum w_j C_j, \sum w_j T_j, \sum w_j U_j$ используются следующие два подхода:

- сведение к исходной задаче задачи типа “Разбиения” (Одномерный ранец, 3-Разбиение), если принять, что p_j зависят от чисел b_j из задачи о Разбиении. При таком сведении количество моментов времени возможного окончания обслуживания требований не ограничено значением $O(n^2)$;
- сведение к исходной задаче задачи теории графов (например, задачи о клике), если заданы отношения предшествования между требованиями.

Однако оба эти подхода не приводят к доказательству NP-трудности задачи $1|r_j, p_j = p|\sum w_j T_j$.

В упомянутых выше подходах к доказательству NP-трудности рассматриваются частные случаи исходных задач, где структура оптимального расписания известна, см., например, [8]. Однако для задач с равными продолжительностями обслуживания требований при известной структуре оптималь-

ного расписания просто построить полиномиальной алгоритм решения и использованием динамического программирования.

Таким образом,

- **с одной стороны**, задача $1|r_j, p_j = p|\sum w_j T_j$ не содержит сложностей типа экспоненциального количества моментов времени завершения обслуживания требований или отношений предшествования,
- **с другой стороны**, для данного примера неизвестны правила предшествования, позволяющие построить полиномиальный алгоритм решения.

Предполагаем, что задача $1|r_j, p_j = p|\sum w_j T_j$ является NP-трудной, но чтобы доказать это, необходим новый подход к доказательству NP-трудности.

7. Задачи с обратными критериями оптимизации

Обычно в теории расписаний рассматриваются задачи, где необходимо найти минимум некоторой целевой функции. Например, минимизация времени завершения обслуживания всех требований является популярным критерием оптимизации. Также в классических задачах рассматриваются критерии минимизации суммарного запаздывания, минимизации количества запаздывающих требований. В этом разделе рассмотрим задачи с *обратными* критериями оптимизации, а именно: максимизация времени завершения обслуживания всех требований, максимизация суммы моментов времен завершения обслуживания требований, максимизация количества запаздывающих требований. В этих задачах допустимыми являются только активные расписания, иначе задачи были бы тривиальны.

Исследование задач с *обратными* критериями оптимизации имеет теоретическую значимость, сами задачи имеют практическую интерпретацию [9–11]. Максимальное значение времени завершения обслуживания всех требований может быть использовано, чтобы сократить множество Θ в задаче минимизации.

В задаче максимизации времени завершения обслуживания всех требований необходимо найти активное расписание π^*, при котором максимизировано значение $C_{\max}(\pi) = \max_{j=1}^{n}\{C_j\}$. Обозначим данную задачу как $1|r_j, p_j = p|\max C_{\max}$. Аналогично обозначим другие две рассматриваемые задачи как $1|r_j, p_j = p|\max\sum C_j$ и $1|r_j, p_j = p|\max\sum U_j$ — задачи максимизации суммы моментов времен завершения обслуживания требований и максимизации количества запаздывающих требований.

Представим полиномиальный алгоритм решения задач

$1|r_j, p_j = p|\max C_{\max}$ и $1|r_j, p_j = p|\max\sum C_j$ и некоторые свойства задачи $1|r_j, p_j = p|\max\sum U_j$.

7.1. Алгоритм решения задач $1|r_j, p_j = p|\max C_{\max}$ *и* $1|r_j, p_j = p|\max\sum C_j$

Обозначим работы согласно порядку $r_1 \leqslant r_2 \leqslant \ldots \leqslant r_n$. Без потери общности примем $r_1 = 0$.

Идея алгоритма заключается в следующем. В перестановке, однозначно определяющей расписание обслуживания требований, одна за одной рассматриваются позиции. Для каждой позиции выбирается одно требование j в соответствии с r_j так, чтобы сделать промежуток (простой) между временем завершения обслуживания предыдущего требования и r_j как можно больше, но меньше, чем p. Таким образом, ни одно другое требование не может быть обслужено раньше момента времени r_j.

Алгоритм 1.

1. Пусть $t := 0$ — время завершения обслуживания последнего требования, поставленного в перестановку (в расписание), т.е. требования, время обслуживания которого определено, и $N_u := N$ — множество требований, время обслуживания которых не определено. Примем также $\pi := ()$ — найденная частичная перестановка.
2. Для $l := 1$ до n выполнить
   2.1. Выбрать требование $j := \underset{i \in N_u}{\operatorname{argmax}}\{r_i, t \leqslant r_i < t + p\}$.
   2.2. Если такого требования j нет, тогда
   Выбрать требование j из множества N_u, где $r_i \leqslant t$.
   Если такого требования j нет, тогда
   $t := \min\limits_{i \in N_u}\{r_i, t < r_i\}$. Перейти к шагу 2.1.
   2.2. $\pi := (\pi, j)$. $N_u := N_u \setminus \{j\}$. $t := C_j(\pi)$.
3. π — оптимальное расписание.

Трудоемкость алгоритма 1 — $O(n)$ операций, и $O(n \log n)$ необходимо для упорядочивания работ согласно правилу $r_1 \leqslant r_2 \leqslant \ldots \leqslant r_n$.

Лемма 1. Алгоритм 1 строит оптимальное расписание для задач $1|r_j, p_j = p|\max C_{\max}$ *и* $1|r_j, p_j = p|\max \sum C_j$.

Доказательство. Предположим, что существует оптимальное расписание $\pi^* = (l, \pi_1, j, \pi_2)$, при котором первым обслуживается требование l, хотя в алгоритме 1 строится расписание, в котором первое обслуживаемое требование — l. Если $r_l \leqslant r_j$, тогда при расписании $\pi = (j, \pi_1, l, \pi_2)$ имеем $C_i(\pi) \geqslant C_i(\pi^*)$ для кажого требования $i \in N \setminus \{j\}$ и $C_{\max}(\pi) \geqslant C_{\max}(\pi^*)$.

Если $r_j < r_l$, тогда расписание π^* не является активным, так как требование 1 может быть обслужено первым, начиная с момента времени $S_1 = 0$.

Дальнейшее доказательство может быть выполнено по индукции.

7.2. Свойства задачи $1|r_j, p_j = p|\max \sum U_j$

Известно, что задача $1|r_j|\max \sum U_j$ является NP-трудной [11], но ее частный случай $1|r_j = 0|\max \sum U_j$ может быть решен за $O(n^2)$ операций [9]. Для данного частного случая существует оптимальная последовательность обслуживания требований (G, W), при которой все требования из G обслуживаются вовремя, а все требования из W запаздывают и обслуживаются в порядке неуменьшения директивных сроков.

Представим найденные свойства для задачи $1|r_j, p_j = p|\max \sum U_j$.

Свойство 12. Существует оптимальное расписание, при котором обслуживание запаздывающего требования i предшествует обслуживанию не запаздывающего требования j и $r_j < r_i$.

Подобный пример представлен на рисунке, *г*, где требования i равные 2 и 4 запаздывают, а требования j равные 1 и 3 — не запаздывают.

Свойство 13. Существует оптимальное расписание, при котором обслуживание запаздывающего требования i предшествует обслуживанию запаздывающего требования j и $d_i < d_j$.

Подобный пример представлен на рисунке, *г*, где $i = 2$, $d_2 = r_2 + p - 1$ и $j = 1$, $d_1 = r_2 + p$.

Свойство 14. Существует пример, для которого алгоритм 1 строит неоптимальное расписание.

Подобный пример представлен на рисунке, *д*, при котором только требование 2 запаздывает. А на рисунке, *е* показано, что максимальное время простоя не приводит к оптимальному расписанию, где требование 3 — единственное запаздывающее требование.

Для решения задачи предлагается следующая эвристика: если при расписании π требование j обслуживается вовремя и $S_j(\pi) > r_j$, тогда необходимо переставить это требование на более раннее время обслуживания.

Свойство 15. Если при расписании π требование j обслуживается вовремя, а требование i запаздывает и оба требования обслуживаются не ранее чем с момента времени $t \geqslant \max\{r_i, r_j\}$, тогда существует расписание π', где обслуживание требования j предшествует обслуживанию i и $\sum U_j(\pi') \geqslant \sum U_j(\pi)$.

Другими словами, если два требования обслуживаются не ранее чем с момента времени $t \geqslant \max\{r_i, r_j\}$, то при оптимальном расписании обслуживание незапаздывающего требования предшествует обслуживанию запаздывающего.

Свойство 16. Если при расписании π оба требования j и i запаздывают, $d_j \leqslant d_i$ и оба требования обслуживаются не ранее чем с момента времени $t \geqslant \max\{r_i, r_j\}$, тогда существует расписание π', где обслуживание требования j предшествует обслуживанию i и $\sum U_j(\pi') \geqslant \sum U_j(\pi)$.

Другими слова, если два запаздывающих требования обслуживаются не ранее чем с момента времени $t \geqslant \max\{r_i, r_j\}$, то при оптимальном расписании они могут быть обслужены в порядке неубывания директивных сроков.

Свойство 17. Если необходимо из двух требований j и i выбрать то, что будет обслуживаться с момента времени

$$\max\{r_i, r_j\} \leqslant t \leqslant \min\{d_j - p, d_i - p\},$$

а второе будет обслужено позже, то необходимо выбрать требование с наибольшим директивным сроком.

Свойство 18. Пусть при расписании $\pi = (\pi_1, j_1, j_2, j_3, j_4, \pi_2, j_5, \pi_3)$ *выполняется:*

$$S_{j_2}(\pi) < r_{j_5} < C_{j_2}(\pi), \quad C_{j_1} > r_{j_5} - p,\ S_{j_4} = r_{j_4} < r_{j_5} + 2p \text{ и } C_{j_5}(\pi) < d_{j_5}.$$

Тогда при расписании $\pi' = (\pi_1, j_5, j_3, j_4, \pi_2, j_1, j_2, \pi_3)$ *имеем* $F(\pi) \leqslant F(\pi')$.

См. рисунок, *ж*. Согласно этому свойству необходимо исключить представленную ситуацию перестановкой требования j_5 и требований j_2, j_3.

Свойство 19. Существует расписание, при котором любая попарная перестановка уменьшает значение целевой функции, но одновременное применение нескольких перестановок увеличивает значение целевой функции.

См. рисунок, *з*. При расписании $\pi = (4, 5, 6, 1, 2, 3, 7, 8, 9, 10)$ любая попарная перестановка,например, перестановки $4 \leftrightarrow 1$, $5 \leftrightarrow 2$, сокращает количество запаздывающих требований, но их одновременное применение $\pi = (1, 2, 3, 4, 5, 6, 7, 8, 9, 10)$ увеличивает количество запаздывающих требований.

8. Заключение

Представлены свойства задачи и алгоритм решения задачи $1|r_j, p_j = p|\sum w_j T_j$. Вопрос о статусе вычислительной сложности задачи остается открытым. В том числе авторам не известны псевдополиномиальные алгоритмы решения или схемы приближенного решения с заданной относительной погрешностью.

Предполагаем, что задача $1|r_j, p_j = p|\sum w_j T_j$ является NP-трудной, но чтобы доказать это, необходим новый подход к доказательству NP-трудности.

Также в статье представлены новые задачи теории расписаний с обратными критериями оптимизации и их свойства.

СПИСОК ЛИТЕРАТУРЫ

1. *Graham R.L., Lawler E.L., Lenstra J.K., Rinnooy Kan A.H.G.* Optimization and Approximation in Deterministic Machine Scheduling: a Survey // Ann. Discret. Math. 1979. No. 5. P. 287–326.
2. *Leung J.Y.-T.* Handbook of Scheduling. N.Y.: Chapmann and Hall/CRC, 2004.
3. *Baptiste P.* Scheduling Equal-Length Jobs on Identical Parallel Machines // Discret. Appl. Math. 2000. No. 103(1-3). P. 21–32.
4. *Verma S., Dessouky M.* Single-Machine Scheduling of Unit-Time Jobs with Earliness and Tardiness Penalties // Math. Oper. Res. 1998. No. 23(4). P. 930–943.
5. *Van den Akker J.M., Diepen G., Hoogeveen J.A.* Minimizing Total Weighted Tardiness on a Single Machine with Release Dates and Equal-Length Jobs // J. Scheduling. 2010. No. 13. P. 561–576.
6. *Kravchenko S.A., Werner F.* Parallel Machine Problems with Equal Processing Times: a Survey // J. Scheduling. 2011. No. 14. P. 435–444.
7. *Held M., Karp R.M.* A Dynamic Programming Approach to Sequencing Problems // J. Soc. Indust. Appl. Math. 1962. No. 10. P. 196–210.

8. *Du J., Leung J.Y.-T.* Minimizing Total Tardiness on One Processor is NP-hard // Math. Oper. Res. 1990. No. 15. P. 483–495.
9. *Aloulou M.A., Kovalyov M.Y., Portmann M.-C.* Evaluation Flexible Solutions in Single Machine Scheduling via Objective Function Maximization: the Study of Computational Complexity // RAIRO Oper. Res. 2007. No. 41. P. 1–18.
10. *Gafarov E.R., Lazarev A.A., Werner F.* Transforming a Pseudo-Polynomial Algorithm for the Single Machine Total Tardiness Maximization // Probl. Polynom. One. Annal. Oper. Res. 2012. No. 196(1). P. 247–261.
11. *Gafarov E.R., Lazarev A.A., Werner F.* Single Machine Total Tardiness Maximization Problems: Complexity and Algorithms // Ann. Oper. Res. 2013. No. 207. P. 121–136.

Статья представлена к публикации членом редколлегии Ф.Т. Алескеровым.

Поступила в редакцию 07.07.2019

После доработки 03.11.2019

Принята к публикации 28.11.2019

© 2020 г. И.Н. ЛУЩАКОВА, канд. физ.-мат. наук (IrinaLushchakova@yandex.ru)
(Белорусский государственный университет
информатики и радиоэлектроники, Минск)

ГЕОМЕТРИЧЕСКИЕ АЛГОРИТМЫ ОПРЕДЕЛЕНИЯ ТОЧКИ В ПЕРЕСЕЧЕНИИ ШАРОВ

Рассматривается задача определения точки в пересечении n шаров в евклидовом пространстве E^m. Для случая $m = 2$ предлагаются два алгоритма сложности $O(n^2 \log n)$ и $O(n^3)$ операций. Для общего случая предлагается точный полиномиальный рекурсивный алгоритм, использующий ортогональное преобразование пространства E^m.

Ключевые слова: пересечение шаров, аппроксимация эллипсоидами выпуклого множества, полиномиальный алгоритм, доставка с помощью дронов, конфигурация роя дронов.

DOI: 10.31857/S0005231020050098

1. Введение

В [1] была сформулирована следующая задача. В m-мерном евклидовом пространстве E^m рассматривается множество n шаров. Каждый шар $\mathcal{B}_i$, $1 \leqslant i \leqslant n$, задается указанием его центра O_i и радиуса R_i. Необходимо определить, является ли пересечение n шаров непустым множеством, и в случае положительного ответа найти точку из этого пересечения.

В [1] отмечается, что при $m = 2$ данная задача является математической моделью для следующей практической задачи. На плоскости располагаются n передающих станций различной мощности. Сигнал от станции i, расположенной в точке O_i, может быть получен на расстоянии, не превышающем R_i единиц измерения. Необходимо определить, можно ли найти место для строительства принимающей станции (принимая во внимание только характеристики дальности распространения сигналов от передающих станций) таким образом, чтобы принимающая станция могла получать сигналы от всех передающих станций. Такая техническая интерпретация может быть обобщена и на случай трехмерного пространства ($m = 3$), когда в модели учитывается высота расположения передающих и принимающей станций. Кроме того, в трехмерном случае рассматриваемой задаче можно придать различные интерпретации в контексте актуального направления, связанного с использованием непилотируемых летательных аппаратов (дронов) [2]. Например, жители небольших поселений, расположенных в труднодоступной гористой местности, могут использовать дроны для доставки лекарств, различных мелких товаров и почтовых пакетов. Стартовая площадка для дрона в населенном пункте i находится в точке O_i. С учетом технических характеристик дрон, которым владеют жители поселка i, может без дозарядки преодолеть расстояние, не превышающее $2R_i$ единиц измерения (туда и обратно). Необходимо

определить координаты точки зависания воздушного транспортного средства (вертолета, дирижабля), используемого в качестве склада товаров, таким образом, чтобы этот склад был достижим для дронов каждого поселка. Аналогичная постановка может быть рассмотрена и при планировании доставки с помощью дронов лекарств, медицинских приборов, расходных материалов и т.д. различным бригадам спасателей в районах стихийных бедствий или широкомасштабных катастроф. При этом постоянно меняющаяся обстановка может потребовать многократного эффективного решения задачи определения координат точки зависания воздушного транспортного средства-склада, достижимого для дронов всех бригад спасателей.

В [1] предлагаются два подхода к решению рассматриваемой задачи в общем m-мерном случае. Первый подход связан с задачей минимизации линейной функции с квадратичными ограничениями. Второй подход основан на известном методе эллипсоидов (см., например, [3]). Следует отметить, что рассматриваемую задачу можно отнести к широкому классу задач аппроксимации эллипсоидами выпуклого множества в пространстве E^m [4]. Этот класс задач подразделяется на задачи внутренней и внешней аппроксимации. Решение рассматриваемой задачи может быть получено из решения следующей задачи внутренней аппроксимации: найти эллипсоид наибольшего объема, содержащийся в пересечении n заданных эллипсоидов, если это пересечение непусто (естественно, для решения интересующей задачи надо рассматривать частный случай, в формулировке которого эллипсоиды заменяются на шары). В [4] демонстрируется, что последняя задача может быть сведена к задаче выпуклого программирования. Аналогичная задача внешней аппроксимации может быть сформулирована следующим образом: найти эллипсоид наименьшего объема, содержащий пересечение заданных эллипсоидов, если это пересечение непусто. Однако (в отличие от указанной выше задачи внутренней аппроксимации) данная задача внешней аппроксимации является NP-трудной [4]. В [5] рассматривается ее частный случай: найти шар наименьшего радиуса, содержащий пересечение заданных n шаров. В [5] показано, что если пересечение заданных шаров непусто и $n \leqslant m - 1$, то задача нахождения шара наименьшего радиуса может быть решена с помощью задачи минимизации выпуклой квадратичной функции.

Следует отметить, что все ранее предложенные подходы к рассмотренным задачам с теоретической точки зрения являются полиномиальными. Однако их реализация на практике либо может оказаться не очень эффективной, особенно при больших значениях n (см., например, комментарий в [6], касающийся алгоритма эллипсоидов), либо затруднена в связи с достаточно абстрактным характером используемых конструкций (см., например, в [4] сведение задачи нахождения эллипсоида наибольшего объема, содержащегося в пересечении заданных эллипсоидов, к задаче выпуклого программирования). В данной статье представлен альтернативный геометрический подход к задаче нахождения точки в пересечении n шаров из пространства E^m, результатом которого явилась разработка точных полиномиальных алгоритмов ее решения. Предлагаемый подход использует известный аппарат линейной алгебры и аналитической геометрии.

Статья организована следующим образом. В разделе 2 описаны два алгоритма решения задачи для случая $m = 2$. Более специфический алгоритм $BALLS1$ имеет вычислительную сложность $O(n^2 \log n)$ операций. Алгоритм $BALLS2$ проще по своей структуре, но имеет более высокую сложность $O(n^3)$ операций. В разделе 3 рассматривается общий m-мерный случай, для которого разработан рекурсивный алгоритм $BALLS3(m, n)$. В процессе работы алгоритм $BALLS3(m, n)$ использует либо алгоритм $BALLS1$, либо алгоритм $BALLS2$, отчего зависит его вычислительная сложность — $O(n^{2m-4}(nm^2 + m^3 + n^2 \log n))$ или $O(n^{2m-4}(nm^2 + m^3 + n^3))$ операций. Заключительные замечания представлены в разделе 4.

2. Алгоритмы определения точки в пересечении кругов

Несмотря на то, что в данном разделе будут описаны алгоритмы определения точки в пересечении кругов на плоскости, для построения первого алгоритма необходимо точки на плоскости трактовать как точки в трехмерном пространстве с нулевой третьей координатой. Поэтому введем в рассмотрение декартову прямоугольную систему координат $(O; \vec{i}, \vec{j}, \vec{k})$.

Пусть на плоскости Oxy имеется n кругов. Каждый круг B_i, $1 \leqslant i \leqslant n$, задается указанием его центра $O_i(x_i, y_i, 0)$ и радиуса r_i. Границей круга B_i является окружность C_i, определяемая уравнением $(x - x_i)^2 + (y - y_i)^2 = r_i^2$. Занумеруем круги B_i, $1 \leqslant i \leqslant n$, в порядке невозрастания их радиусов: $r_1 \geqslant \geqslant r_2 \geqslant \ldots \geqslant r_n$.

2.1. Предварительная обработка

Вначале проверим, пересекаются ли заданные круги попарно. Для каждой пары кругов B_i и B_j, $1 \leqslant i \leqslant n - 1$, $i < j \leqslant n$, вычислим расстояние d_{ij} между их центрами.

Если $d_{ij} > r_i + r_j$, то круги B_i и B_j не пересекаются, следовательно, задача не имеет решения.

Если $d_{ij} \leqslant r_i - r_j$, то круг B_j находится внутри круга B_i. Поэтому можно удалить из дальнейшего рассмотрения круг B_i и решать задачу для $n - 1$ круга.

Если $d_{ij} = r_i + r_j$, то круги B_i и B_j имеют единственную общую точку M – точку касания их границ C_i и C_j. Найдем вектор $\overrightarrow{O_iO_j}$ и нормируем его. Пусть $\overrightarrow{e_{ij}}$ – это орт вектора $\overrightarrow{O_iO_j}$. От точки O_i отложим вектор $r_i\overrightarrow{e_{ij}}$, получим точку M – точку касания кругов B_i и B_j. Остается проверить, принадлежит ли точка M всем остальным кругам. Если да, то найденная точка M является решением задачи. В противном случае задача не имеет решения.

2.2. Основная часть алгоритма $BALLS1$

В дальнейшем изложении будем без ограничения общности считать, что для любой пары кругов B_i и B_j, $1 \leqslant i \leqslant n - 1$, $i < j \leqslant n$, выполняется неравенство $r_i - r_j < d_{ij} < r_i + r_j$. Это означает, что окружности C_i и C_j пересекаются в двух точках.

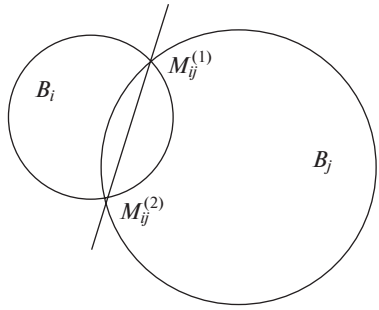


Рис. 1. Пересечение кругов B_i и B_j.

Зафиксируем одну из окружностей, например окружность C_i, и найдем точки пересечения ее с какой-либо другой окружностью C_j, $i \neq j$. Для этого решим систему уравнений:

$$(1) \qquad \begin{cases} (x - x_i)^2 + (y - y_i)^2 = r_i^2, \\ (x - x_j)^2 + (y - y_j)^2 = r_j^2. \end{cases}$$

Вычитая первое уравнение системы (1) из второго, получим уравнение вида $fx + gy + h = 0$, задающее прямую, проходящую через точки пересечения окружностей C_i и C_j. Выражая y из уравнения $fx + gy + h = 0$ и подставляя его в первое уравнение системы (1), получим квадратное уравнение вида $ax^2 + bx + c = 0$, которое имеет два различных действительных корня $x_{ij}^{(1)}$ и $x_{ij}^{(2)}$. Из уравнения $fx + gy + h = 0$ получим соответствующие значения $y_{ij}^{(1)}$ и $y_{ij}^{(2)}$ и тем самым определим точки $M_{ij}^{(1)}(x_{ij}^{(1)}, y_{ij}^{(1)}, 0)$ и $M_{ij}^{(2)}(x_{ij}^{(2)}, y_{ij}^{(2)}, 0)$ пересечения окружностей C_i и C_j. Отметим, что круги B_i и B_j гарантированно будут иметь общий отрезок $[M_{ij}^{(1)}, M_{ij}^{(2)}]$ (см. рис. 1).

Точки $M_{ij}^{(1)}$ и $M_{ij}^{(2)}$ разбивают окружность C_i на две дуги. В дальнейшем представляет интерес только та дуга, которая находится внутри круга B_j. Договоримся, что движение вдоль выбранной дуги будет осуществляться против часовой стрелки. Тем самым одна из точек $M_{ij}^{(1)}$, $M_{ij}^{(2)}$ будет выбрана в качестве начальной точки пути, а другая — в качестве конечной. Рассмотрим тройку некомпланарных векторов $\overrightarrow{O_iO_j}$, $\overrightarrow{M_{ij}^{(1)}M_{ij}^{(2)}}$, $\vec{k}$ и найдем их смешанное произведение. Если $(\overrightarrow{O_iO_j}, \overrightarrow{M_{ij}^{(1)}M_{ij}^{(2)}}, \vec{k}) > 0$, т.е. тройка векторов $\overrightarrow{O_iO_j}$, $\overrightarrow{M_{ij}^{(1)}M_{ij}^{(2)}}$, $\vec{k}$ — правая, то точка $M_{ij}^{(1)}$ будет начальной точкой пути (покрасим ее белым цветом), а точка $M_{ij}^{(2)}$ будет конечной точкой пути (покрасим ее черным цветом). Если же $(\overrightarrow{O_iO_j}, \overrightarrow{M_{ij}^{(1)}M_{ij}^{(2)}}, \vec{k}) < 0$, т.е. тройка векторов $\overrightarrow{O_iO_j}$, $\overrightarrow{M_{ij}^{(1)}M_{ij}^{(2)}}$, $\vec{k}$ — левая, то точка $M_{ij}^{(2)}$ будет начальной точкой пути (покрасим ее белым цветом), а точка $M_{ij}^{(1)}$ будет конечной точкой пути (покрасим ее черным цветом). Указание начальной (белой) и конечной (черной) точек пути

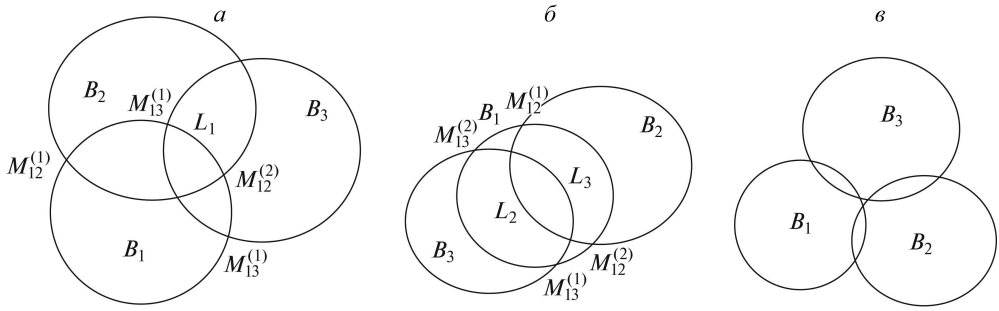


Рис. 2. Ситуации взаимного расположения трех кругов.

при договоренности о направлении обхода против часовой стрелки полностью определяет дугу окружности C_i, находящуюся внутри круга B_j. Обозначим такую дугу через L_{ij} и назовем ее подходящей дугой окружности C_i для круга B_j.

Рассмотрим последовательно все круги B_j, $1 \leqslant j \leqslant n$, $j \neq i$, и найдем множество всех подходящих дуг L_{ij} окружности C_i. Если пересечение L_i всех подходящих дуг непусто, т.е. $L_i = \bigcap_{1 \leqslant j \leqslant n, j \neq i} L_{ij} \neq \emptyset$, то дуга L_i окружности C_i будет находиться внутри каждого круга B_j, $1 \leqslant j \leqslant n$, $j \neq i$. Кроме того, хорда, соединяющая концы дуги L_i, будет находиться внутри пересечения всех кругов B_j, $1 \leqslant j \leqslant n$. Отметим, что дуга L_i является частью границы области пересечения всех кругов. Если же пересечение всех подходящих дуг пусто, т.е. $L_i = \bigcap_{1 \leqslant j \leqslant n, j \neq i} L_{ij} = \emptyset$, то либо никакая часть границы области пересечения всех кругов не принадлежит окружности C_i (т.е. пересечение всех кругов находится внутри открытого круга $B_i \setminus C_i$), либо пересечение всех кругов пусто. Если пересечение всех кругов находится внутри открытого круга $B_i \setminus C_i$, то найдется другой круг, например круг B_k, $k \neq i$, такой что часть границы области пересечения всех кругов является некоторой дугой L_k окружности C_k. Перебирая последовательно все круги B_i, $1 \leqslant i \leqslant n$, либо найдем такой круг B_k, либо придем к выводу, что пересечение всех кругов пусто.

Рассмотрим все возможные типичные ситуации взаимного расположения кругов на примере трех кругов (см. рис. 2).

1. Дуга L_1 окружности C_1 является пересечением подходящих дуг L_{12} и L_{13}, при этом дуга L_1 является частью границы области пересечения кругов B_1, B_2 и B_3.

2. Граница области пересечения кругов B_1, B_2 и B_3 не содержит какой-либо дуги окружности C_1. При этом граница области пересечения всех кругов состоит из дуг L_2 и L_3 окружностей C_2 и C_3.

3. Пересечение попарно пересекающихся кругов B_1, B_2 и B_3 пусто.

Рассмотрим вопрос об эффективном определении пересечения $L_i = \bigcap_{1 \leqslant j \leqslant n, j \neq i} L_{ij}$ всех подходящих дуг окружности C_i, $1 \leqslant i \leqslant n$.

Каждая подходящая дуга L_{ij} определяется заданием точек $M_{ij}^{(1)}(x_{ij}^{(1)}, y_{ij}^{(1)}, 0)$ и $M_{ij}^{(2)}(x_{ij}^{(2)}, y_{ij}^{(2)}, 0)$ пересечения окружностей C_i и C_j и указанием, какая

из этих точек является начальной, а какая — конечной точкой пути. Поставим в соответствие каждой точке $M_{ij}^{(l)}(x_{ij}^{(l)}, y_{ij}^{(l)}, 0)$, $1 \leqslant l \leqslant 2$, элемент $\phi_{ij}^{(l)}(x_{ij}^{(l)}, y_{ij}^{(l)}, \alpha_{ij}^{(l)})$. Индикатор $\alpha_{ij}^{(l)}$ указывает, в какой цвет покрашена точка $M_{ij}^{(l)}$. Если $M_{ij}^{(l)}$ покрашена в черный цвет, то полагаем $\alpha_{ij}^{(l)} = 1$, в противном случае $\alpha_{ij}^{(l)} = 0$. Таким образом, каждая подходящая дуга определяется заданием двух элементов $\phi_{ij}^{(1)}$ и $\phi_{ij}^{(2)}$.

Пусть для окружности C_i найдено множество всех элементов $\phi_{ij}^{(l)}$, $1 \leqslant l \leqslant 2$, $1 \leqslant j \leqslant n$, $j \neq i$. Разобьем множество элементов $\phi_{ij}^{(l)}$ на два подмножества. Если $y_{ij}^{(l)} < 0$, то отнесем элемент $\phi_{ij}^{(l)}(x_{ij}^{(l)}, y_{ij}^{(l)}, \alpha_{ij}^{(l)})$ ко множеству N_1, а если $y_{ij}^{(l)} \geqslant 0$, то отнесем $\phi_{ij}^{(l)}$ ко множеству N_2. Таким образом, множество N_1 (множество N_2) элементов $\phi_{ij}^{(l)}$ задает множество точек $M_{ij}^{(l)}$ окружности C_i, которым можно поставить во взаимно однозначное соответствие их проекции на ось Ox (естественно, совпадающие точки $M_{ij}^{(l)}$ проектируются в одинаковые точки на оси Ox).

Упорядочим множество N_1 элементов $\phi_{ij}^{(l)}$ по неубыванию $x_{ij}^{(l)}$, а множество N_2 по невозрастанию $x_{ij}^{(l)}$. Полученные последовательности элементов $\phi_{ij}^{(l)}$ обозначим соответственно как π_1 и π_2. Составим последовательность $\pi = (\pi_1, \pi_2)$, которая представляет собой перечисление начальных и конечных точек подходящих дуг L_{ij} в процессе обхода окружности C_i против часовой стрелки. Занумеруем элементы последовательности π, обозначив их через $\psi_k(x_k, y_k, \alpha_k)$, т.е. $\pi = (\psi_1, \psi_2, \ldots, \psi_{2n-2})$.

Просматривая последовательность π слева направо, выделим в ней подпоследовательности элементов, соответствующих совпадающим точкам окружности C_i, если такие существуют. Пусть $\bar{\pi} = (\psi_k, \psi_{k+1}, \ldots, \psi_{k+t})$ — одна из таких подпоследовательностей. У всех элементов $\psi_p(x_p, y_p, \alpha_p)$ подпоследовательности $\bar{\pi}$ одинаковые значения x_p и одинаковые значения y_p, а вот индикаторы α_p у них могут быть разные. Разобьем элементы $\bar{\pi}$ на две подпоследовательности: к подпоследовательности $\bar{\pi}'$ отнесем все элементы, у которых $\alpha_p = 0$, остальные элементы отнесем к подпоследовательности $\bar{\pi}''$ (у них $\alpha_p = 1$). Если у элемента ψ_{k-1}, предшествующего подпоследовательности $\bar{\pi}$ в последовательности π, индикатор $\alpha_{k-1} = 0$, то в последовательности π заменим подпоследовательность $\bar{\pi}$ подпоследовательностью $(\bar{\pi}', \bar{\pi}'')$. Если же $\alpha_{k-1} = 1$, то заменим в π подпоследовательность $\bar{\pi}$ подпоследовательностью $(\bar{\pi}'', \bar{\pi}')$. Аналогичные действия проделаем со всеми подпоследовательностями элементов, соответствующих совпадающим точкам окружности C_i. Такие построения выполняются с целью минимизации количества переключений индикатора α_k в процессе перечисления начальных и конечных точек подходящих дуг L_{ij} при обходе окружности C_i.

Перенумеруем элементы преобразованной последовательности π в соответствии с их расположением. Если у соседних элементов $\psi_k(x_k, y_k, \alpha_k)$ и $\psi_{k+1}(x_{k+1}, y_{k+1}, \alpha_{k+1})$, $1 \leqslant k \leqslant 2n-3$, последовательности π имеем $\alpha_k \neq$

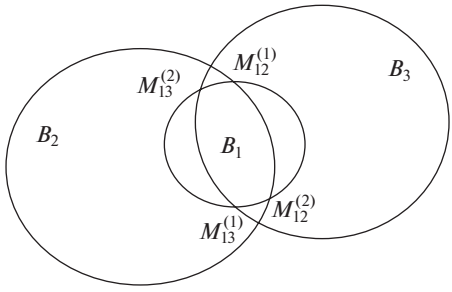


Рис. 3. Граница области пересечения кругов содержит две дуги окружности C_1.

$\neq \alpha_{k+1}$, то назовем эту ситуацию переключением индикатора. Просматривая слева направо последовательность π, подсчитаем у нее количество ρ переключений индикатора.

Лемма 1. Пусть последовательность π элементов $\psi_k(x_k, y_k, \alpha_k)$, $1 \leqslant k \leqslant 2n-2$, соответствующих начальным и конечным точкам подходящих дуг окружности C_i, построена по правилам, описанным выше. Тогда если $1 \leqslant \rho \leqslant 2$, то пересечение L_i всех подходящих дуг окружности C_i непусто, т.е. $L_i = \bigcap_{1 \leqslant j \leqslant n, j \neq i} L_{ij} \neq \emptyset$.

В Приложении приводится конструктивное доказательство леммы 1, в процессе которого показано, как найти дугу $L_i = \bigcap_{1 \leqslant j \leqslant n, j \neq i} L_{ij} \neq \emptyset$, а значит, решить задачу, поскольку любая точка на дуге L_i, а также любая точка, принадлежащая хорде, соединяющей концы L_i, принадлежит пересечению всех кругов.

Отметим, что если $\rho \geqslant 3$, то, как показывает следующий пример, никакого вывода о существовании непустого пересечения $L_i = \bigcap_{1 \leqslant j \leqslant n, j \neq i} L_{ij}$ подходящих дуг окружности C_i сделать нельзя.

Пример 1. Пусть задана последовательность индикаторов $(0, 1, 0, 1)$. Для этой последовательности количество переключений индикатора $\rho = 3$. На рис. 2,*б* и рис. 2,*в* изображены две возможные ситуации. Для каждой из этих ситуаций пересечение подходящих дуг окружности C_1 пусто. На рис. 3 изображена третья возможная ситуация, когда пересечение подходящих дуг окружности C_1 состоит из двух дуг $\smile M_{12}^{(1)} M_{13}^{(2)}$ и $\smile M_{13}^{(1)} M_{12}^{(2)}$. Заметим, что при этом градусная мера по крайней мере одной из дуг L_{12} и L_{13} больше $180°$.

Лемма 2. Градусная мера подходящей дуги L_{ij} окружности C_i больше $180°$ только в том случае, если ее радиус r_i меньше радиуса r_j окружности C_j, $j \neq i$.

Следствие 1. Если радиус r_i окружности C_i больше радиусов всех остальных окружностей C_j, $j \neq i$, и $\rho \geqslant 3$, то $L_i = \bigcap_{1 \leqslant j \leqslant n, j \neq i} L_{ij} = \emptyset$.

Напомним, что все круги B_i, $1 \leqslant i \leqslant n$, занумерованы в порядке невозрастания их радиусов: $r_1 \geqslant r_2 \geqslant \ldots \geqslant r_n$. Рассмотрим круг B_1 и определим для окружности C_1 количество ρ переключений индикатора. В соответствии с леммой 1 и следствием 1 для окружности C_1, имеющей наибольший радиус, в зависимости от значения ρ можно сделать вывод, имеет ли место ситуация $L_1 \neq \emptyset$.

Предположим, исследовали окружность C_1 и пришли к выводу, что $L_1 = \emptyset$. Если при этом пересечение всех кругов — непустое множество, то оно находится внутри открытого круга $B_1 \setminus C_1$. Временно удаляем круг B_1 и переходим к рассмотрению круга B_2, т.е. круга со следующим по величине радиусом. Процесс продолжаем до тех пор, пока не найдем круг B_k такой, что для окружности C_k имеем $L_k = \bigcap_{k<j\leqslant n} L_{kj} \neq \emptyset$. Пусть M_{kp} и M_{kq} — начальная и конечная точки дуги L_k окружности C_k. По построению точка M_{kp} (и вообще любая точка дуги L_k и любая точка хорды, соединяющей ее концы) принадлежит всем кругам $B_k, B_{k+1}, \ldots, B_n$. Остается проверить, принадлежит ли точка M_{kp} (либо какая-то из указанных точек) кругам $B_1, \ldots, B_{k-1}$. Если да, то проверяемая точка — искомая. В противном случае задача не имеет решения.

Ниже приводится формальное описание алгоритма.

*Алгоритм 1. BALLS*1

Вход: Круги $B_1, \ldots, B_n$, заданные указанием их центров $O_i(x_i, y_i, 0)$ и радиусов $r_i, i = \overline{1,n}$.

Выход: Общая точка для всех кругов $B_1, \ldots, B_n$ либо ответ, что задача не имеет решения.

1. Занумеровать круги $B_1, \ldots, B_n$ таким образом, что $r_1 \geqslant r_2 \geqslant \ldots \geqslant r_n$.
2. **FOR** $i = 1$ to n **DO**
3. $N_1 = \emptyset, N_2 = \emptyset$.
4. **FOR** $j = i + 1$ to n **DO**
5. Для окружностей C_i и C_j найти их точки пересечения $M_{ij}^{(1)}(x_{ij}^{(1)}, y_{ij}^{(1)}, 0)$ и $M_{ij}^{(2)}(x_{ij}^{(2)}, y_{ij}^{(2)}, 0)$, решив систему (1).
6. **IF** $(\overrightarrow{O_iO_j}, \overrightarrow{M_{ij}^{(1)}M_{ij}^{(2)}}, \vec{k}) > 0$ **THEN** полагаем $\alpha_{ij}^{(l)} = 0$, $\alpha_{ij}^{(2)} = 1$ (для дуги L_{ij} точка $M_{ij}^{(1)}$ — начальная, а $M_{ij}^{(2)}$ — конечная) **ELSE** полагаем $\alpha_{ij}^{(l)} = 1$, $\alpha_{ij}^{(2)} = 0$ (для дуги L_{ij} точка $M_{ij}^{(2)}$ — начальная, а $M_{ij}^{(1)}$ — конечная).
7. **FOR** $l = 1$ to 2 **DO**
8. Создать элемент $\phi_{ij}^{(l)}(x_{ij}^{(l)}, y_{ij}^{(l)}, \alpha_{ij}^{(l)})$.
9. **IF** $y_{ij}^{(l)} < 0$ **THEN** $N_1 = N_1 \cup \{\phi_{ij}^{(l)}\}$ **ELSE** $N_2 = N_2 \cup \{\phi_{ij}^{(l)}\}$. **END FOR** l

END FOR j

10. Упорядочить множество N_1 элементов $\phi_{ij}^{(l)}(x_{ij}^{(l)}, y_{ij}^{(l)}, \alpha_{ij}^{(l)})$ по неубыванию $x_{ij}^{(l)}$, а множество N_2 по невозрастанию $x_{ij}^{(l)}$. Обозначить полученные последовательности π_1 и π_2 соответственно.
11. Составить последовательность $\pi = (\pi_1, \pi_2)$. Найти в последовательности π подпоследовательности элементов, соответствующих совпадающим точкам, и переупорядочить их (см. выше описание процесса переупорядочивания).
12. Просматривая слева направо последовательность π, определить количество ρ переключений индикатора.

13. **IF** $1 \leqslant \rho \leqslant 2$, **THEN** определить дугу $L_i = \bigcap_{1 \leqslant j \leqslant n, j \neq i} L_{ij} \neq \emptyset$ (см. доказательство леммы 1), положить $k = i$ и перейти на шаг 15.
END FOR i
14. Перейти на шаг 19.
15. Определить M_{kp} и M_{kq}-начальную и конечную точки дуги L_k окружности C_k.
16. **FOR** $i = 1$ to $k - 1$ **DO**
17. **IF** $M_{kp} \notin B_i$, **THEN** перейти на шаг 19.
END FOR i
18. Точка M_{kp} — решение задачи. Стоп.
19. Задача не имеет решения. Стоп.

Упорядочение множеств N_1 и N_2 (шаг 10) представляет собой наиболее трудоемкую операцию, входящую в цикл по i (шаги 2–13). Общая вычислительная сложность алгоритма составляет $O(n^2 \log n)$ операций.

2.3. Основная часть алгоритма BALLS2

Каждый круг B_i, $1 \leqslant i \leqslant n$, представляет собой выпуклое множество точек. Известно, что пересечение конечного числа выпуклых множеств есть выпуклое множество (см., например, [6]). Граница области пересечения n кругов состоит из дуг некоторых из окружностей C_i, $1 \leqslant i \leqslant n$. Пересечение кругов фактически является выпуклой комбинацией (см. [6]) точек своей границы. Среди всех граничных точек области пересечения кругов в первую очередь представляют интерес точки стыковки дуг окружностей. Используя выпуклые комбинации этих точек, при необходимости можно будет найти и другие точки пересечения кругов.

Алгоритм просматривает все пары окружностей C_i и C_j, $1 \leqslant i, j \leqslant n$, $i \neq j$, определяет их точки пересечения $M_{ij}^{(1)}$ и $M_{ij}^{(2)}$ и проверяет, принадлежит ли хотя бы одна из них всем остальным кругам B_k, $1 \leqslant k \leqslant n$, $k \neq i, j$. В случае положительного ответа найденная точка (например, $M_{ij}^{(1)}$) — решение задачи.

Если пересечение кругов не является пустым множеством, то граница области пересечения содержит не менее двух дуг (с учетом предварительной обработки), а значит, и не менее двух точек стыковки дуг. Поэтому перебрав точки пересечения всех пар окружностей C_i и C_j, $1 \leqslant i, j \leqslant n$, $i \neq j$, алгоритм определит искомую точку.

Ниже приводится формальное описание алгоритма.

Алгоритм 2. BALLS2

Вход: Круги $B_1, \ldots, B_n$, заданные указанием их центров $O_i(x_i, y_i, 0)$ и радиусов $r_i, i = \overline{1, n}$.

Выход: Общая точка для всех кругов $B_1, \ldots, B_n$ либо ответ, что задача не имеет решения.

1. **FOR** $i = 1$ to $n - 1$ **DO**
2. **FOR** $j = i + 1$ to n **DO**
3. Для окружностей C_i и C_j найти их точки пересечения $M_{ij}^{(1)}$ и $M_{ij}^{(2)}$.

4. $l = 1$
5. **FOR** $k = 1$ to n, $k \neq i$, $k \neq j$ **DO**
6. **IF** $M_{ij}^{(l)} \notin B_k$, **THEN** перейти на шаг 8.
END FOR k
7. Точка $M_{ij}^{(l)}$ — решение задачи. Стоп.
8. $l = l + 1$
9. **IF** $l \leqslant 2$, **THEN** перейти на шаг 5.
END FOR j
END FOR i
10. Задача не имеет решения. Стоп.

Цикл по переменной i содержит вложенный цикл по переменной j, который в свою очередь содержит цикл по переменной k. Следовательно, общая вычислительная сложность алгоритма составляет $O(n^3)$ операций.

3. Определение точки в пересечении шаров в пространстве E^m

Пусть в m-мерном аффинном евклидовом пространстве E^m задана декартова прямоугольная система координат $(O, \vec{e_1}, \ldots, \vec{e_m})$, где $\vec{e_1}, \ldots, \vec{e_m}$ — ортонормированный базис. Имеется n шаров. Каждый m-мерный шар $\mathcal{B}_i$, $i = \overline{1, n}$, задается указанием его центра $O_i(x_1^{(i)}, \ldots, x_m^{(i)})$ и радиуса R_i. Границей шара $\mathcal{B}_i$ является m-мерная сфера S_i, определяемая уравнением $(x_1 - x_1^{(i)})^2 + \ldots$ $\ldots + (x_m - x_m^{(i)})^2 = R_i^2$. Предварительная обработка, т.е. проверка, пересекаются ли заданные шары попарно, проводится аналогично тому, как это было проделано для случая $m = 2$ (см. раздел 2.1) с учетом, что в аффинном евклидовом пространстве E^m расстояние между двумя точками $O_i(x_1^{(i)}, \ldots, x_m^{(i)})$ и $O_j(x_1^{(j)}, \ldots, x_m^{(j)})$ определяется по формуле

$$d_{ij} = \sqrt{(x_1^{(i)} - x_1^{(j)})^2 + \ldots + (x_m^{(i)} - x_m^{(j)})^2}.$$

В дальнейшем будем без ограничения общности считать, что для любой пары шаров $\mathcal{B}_i$ и $\mathcal{B}_j$ такой, что $R_i \geqslant R_j$, выполняется неравенство $R_i - R_j < d_{ij} <$ $< R_i + R_j$, где d_{ij} — расстояние между их центрами O_i и O_j. Это означает, что m-мерные сферы S_i и S_j пересекаются по "окружности" (т.е. по $(m-1)$-мерной сфере) C_{ij}.

Зафиксируем две сферы S_i и S_j с центрами в точках $O_i(x_1^{(i)}, \ldots, x_m^{(i)})$ и $O_j(x_1^{(j)}, \ldots, x_m^{(j)})$ соответственно и рассмотрим систему уравнений

$$\begin{cases} (x_1 - x_1^{(i)})^2 + \ldots + (x_m - x_m^{(i)})^2 = R_i^2, \\ (x_1 - x_1^{(j)})^2 + \ldots + (x_m - x_m^{(j)})^2 = R_j^2. \end{cases} \tag{2}$$

С геометрической точки зрения ее решение представляет собой $(m-1)$-мерную сферу (окружность при $m - 1 = 2$) C_{ij}, по которой пересекаются m-мер-

ные сферы S_i и S_j. Вычитая первое уравнение системы (2) из второго, получим уравнение вида

$$\alpha_1 x_1 + \ldots + \alpha_m x_m + \beta = 0, \tag{3}$$

которое представляет собой уравнение гиперплоскости в аффинном пространстве E^m. Гиперплоскость (3) как бы отрезает от шаров $\mathcal{B}_i$ и $\mathcal{B}_j$ части, из которых состоит их область пересечения, причем $(m-1)$-мерная сфера C_{ij} лежит в этой гиперплоскости и является границей "срезов" шаров $\mathcal{B}_i$ и $\mathcal{B}_j$. "Срезы" шаров $\mathcal{B}_i$ и $\mathcal{B}_j$, т.е. $(m-1)$-мерные шары (круги при $m-1=2$) с границей C_{ij}, склеены по гиперплоскости (3). В пространстве E^m $(m-1)$-мерный шар с границей C_{ij} является аналогом хорды, соединяющей точки пересечения двух окружностей (в случае $m=2$).

С помощью некоторого ортогонального преобразования декартовой системы координат (некоторого поворота системы координат вокруг ее начала в случае $m=3$) уравнение (3) можно привести к виду $x_m = x^*$, где x^* — константа. Подставляя $x_m = x^*$ в уравнение $(x_1 - x_1^{(i)})^2 + \ldots + (x_m - x_m^{(i)})^2 = R_i^2$ (первое уравнение системы (2)), получаем каноническое уравнение $(m-1)$-мерной сферы C_{ij}. Покажем, как построить матрицу T такого ортогонального преобразования (оператора).

Просматривая коэффициенты $\alpha_1, \ldots, \alpha_m$ уравнения (3), найдем коэффициент $\alpha_l \neq 0$. Зададим координаты x_i, $i = \overline{1,m}$, $i \neq l$ точек $P_0, P_1, \ldots, P_{m-1}$, принадлежащих гиперплоскости (3), в соответствии со следующей таблицей:

	x_1	x_2	$\ldots$	x_l	$\ldots$	x_m
P_0	0	0	$\ldots$	$\tilde{x}_l^{(0)}$	$\ldots$	0
P_1	1	0	$\ldots$	$\tilde{x}_l^{(1)}$	$\ldots$	0
P_2	0	1	$\ldots$	$\tilde{x}_l^{(2)}$	$\ldots$	0
$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$
P_{m-1}	0	0	$\ldots$	$\tilde{x}_l^{(m-1)}$	$\ldots$	1

Координата x_l каждой из этих точек находится из уравнения (3).

Система векторов $\vec{f}_1 = \overrightarrow{P_0P_1}, \ldots, \vec{f}_{m-1} = \overrightarrow{P_0P_{m-1}}$ линейно независима, причем каждый из этих векторов ортогонален вектору $\vec{f}_m = (\alpha_1, \ldots, \alpha_m)$. Таким образом, система векторов $\vec{f}_1, \ldots, \vec{f}_m$ может служить базисом в пространстве E^m. Для системы векторов $\vec{f}_1, \ldots \vec{f}_m$ выполним процесс ортогонализации:

$$\vec{h}_1 = \frac{\vec{f}_1}{|\,\vec{f}_1\,|};$$

$$\vec{g}_i = \vec{f}_i - (\vec{f}_i, \vec{h}_1)\vec{h}_1 - \ldots - (\vec{f}_i, \vec{h}_{i-1})\vec{h}_{i-1}, \vec{h}_i = \frac{\vec{g}_i}{|\,\vec{g}_i\,|}, \quad i = 2, \ldots, m-1;$$

$$\vec{h}_m = \frac{\vec{f}_m}{|\,\vec{f}_m\,|}.$$

В итоге получили ортонормированный базис $\vec{h}_1, \ldots, \vec{h}_m$. Составим матрицу T перехода от исходного ортонормированного базиса $\vec{e}_1, \ldots, \vec{e}_m$ к базису $\vec{h}_1, \ldots, \vec{h}_m$, записав в столбцы матрицы T координаты векторов $\vec{h}_1, \ldots, \vec{h}_m$ в базисе $\vec{e}_1, \ldots, \vec{e}_m$. Матрица T также является матрицей ортогонального оператора, переводящего гиперплоскость (3) в гиперплоскость вида $x_m = x^*$.

Пересчитаем координаты центров O_k шаров $\mathcal{B}_k$, $k = \overline{1,n}$, по формуле

$$\widetilde{X}^{(k)} = T^{-1} X^{(k)}, \tag{4}$$

где $X^{(k)} = (x_1^{(k)}, \ldots, x_m^{(k)})^T$ — вектор-столбец координат точки O_k в базисе $\vec{e}_1, \ldots, \vec{e}_m$, а $\widetilde{X}^{(k)} = (\tilde{x}_1^{(k)}, \ldots, \tilde{x}_m^{(k)})^T$ — вектор-столбец ее координат в базисе $\vec{h}_1, \ldots, \vec{h}_m$.

Для сфер S_i и S_j рассмотрим систему уравнений (2) с новыми координатами их центров O_i и O_j. Вычитая первое уравнение системы (2) из второго, получим уравнение $x_m = x^*$.

Далее, для каждой сферы S_k, $k = \overline{1,n}$, $k \neq j$, рассмотрим ее уравнение

$$(x_1 - \tilde{x}_1^{(k)})^2 + \ldots + (x_m - \tilde{x}_m^{(k)})^2 = R_k^2 \tag{5}$$

и подставим в него $x_m = x^*$. Если $R_k^2 - (x^* - \tilde{x}_m^{(k)})^2 \geqslant 0$, то полагаем

$$r_k^2 = R_k^2 - (x^* - \tilde{x}_m^{(k)})^2. \tag{6}$$

Это будет означать, что m-мерная сфера S_k пересекается с плоскостью $x_m = x^*$ и в результате получается $(m-1)$-мерная сфера (окружность при $m - 1 = 2$), задаваемая уравнением

$$(x_1 - \tilde{x}_1^{(k)})^2 + \ldots + (x_{m-1} - \tilde{x}_{m-1}^{(k)})^2 = r_k^2.$$

Если все сферы S_k (а значит, и все шары $\mathcal{B}_k$), $k = \overline{1,n}$, $k \neq j$, пересекаются с гиперплоскостью $x_m = x^*$, то на данном этапе задача сводится к определению точки в пересечении множества $n-1$ шаров в пространстве E^{m-1}. Если такая точка $M_0(\tilde{x}_1^0, \ldots, \tilde{x}_{m-1}^0, x^*)$ будет найдена, то остается только пересчитать ее координаты в соответствии со следующей формулой:

$$X^0 = T\widetilde{X}^0, \tag{7}$$

где $\widetilde{X}^0 = (\tilde{x}_1^0, \ldots, \tilde{x}_{m-1}^0, x^*)^T$ — координаты точки M_0 в базисе $\vec{h}_1, \ldots, \vec{h}_m$, а $X^0 = (x_1^0, \ldots, x_m^0)^T$ — ее координаты в исходном базисе $\vec{e}_1, \ldots, \vec{e}_m$. Формула (7) задает преобразование координат, обратное преобразованию, определяемому формулой (4).

Если же полученное на данном этапе множество $n-1$ шаров из пространства E^{m-1}, определяемого гиперплоскостью $x_m = x^*$, не пересекается, то переходим к следующему этапу алгоритма, т.е. к рассмотрению следующей фиксированной пары сфер S_i и S_j. Переход к следующему этапу алгоритма

осуществляется и в случае, когда не все сферы S_k, $k = \overline{1,n}$, $k \neq j$, пересекаются с гиперплоскостью $x_m = x^*$, т.е. если для некоторой сферы S_k имеем $R_k^2 - (x^* - \tilde{x}_m^{(k)})^2 < 0$.

Вообще, если пересечение шаров не является пустым множеством, то граница области пересечения состоит из частей некоторых ограничивающих их сфер (с учетом предварительной обработки таких сфер будет не менее двух). Гиперплоскости, по которым стыкуются части сфер, ограничивающих область пересечения шаров, проходят через область пересечения шаров. Перебирая все такие гиперплоскости (т.е. все пары m-мерных сфер S_i и S_j, $1 \leqslant i, j \leqslant n$, $i \neq j$) найдем гиперплоскость, содержащую искомую точку.

Ниже приводится краткое формальное описание рассмотренных выше основных этапов рекурсивного алгоритма $BALLS3(m, n)$, решающего задачу в пространстве E^m, где $m \geqslant 3$.

Алгоритм 3. $BALLS3(m, n)$

Вход: m-мерные шары $\mathcal{B}_1^{(m)}, \ldots, \mathcal{B}_n^{(m)}$, заданные указанием их центров $O_i^{(m)}(x_1^{(i)}, \ldots, x_m^{(i)})$ и радиусов $R_i^{(m)}$, $i = \overline{1,n}$.

Выход: Общая точка для всех шаров $\mathcal{B}_1^{(m)}, \ldots, \mathcal{B}_n^{(m)}$ либо ответ, что задача не имеет решения.

1. Выполнить предварительную обработку.
2. **FOR** $i = 1$ to $n - 1$ **DO**
3. $j = 2$.
4. **FOR** $j \leqslant n$ **DO**
5. Из системы (2) определить гиперплоскость $\alpha_1 x_1 + \ldots + \alpha_m x_m + \beta = 0$, содержащую пересечение сфер $S_i^{(m)}$ и $S_j^{(m)}$.
6. Построить матрицу T ортогонального преобразования, переводящего гиперплоскость $\alpha_1 x_1 + \ldots + \alpha_m x_m + \beta = 0$ в гиперплоскость $x_m = x^*$.
7. Найти матрицу T^{-1}.
8. Ортогональное преобразование: для всех шаров $\mathcal{B}_k^{(m)}$, $k = \overline{1,n}$, $k \neq i$, пересчитать координаты их центров $O_k^{(m)}$ по формуле (4).
9. Из системы (2) уравнений сфер с новыми координатами их центров определить плоскость $x_m = x^*$, содержащую пересечение сфер $S_i^{(m)}$ и $S_j^{(m)}$.
10. **FOR** $k = 1$ to n, $k \neq j$ **DO**
11. **IF** гиперплоскость $x_m = x^*$ не пересекает сферу $S_k^{(m)}$ **THEN** перейти на шаг 14 **ELSE** для $(m-1)$-мерного шара $\mathcal{B}_k^{(m-1)}$ определить его радиус $R_k^{(m-1)} = r_k$ по формуле (6) и центр $O_k^{(m-1)}$ (отбрасывая последнюю координату у точки $O_k^{(m)}$). **END FOR** k
12. **IF** $m > 3$ **THEN** для множества шаров $\mathcal{B}_k^{(m-1)}$, $1 \leqslant k \leqslant n$, $k \neq j$, выполнить $BALLS3(m-1, n-1)$ **ELSE** выполнить для этого множества шаров $BALLS1$ или $BALLS2$.
13. **IF** найдена точка $M_0(\tilde{x}_1^0, \ldots, \tilde{x}_{m-1}^0, x^*)$ в пересечении $(m-1)$-мерных

шаров $\mathcal{B}_k^{(m-1)}$, $1 \leqslant k \leqslant n$, $k \neq j$ **THEN** обратное ортогональное преобразование: пересчитать координаты точки M_0 по формуле (7) и стоп.

14. Положить $j = j + 1$ и перейти на шаг 4.

END FOR j

END FOR i

15. Задача не имеет решения. Стоп.

Оценим вычислительную сложность алгоритма $BALLS3(m, n)$. В пространстве E^m расстояние d_{ij} между двумя точками O_i и O_j может быть вычислено за $O(m)$ операций. Поэтому предварительная обработка (шаг 1) требует $O(n^2 m)$ операций. Шаги 5 и 9 выполняются за $O(m)$ операций.

Построение системы векторов $\vec{f}_1, \ldots, \vec{f}_m$ может быть выполнено за $O(m^2)$ операций (см. таблицу), а процесс ортогонализации системы векторов — за $O(m^3)$ операций (с учетом того, что скалярное произведение двух векторов вычисляется за $O(m)$ операций). Следовательно, матрица T, столбцами которой являются координаты ортонормированной системы векторов $\vec{h}_1, \ldots \vec{h}_m$, может быть построена за $O(m^3)$ операций (шаг 6). Обратная матрица T^{-1} может быть построена за $O(m^2)$ операций (шаг 7) с помощью следующего известного метода, основанного на методе Гаусса. К расширенной матрице $[T|I_m]$, где I_m — единичная матрица порядка m, применяются элементарные преобразования над строками, приводящие эту матрицу к виду $[I_m|C]$. В итоге получается матрица $T^{-1} = C$.

Вычисление координат точки в новом базисе по формуле (4) может быть выполнено за $O(m^2)$ операций. Следовательно, шаг 8 выполняется за $O(nm^2)$ операций. Цикл по k (шаги 10–11) выполняется за $O(n)$ операций. Пересчет координат точки M_0 по формуле (7) (шаг 13) выполняется за $O(m^2)$ операций.

С учетом двух основных циклов — по i (шаги 2–14) и по j (шаги 4–14) — общая вычислительная сложность $\mathcal{T}(m, n)$ алгоритма $BALLS3(m, n)$ может быть выражена следующим рекуррентным уравнением:

$$\mathcal{T}(m, n) = O(n^3 m^2 + n^2 m^3) + n^2 \mathcal{T}(m - 1, n - 1), \quad m \geqslant 3, \tag{8}$$

где $\mathcal{T}(2, n) = O(n^2 \log n)$, если используется алгоритм $BALLS1$, и $\mathcal{T}(2, n) = O(n^3)$, если используется алгоритм $BALLS2$. Решение $\mathcal{T}(m, n)$ рекуррентного уравнения (8) при $m \geqslant 3$ не превосходит $O(n^{2m-4}(nm^2 + m^3 + \mathcal{T}(2, n)))$. Следовательно, в пространстве E^m исходная задача может быть решена за $O(n^{2m-4}(nm^2 + m^3 + n^2 \log n))$ или $O(n^{2m-4}(nm^2 + m^3 + n^3))$ операций. В частности, для трехмерного пространства сложность алгоритма составит $O(n^4 \log n)$ либо $O(n^5)$ операций.

4. Заключение

В статье предлагаются точные полиномиальные алгоритмы для определения некоторой точки, принадлежащей пересечению n шаров в m-мерном евклидовом пространстве. В практически значимых случаях $m = 2$ и $m = 3$ представленные алгоритмы могут быть использованы при разработке программного обеспечения в задачах управления различными динамическими

системами, включающими в себя множество дронов. Например, для заданной конфигурации роя дронов необходимо определить местоположение управляющего дрона либо определить, возможно ли желаемое изменение конфигурации без потери управления всеми дронами управляющим дроном. Другие практические интерпретации рассматривались в разделе 1.

Отметим также, что можно выполнить небольшую модификацию представленных алгоритмов таким образом, что будет найдена не одна точка, принадлежащая пересечению шаров, а несколько таких точек (если это пересечение непусто и не состоит из единственной точки). Значит, возможно определить и линейную комбинацию этих точек, целиком принадлежащую области пересечения шаров. Заметим, что такая линейная комбинация угловых точек границы области пересечения шаров не совпадает с множеством точек шара наибольшего объема, содержащегося в области пересечения шаров, который необходимо найти в задаче внутренней аппроксимации [4]. Поэтому предложенный в данной статье подход является альтернативным не только по методу, но и по множеству получаемых решений, что предоставляет дополнительные возможности при принятии решений.

ПРИЛОЖЕНИЕ

Доказательство леммы 1. 1. Пусть $\rho = 1$, т.е. последовательность π имеет единственное переключение индикатора. Заметим, что в этом случае переключение индикатора произошло на элементе с номером $n-1$, т.е. $\alpha_{n-1} \neq \alpha_n$. Возможны две ситуации.

а) $\alpha_{n-1} = 0,\ \alpha_n = 1$. В этом случае последовательность π может быть разбита на две подпоследовательности: $\pi = (\pi^{(1)}, \pi^{(2)})$, где $\pi^{(1)} = (\psi_1, \psi_2, \ldots$ $\ldots, \psi_{n-1}),\ \pi^{(2)} = (\psi_n, \psi_{n+1}, \ldots, \psi_{2n-2})$. Все элементы подпоследовательности $\pi^{(1)}$ имеют нулевой индикатор, т.е. они соответствуют белым точкам на окружности C_i (начальным точкам подходящих дуг). Все элементы подпоследовательности $\pi^{(2)}$ имеют индикатор 1, т.е. они соответствуют черным точкам на окружности C_i (конечным точкам подходящих дуг). Любая подходящая дуга L_{ij} окружности C_i начинается в белой точке и заканчивается в черной точке при договоренности, что движение вдоль дуги происходит против часовой стрелки. Последовательность π соответствует перечислению точек на окружности C_i в процессе обхода ее против часовой стрелки. Поэтому обходу любой дуги L_{ij} от начальной до конечной точки соответствует некоторая подпоследовательность $(\psi_p, \psi_{p+1}, \ldots, \psi_{n-1}, \psi_n, \psi_{n+1}, \ldots, \psi_q)$ последовательности π, где $(\psi_p, \psi_{p+1}, \ldots, \psi_{n-1})$ — последовательность элементов с индикатором 0, а $(\psi_n, \psi_{n+1}, \ldots, \psi_q)$ — последовательность элементов с индикатором 1. Заметим, что каковы бы ни были номера p и q, $1 \leqslant p \leqslant n-1$, $n \leqslant q \leqslant 2n-2$, в любую такую подпоследовательность обязательно войдут элементы ψ_{n-1} и ψ_n. Поэтому дуга L_i, соединяющая точки окружности C_i, соответствующие элементам ψ_{n-1} (начальная точка) и ψ_n (конечная точка), будет содержаться в любой подходящей дуге L_{ij}, т.е. $L_i = \bigcap_{1 \leqslant j \leqslant n, j \neq i} L_{ij} \neq \emptyset$.

б) $\alpha_{n-1} = 1,\ \alpha_n = 0$. В этом случае все элементы подпоследовательности $\pi^{(1)}$ имеют индикатор 1, а все элементы подпоследовательности $\pi^{(2)}$ имеют индикатор 0. Обход окружности C_i против часовой стрелки может быть осуществлен в порядке $\pi^* = (\pi^{(2)}, \pi^{(1)}) = (\psi_n, \psi_{n+1}, \ldots, \psi_{2n-2}, \psi_1, \psi_2, \ldots$

$\dots, \psi_{n-1})$. Тогда обходу любой подходящей дуги L_{ij} от начальной до конечной точки соответствует некоторая подпоследовательность $(\psi_p, \psi_{p+1}, \dots$ $\dots, \psi_{2n-2}, \psi_1, \psi_2, \dots, \psi_q)$ последовательности π^*, где $(\psi_p, \psi_{p+1}, \dots, \psi_{2n-2})$ — подпоследовательность элементов с индикатором 0, а $(\psi_1, \psi_2, \dots, \psi_q)$ — подпоследовательность элементов с индикатором 1. Каковы бы ни были номера p и q, $n \leqslant p \leqslant 2n-2$, $1 \leqslant q \leqslant n-1$, в любую такую подпоследовательность $(\psi_p, \psi_{p+1}, \dots, \psi_{2n-2}, \psi_1, \psi_2, \dots, \psi_q)$ обязательно войдут элементы ψ_{2n-2} и ψ_1. Поэтому дуга L_i, соединяющая точки окружности C_i, соответствующие элементам ψ_{2n-2} и ψ_1, будет содержаться в любой подходящей дуге, т.е. $L_i = \bigcap_{1 \leqslant j \leqslant n, j \neq i} L_{ij} \neq \emptyset$.

2. Пусть $\rho = 2$, т.е. переключение индикатора происходит дважды: один раз с 0 на 1, а другой раз с 1 на 0, причем порядок этих переключений может быть произвольным. Следовательно, возможны две ситуации.

а) Пусть первое переключение индикатора присходит с 0 на 1 на элементе с номером k, т.е. $\alpha_k = 0$, $\alpha_{k+1} = 1$. В этом случае последовательность π может быть разбита на три подпоследовательности: $\pi = (\pi^{(1)}, \pi^{(2)}, \pi^{(3)})$, где $\pi^{(1)} =$ $= (\psi_1, \psi_2, \dots, \psi_k)$, $\pi^{(2)} = (\psi_{k+1}, \psi_{k+2}, \dots, \psi_{k+n-1})$, $\pi^{(3)} = (\psi_{k+n}, \psi_{k+n+1}, \dots$ $\dots, \psi_{2n-2})$, $1 \leqslant k \leqslant n-2$. Все элементы подпоследовательностей $\pi^{(1)}$ и $\pi^{(3)}$ имеют индикатор 0, а все элементы подпоследовательности $\pi^{(2)}$ имеют индикатор 1. Обход окружности C_i против часовой стрелки может быть осуществлен в порядке $\hat{\pi} = (\pi^{(3)}, \pi^{(1)}, \pi^{(2)})$. Тогда обходу любой подходящей дуги L_{ij} от начальной до конечной точек соответствует некоторая подпоследовательность $(\psi_p, \dots, \psi_k, \psi_{k+1}, \dots, \psi_q)$, где $(\psi_p, , \dots, \psi_k)$ — подпоследовательность элементов с индикатором 0, а $(\psi_{k+1}, \dots, \psi_q)$ — подпоследовательность элементов с индикатором 1, $k+n \leqslant p \leqslant 2n-2$, $1 \leqslant p \leqslant k$, $k+1 \leqslant q \leqslant k+n-1$, $1 \leqslant k \leqslant n-2$. В любую такую подпоследовательность $(\psi_p, \dots, \psi_k, \psi_{k+1}, \dots, \psi_q)$ обязательно войдут элементы ψ_k и ψ_{k+1}. Поэтому дуга L_i, соединяющая точки, соответствующие элементам ψ_k и ψ_{k+1}, будет содержаться в любой подходящей дуге, т.е. $L_i = \bigcap_{1 \leqslant j \leqslant n, j \neq i} L_{ij} \neq \emptyset$.

б) Пусть первое переключение индикатора происходит с 1 на 0 на элементе с номером k, т.е. $\alpha_k = 1$, $\alpha_{k+1} = 0$. Как и в случае 2а, последовательность π может быть разбита на те же три подпоследовательности $\pi = (\pi^{(1)}, \pi^{(2)}, \pi^{(3)})$. Однако в данном случае все элементы подпоследовательностей $\pi^{(1)}$ и $\pi^{(3)}$ имеют индикатор 1, а все элементы подпоследовательности $\pi^{(2)}$ имеют индикатор 0. Обход окружности C_i против часовой стрелки может быть осуществлен в порядке $\tilde{\pi} = (\pi^{(2)}, \pi^{(3)}, \pi^{(1)})$. Обходу любой подходящей дуги L_{ij} от начальной до конечной точек соответствует некоторая подпоследовательность $(\psi_p, \dots, \psi_{k+n-1}, \psi_{k+n}, \dots, \psi_q)$, где $(\psi_p, , \dots, \psi_{k+n-1})$ — подпоследовательность элементов с индикатором 0, а $(\psi_{k+n}, \dots, \psi_q)$ — подпоследовательность элементов с индикатором 1, $k+1 \leqslant p \leqslant k+n-1$, $k+n \leqslant q \leqslant 2n-2$, $1 \leqslant q \leqslant k$, $1 \leqslant k \leqslant 2n-2$. В любую такую подпоследовательность $(\psi_p, \dots$ $\dots, \psi_{k+n-1}, \psi_{k+n}, \dots, \psi_q)$ обязательно войдут элементы ψ_{k+n-1} и ψ_{k+n}. Поэтому дуга L_i, соединяющая точки, соответствующие элементам ψ_{k+n-1} и ψ_{k+n}, будет содержаться в любой подходящей дуге, т.е. $L_i = \bigcap_{1 \leqslant j \leqslant n, j \neq i} L_{ij} \neq$ $\neq \emptyset$.

Лемма 1 доказана.

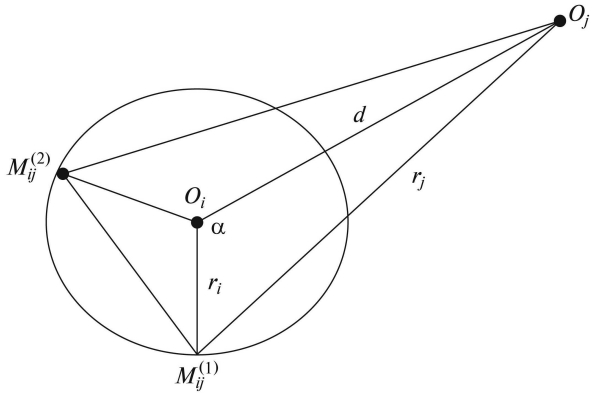


Рис. 4.

Доказательство леммы 2. Пусть градусная мера подходящей дуги L_{ij} равна $2\alpha > 180°$. Для определенности будем считать точку $M_{ij}^{(1)}$ началом дуги L_{ij}, а точку $M_{ij}^{(2)}$ — ее концом. Градусная мера подходящей дуги L_{ij} равна сумме двух одинаковых тупых углов $M_{ij}^{(1)}O_iO_j$ и $M_{ij}^{(2)}O_iO_j$ (см. рис. 4). Пусть d — это расстояние между центрами O_i и O_j окружностей C_i и C_j. Тогда из треугольника $M_{ij}^{(1)}O_iO_j$ по теореме косинусов имеем $r_j^2 = r_i^2 + d^2 - 2r_i d\cos\alpha$. Поскольку α — тупой угол, то $\cos\alpha < 0$, следовательно, $r_j > r_i$, что и требовалось доказать.

СПИСОК ЛИТЕРАТУРЫ

1. *Баркетов М.С.* Полиномиальные методы определения точки в пересечении некоторого числа шаров // Танаевские чтения. Докл. Восьмой междунар. научн. конф. 27–30 марта 2018 г. ОИПИ, Минск. 2018. С. 18–22.
2. *Otto A., Agatz N., Campbell J., Golden B., Pesch E.* Optimization approaches for civil applications of unmanned aerial vehicles (UAVs) or aerial drones: A survey / Networks, Wiley Periodicals. 2018. V.72. P. 411–458.
3. *Пападимитриу Х., Стайглиц К.* Комбинаторная оптимизация: алгоритмы и сложность. М.: Мир, 1985.
4. *Boyd S.S., El Ghaoui L., Feron E., Balakrishnan V.* Linear matrix inequalities in system and control theory // SIAM Studies Appl. Math. 1994. V. 15. Philadelphia: PA.
5. *Beck A.* On the convexity of a class of quadratic mappings and its application to the problem of finding the smallest ball enclosing a given intersection of balls // J. Global Optim., Elsevier Publ. 2007. V. 39. No. 1. P. 113–126.
6. *Мину М.* Математическое программирование. Теория и алгоритмы. М.: Наука, 1990.

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

Поступила в редакцию 18.07.2019
После доработки 15.09.2019
Принята к публикации 28.11.2019

© 2020 г. А.И. ПОТЕХИН, канд. техн. наук (an_pot@mail.ru)
(Институт проблем управления им. В.А. Трапезникова РАН, Москва)

ЛОГИЧЕСКИЕ ОСНОВЫ УПРАВЛЕНИЯ ГРУППОВЫМ ДВИЖЕНИЕМ ПОЕЗДОВ

Показано применение методов математической логики при проектировании системы управления групповым движением поездов железнодорожной станции. Разработан типовой жизненный цикл маршрута (проектирование маршрута, сборка, контроль движения поезда по маршруту, разборка маршрута), гарантирующий безопасное групповое движение поездов на станции. Разработаны модель маршрута в виде логической функции проходимости и модель железнодорожной станции в виде логической схемы. На основе этих моделей находится множество потенциально возможных станционных маршрутов, определяются состояния стрелочных переводов, светофоров маршрута. Задаются различные отношения между маршрутами: совместимые, несовместимые, альтернативные.

Ключевые слова: модель железнодорожной станции, логическая модель маршрута, жизненный цикл маршрута, совместимые, несовместимые, альтернативные маршруты.

DOI: 10.31857/S0005231020050104

1. Введение

В работе [1] предложен подход к логическому управлению технологическими процессами. Он основан на обследовании текущего состояния структуры технологических потоков по их логической модели. При этом применяется принцип управления с обратной связью по отклонению текущего состояния структуры потоков от требуемого состояния. Это существенно расширяет область применения систем логического управления. Традиционно система логического управления представлялась в виде совокупности жесткого логического алгоритма (логическая схема, конечный автомат) и объекта управления в виде множества параметров объекта. Следствием этого задачами логического управления, как правило, были управление режимами пуска, останова, защиты, блокировки и т.д., т.е. те алгоритмы, которые были постоянными вне зависимости от технологических процессов. В то время как было необходимо решать задачи управления конфигурацией объекта, целенаправленным изменением состава активных элементов объекта, связей между ними, управлением структурой технологических потоков.

В [2] этот подход был использован при разработке событийного логического управления производственными процессами поточного типа. Разработаны принципы построения системы логического управления технологической системой на основе событийных логических моделей технологических процессов, модели структуры производства, элементов транспортной сети.

Это направление получило название «управление по логическим моделям». Управление по логическим моделям позволяет создавать качественное программное обеспечение на всех этапах его жизненного цикла (на этапах проектирования, тестирования, верификации и т.д.). В этой статье рассмотрена попытка применения «управления по логическим моделям» для безопасного управления групповым движением поездов на железнодорожной станции.

Увеличение нагрузки на железнодорожный транспорт, повышение требований к безопасности движения при ограниченных возможностях развития железнодорожной сети требуют все более эффективного использования ресурсов существующей инфраструктуры. Главные требования — обеспечение безопасности и выполнение графика движения поездов. При этом задача оперативного управления осложняется тем, что:

— большой объем информации для принятия решений;

— огромное количество инструкций, правил и исключений.

Поэтому принимаемые станционным диспетчером (дежурным по станции) решения связаны с риском в прогнозном анализе развития текущей ситуации. В случае серьезных нарушений у диспетчера отсутствуют эффективные средства обеспечения безопасности и выполнения графика движения поездов. В общем случае управление железнодорожной системой состоит из двух частей: оперативное планирование движения поездов в условиях разнообразных помех и оперативное управление компонентами железнодорожной системы с целью реализации плана движения поездов. В работе [3] рассмотрены некоторые пионерские зарубежные проекты по созданию интеллектуальных систем оперативного планирования движением поездов в условиях разнообразных помех. В их основе лежит вычисление приближающихся конфликтов, нахождение оптимальных решений группового движения поездов путем перепланировки планов движения каждого поезда в режиме реального времени. Оперативное управление компонентами железнодорожной системы направлено на автоматическую реализацию функций управления элементами железнодорожных объектов (стрелками, светофорами, сборкой-разборкой маршрутов). С этой целью в работе [4] разработаны логические модели элементов инфраструктуры типовой станции (стрелочного перевода, последовательности стрелок, маршрута).

В данной работе исследуется применение методов математической логики для управления групповым движением поездов железнодорожной станции: поиск альтернативных маршрутов, выбор маршрута, совместимого с действующими маршрутами, проектирование жизненного цикла типового маршрута (проектирование маршрута, сборка, движения по маршруту, разборка маршрута).

2. Железнодорожная станция и маршруты

Станцией называется «раздельный пункт» с путевым развитием и устройствами, позволяющими выполнять операции по приему, отправлению, скрещению и обгону поездов, а также по приему, погрузке, выгрузке и выдаче грузов и по обслуживанию пассажиров. Структура путей станции состоит из множества так называемых блок-участков, стрелочных переводов (стрелок),

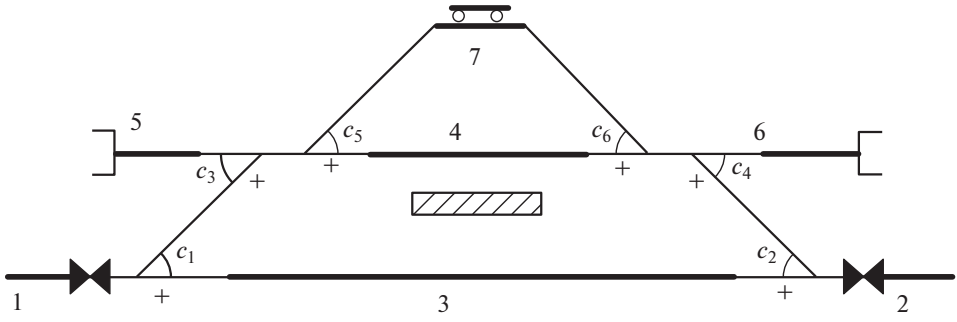


Рис. 1. Схема «промежуточной» железнодорожной станции.

соединительных рельсовых участков и светофоров. Блок-участок — это электрически изолированный рельсовый участок, состояние которого (свободен – занят) контролируется и управляется автоматически или диспетчером и дублируется состоянием светофоров, расположенных на границах блок-участка. Каждый блок-участок соединяется с помощью соединительных участков с другими блок-участками, а также — с электрически изолированными стрелочными переводами. Длина блок-участка должна быть больше самого длинного поезда. Стрелочный перевод (стрелка) служит для соединения рельсовых путей и изменения направления движения поездов. По количеству и расположению в топологическом плане пересекающихся путей стрелочные переводы могут быть обыкновенными, перекрестными и т.д. С помощью переводного механизма обыкновенная стрелка может находиться в одном из двух направлений — прямом и боковом.

На рис. 1 изображена схема так называемой промежуточной станции. Рассматривается упрощенная схема станции, она не содержит некоторых элементов инфраструктуры действующих станций (глухих пересечений, двойных стрелочных переводов и др.). Жирными линиями изображены блок-участки именных путей: 1, 2 – вход-выходные пути станции, 3 – главный путь, 4 – приемоотправочный путь, 5 и 6 – вытяжные пути (тупики), 7 – погрузочно-разгрузочный путь. Обыкновенные стрелки обозначены как $c_1, \ldots, c_6$. Прямоугольником обозначена платформа посадки и высадки пассажиров (путь 4) и место погрузки – разгрузки (путь 7). Светофоры не показаны. Несмотря на различные функциональные назначения станций, модель железнодорожной станции должна сохранить основные внешние свойства, как то: топологическую схему путей, соединительных рельсовых участков и стрелочных переводов (стрелок). С учетом этого исходную схему железнодорожной станции представим в виде неориентированного графа. Это преобразование достаточно простое, покажем его на данном примере.

На рис. 2 железнодорожная станция показана в виде «станционного» графа. Ребра и вершины «станционного» графа определим следующим образом:

— ребра обозначены как $b_1, \ldots, b_7$, они соответствуют блок-участкам станции.

Окончания участков обозначены цифрами 1 и 2 (произвольно). Ребро, соответствующее блок-участку b_i, в тексте будем обозначать как (b_i^1, b_i^2) или

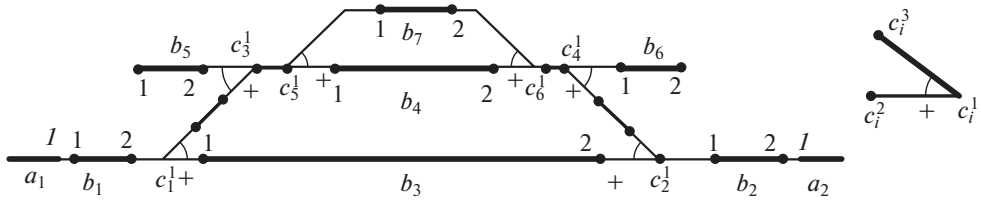


Рис. 2. Графическое представление железнодорожной станции.

(b_i^2, b_i^1) в зависимости от направления движения поезда. Блок-участки приближения к станции (блок-участки перегонов) — a_1 и a_2;

— узлы графа обозначены как $c_1^1, \ldots, c_6^1$, они соответствуют центрам обыкновенных стрелок. Каждому узлу инцидентны два ребра соответственно двум положениям стрелки, они соединены знаком Δ (см. рис. 2). Ребро, обозначенное знаком +, соответствует прямому направлению стрелки. Это ребро связывает центр стрелки c_i^1 и вершину c_i^2 только тогда, когда i-я стрелка находится в положении s. Другое ребро соответствует боковому направлению стрелки, т.е. это ребро связывает центр стрелки c_i^1 и вершину c_i^3 только тогда, когда стрелки находятся в положении $\overline{s}$. В тексте положения i-й стрелки обозначены как $c_i(\overline{s})$ или $c_i(s)$;

— ребра, соединяющие два блок-участка или блок-участки со стрелками, соответствуют соединительным рельсовым участкам.

Из теории графов: любая последовательность вида

$$v_1, e_1, \quad v_2, e_2 \ldots, \quad e_k, v_{k+1},$$

где $v_1, \ldots, v_{k+1}$ – вершины графа, $e_1, \ldots, e_k$ – его ребра, причем $e_i = (v_i, v_{i+1})$, $i = 1, \ldots k$, и все ребра различны, называется цепью (путем).

В данном случае последовательность ребер графа станции будем называть рельсовым путем, в котором:

— все ребра различны;

— концевые ребра последовательности соответствуют блок-участкам;

— каждая стрелка в последовательности представлена не более одного раза;

— стрелки, входящие в последовательность, установлены в положения соответственно стрелочным ребрам последовательности.

Примеры рельсовых путей:

— внутристанционный рельсовый путь: $(b_5, c_3(\overline{s}), c_5(s), b_4, c_6(s), c_4(\overline{s}), b_6)$ (соединительные участки не показаны);

— рельсовый путь с участками приближения a_1, a_2:

$$(a_1, b_1, c_1(s), b_3, c_2(s), b_2, a_2).$$

Определение 1. Ориентированную часть последовательности рельсового пути, имеющую начальный и конечный блок-участки, будем называть маршрутом.

Маршрут с начальным блок-участком b_i и конечным b_j обозначим как $m[b_i, b_j]$.

Примеры маршрутов из рис. 2

— внутристанционный маршрут:

$$m\left[b_5, b_6\right] = \left[b_5, c_3\left(\overline{s}\right), c_5\left(s\right), b_4, c_6\left(s\right), c_4\left(\overline{s}\right), b_6\right],$$

где правая часть маршрута содержит последовательность блок-участков и стрелок рельсового пути;

— маршрут отправления поезда от приемоотправочного блок-участок b_4 на участок приближения a_2:

$$m[b_4, a_2] = [b_4, c_6\left(s\right), c_4(s), c_2(\overline{s}), b_2, a_2];$$

— маршрут приема поезда на блок-участок b_4 с блок-участка приближения a_1:

$$m\left[a_1, b_4\right] = \left[a_1, b_1, c_1\left(\overline{s}\right), c_3\left(s\right), c_5\left(s\right), b_4\right].$$

Утверждение 1. Из рельсового пути, содержащего n блок-участков можно построить $n(n-1)$ различных маршрутов.

3. Основные положения постановки задачи

Дадим несколько простых определений. Множество маршрутов станционного графа обозначим как M. Совместимые маршруты — маршруты, по которым поезда могут двигаться одновременно и независимо друг от друга. Очевидно, что маршруты, которые не имеют общих блок-участков и общих стрелок, являются совместимыми. Множество попарно совместимых маршрутов обозначим как N. В общем случае может быть v таких множеств $(N_1, N_2, \ldots, N_v)$. Предполагается, что движение поезда по маршруту должно быть возможным без остановки от начала до конца маршрута. Для этого на этапе сборки маршрута физические стрелки устанавливаются в состояния в соответствие с направлением стрелок маршрута, входные и выходные светофоры блок-участков и стрелок устанавливаются в состояния, обеспечивающие поезду «зеленый путь».

Действующий маршрут определим как маршрут, начиная с момента установки стрелок и светофоров (сборка маршрута), и включающий процесс движения поезда по маршруту. Обозначим множество действующих маршрутов на момент времени t как $D(t)$.

Безопасность группового движения поездов гарантирована только тогда, когда для множества действующих маршрутов в любой момент времени t выполняется

$$D(t) \subseteq N_i, \quad i \in \{1, 2, \ldots, v\}.$$

Маршруты $m[b_i, b_j] = [b_i, b_{i,1}, \ldots, b_{i,k}, b_j]$ и $m[b_i, b_j] = [b_i, b_{r,1}, \ldots, b_{r,l}, b_j]$ альтернативны, если они имеют одинаковые начало и конец, при этом

$$\{b_{i,1}, \ldots, b_{i,k}\} \neq \{b_{r,1}, \ldots, b_{r,l}\}.$$

Множество альтернативных маршрутов с общим началом b_i и общим концом b_j обозначим как $M(b_i, b_j)$.

Пусть заданы начало b_i и конец b_j будущего маршрута. Требуется найти в множестве $M(b_i, b_j)$ маршрут $m[b_i, b_j] = [b_i, b_{i,1}, \ldots, b_{i,k}, b_j]$, который был бы попарно совместим с действующими маршрутами на станции в данный момент времени $D(t)$. При положительном исходе переходим к его сборке, после чего считаем этот маршрут действующим, формируем $D(t+1)$:

$$D(t+1) = \{[m[b_i, b_j]\} \cup D(t).$$

Таким образом, этапы построения нового маршрута $m\,[b_i, b_j]$ состоят из:

— нахождения множества станционных маршрутов;

— нахождения множества альтернативных маршрутов $M(b_i, b_j)$;

— нахождения маршрута $m_r[b_i, b_j] \in M(b_i, b_j)$, попарно совместимого с действующими на данный момент времени маршрутами $(D(t))$.

4. Логическая модель маршрута

Для построения логической модели маршрута необходимо уточнить, с какого окончания (1-го или 2-го) блок-участка станции начинается и оканчивается маршрут. Например, в маршруте (рис. 2)

$$m[b_5, b_6] = [b_5, c_3\,(\overline{s})\,, c_5(s), b_4, c_6(s), c_4(\overline{s}), b_6],$$

прием поезда на участок b_6 происходит со стороны окончания 1. Поэтому в описании маршрута элемент b_6 заменим на b_6^1. Аналогично, начало маршрута (элемент b_5) заменим на b_5^2, что соответствует отправлению поезда через окончание 2 блок-участка b_5. Проходной блок-участок b_4 заменим на (b_4^1, b_4^2).

Аналогично поступаем со стрелками: $c_3(\overline{s})$ заменяем на $\left(c_3^3, c_3^1\right)$, $c_5\,(s)$ — на (c_5^1, c_5^2), $c_6(s)$ — на (c_6^2, c_6^1), $c_4(\overline{s})$ — на (c_4^3, c_4^1) .

В итоге имеем подробное описание маршрута:

$$m\,[b_5, b_6] = \left[b_5^2, (c_3^3, c_3^1), \left(c_5^1, c_5^2\right), (b_4^1.b_4^2), (c_6^2, c_6^1), (c_4^3, c_4^1), b_6^1\right].$$

Определение 2. Свойство маршрута, обеспечивающее поезду «зеленый путь» от начала до конца маршрута определим как проходимость маршрута.

Проходимость маршрута есть аналог проводимости релейной схемы.

Рассмотрим процесс проектирования проходимости маршрута. Окончаниям каждого блок-участка b_i сопоставляются логические переменные разрешения x_i^1, x_i^2, единичное значение которых означают разрешение на въезд поезда со стороны окончаний 1, 2 соответственно. Одновременно с этим окончаниям блок-участка b_i сопоставляются логические переменные разрешения y_i^1, y_i^2, единичное значение которых означают разрешение на выезд поезда со стороны окончаний 1, 2 соответственно. Единичное значение переменной разрешения означает, что соответствующий ей светофор устанавливается в

состояние «зеленый», инверсное значение переменных разрешения означает, что соответствующий ей светофор находится в состоянии «красный». При этом, имеют место естественные ограничения:

$$x_i^1 x_i^2 = 0, \quad y_i^1 y_i^2 = 0, \quad x_i^1 y_i^1 = 0, \quad x_i^2 y_i^2 = 0.$$

Каждый блок-участок маршрута может быть либо участком приема поезда (концом маршрута), либо участком отправления (началом маршрута), либо проходным участком. Логическую функцию разрешения приема поезда на блок-участок b_i через окончание 1 обозначим как $e(b_i^1)$ (the end): $e(b_i^1) = x_i^1 \overline{x}_i^2 \overline{y}_i^1 \overline{y}_i^2$, что соответствует разрешению приема поезда на блок-участок b_i со стороны окончания 1 ($x_i^1 = 1$) и при этом — запрет на въезд – выезд поезда через окончание 2 (так как $\overline{x}_i^2 \overline{y}_i^2 = 1$) и выезд поезда через окончание 1 $\left(\overline{y}_i^1 = 1\right)$. Это соответствует обязательной остановке поезда на этом блок-участке (конец маршрута). Аналогично, разрешение приема поезда на блок-участок b_i со стороны окончания 2 обозначим как $e(b_i^2)$.

Таким образом, логические функции разрешения приема с остановкой поезда на блок-участок b_i имеют вид:

$$e(b_i^1) = x_i^1 \overline{x}_i^2 \overline{y}_i^1 \overline{y}_i^2, \quad e(b_i^2) = x_i^2 \overline{x}_i^1 \overline{y}_i^1 \overline{y}_i^2. \tag{1}$$

Логическую функцию разрешения отправления поезда (начало маршрута) с блок-участка b_i через окончание 1 обозначим как $d(b_i^1)$, через окончание 2 — $d(b_i^2)$ (departure):

$$d(b_i^1) = y_i^1 \overline{x}_i^1 \overline{x}_i^2 \overline{y}_i^2, \quad d(b_i^2) = y_i^2 \overline{x}_i^1 \overline{x}_i^2 \overline{y}_i^1. \tag{2}$$

При этом имеет место естественное ограничение: только одна из переменных разрешения приема или отправления x_i^1, x_i^2, y_i^1, y_i^2 блок-участка b_i может принимать единичное значение. Логическую функции разрешения проезда через проходной блок-участок b_i (passing) обозначим как $p(b_i^1, b_i^2)$, если въезд происходит через окончание 1, и как $p(b_i^2, b_i^1)$, если въезд — через окончание 2:

$$p(b_i^1, b_i^2) = x_i^1 y_i^2 \overline{x}_i^2 \overline{y}_i^1, \quad p(b_i^2, b_i^1) = x_i^2 y_i^1 \overline{x}_i^1 \overline{y}_i^2. \tag{3}$$

Логические функции разрешения проезда через стрелку c_i определим следующим образом. Функцию разрешения проезда от центра стрелки c_i^1 «станционного» графа по прямому направлению (ребро (c_i^1, c_i^2)) обозначим как $q(c_i^1, c_i^2)$:

$$q(c_i^1, c_i^2) = s_i \, z_i^{12}, \tag{4}$$

где z_i^{12} — переменная разрешения движения поезда от центра стрелки по прямому направлению.

Функцию разрешения проезда от центра стрелки c_i^1 по боковому направлению (ребро (c_i^1, c_i^3)) обозначим как $q(c_i^1, c_i^3)$:

$$q(c_i^1, c_i^3) = \overline{s}_i z_i^{13}. \tag{5}$$

Значения переменных z_i^{12}, z_i^{13} соответствуют состояниям светофоров, расположенных в центре стрелки, при этом $z_i^{12} z_i^{13} = 0$.

Функцию разрешения проезда к центру стрелки c_i^1 по прямому направлению (ребро (c_i^2, c_i^1)) обозначим как $q(c_i^2, c_i^1)$:

$$q(c_i^2, c_i^1) = s_i z_i^{21}. \tag{6}$$

Функцию разрешения проезда к центру стрелки c_i^1 по боковому направлению (ребро (c_i^3, c_i^1)) обозначим как $q(c_i^3, c_i^1)$:

$$q(c_i^3, c_i^1) = \overline{s}_i z_i^{31}. \tag{7}$$

Функцию разрешения проезда к центру стрелки c_i^1 по прямому или боковому направлению обозначим как $q(c_i^1)$:

$$q(c_i^1) = q(c_i^2, c_i^1) \vee q(c_i^3, c_i^1), \tag{8}$$

или

$$q\left(c_i^1\right) = s_i z_i^{21} \vee \overline{s}_i z_i^{31}. \tag{9}$$

При этом во всех случаях имеет место ограничение: только одна из переменных разрешения z_i^{12}, z_i^{13}, z_i^{21}, z_i^{31} стрелки c_i может принимать единичное значение.

Определение 3. Логическое произведение функций разрешения элементов маршрута определим как логическую функцию проходимости маршрута.

Обозначим логическую функцию проходимости маршрута $m[b_i, b_j]$ как $f(b_i, b_j)$.

Пример. Логическая функция проходимости маршрута

$$m[b_5, b_4] = [b_5^2, (c_3^3, c_3^1), \left(c_5^1, c_5^2\right), b_4^1]$$

имеет вид:

$$f\left(b_5, b_4\right) = (d\left(b_5^2\right) q\left(c_3^3, c_3^1\right) q\left(c_5^1, c_5^2\right) e\left(b_4^1\right)),$$

где

$$d(b_5^2) = y_5^2 \overline{x}_5^1 \overline{x}_5^2 \overline{y}_5^1, \quad q(c_3^3, c_3^1) = \overline{s}_3 z_3^{31},$$
$$q(c_5^1, c_5^2) = s_5 z_5^{12}, \quad e(b_4^1) = x_4^1 \overline{x}_4^2 \overline{y}_4^1 \overline{y}_4^2.$$

После замены функций d, q, e их правыми выражениями логическая функция проходимости имеет вид:

$$f(b_5, b_4) = y_5^2 \overline{x}_5^1 \overline{x}_5^2 \overline{y}_5^1 \overline{s}_3 z_3^{31} s_5 z_5^{12} x_4^1 \overline{x}_4^2 \overline{y}_4^1 \overline{y}_4^2.$$

Значения переменных y, x, z соответствуют состоянию светофоров блок-участков и стрелок, значения переменных s соответствуют направлению стрелок. По функции проходимости маршрута система управления движением

поездов на этапе сборки маршрута выдает управляющие воздействия на установку светофоров и стрелок в соответствующие состояния. Таким образом, логическая модель маршрута представляется как функция его проходимости. Существует проблема взаимодействия собираемого маршрута с действующими в данный момент времени маршрутами. В первом приближении все множество маршрутов можно разделить на совместимые, несовместимые и враждебные маршруты.

Совместимые маршруты — маршруты, по которым поезда могут двигаться одновременно и независимо друг от друга. Это маршруты, которые не имеют общих участков, а также общих стрелок. Очевидно следующее утверждение.

Утверждение 2. Логическое произведение функций проходимости совместимых маршрутов не равно нулю.

Несовместимые маршруты — маршруты, которые не могут выполняться одновременно. Пара маршрутов, отличающиеся положением хотя бы одной стрелки, не могут выполняться одновременно, так как одновременное движение поездов по таким маршрутам физически невозможно. Поэтому очевидно следующее утверждение.

Утверждение 3. Логическое произведение функций проходимости несовместимых маршрутов равно нулю.

Враждебные маршруты — пара маршрутов, по которым одновременное движение поездов может привести к аварии (столкновению). Это может быть, когда имеет место либо встречное движение поездов, либо один догоняет другого. В обоих случаях они должны иметь хотя бы один общий блок-участок.

Утверждение 4. Логическое произведение функций проходимости враждебных маршрутов равно нулю.

Доказательство:

Случай 1. При встречном движении поездов с общим конечным блок-участком b_i имеет место (формула 1):

$$e(b_i^1)e(b_i^2) = 0.$$

Случай 2. При встречном движении поездов и общим проходным блок-участком b_i имеет место (формула 3):

$$p(b_i^1, b_i^2)p(b_i^2, b_i^1) = 0.$$

Случай 3. При попутном движении поездов, когда начальный блок-участок b_i первого поезда является проходным для второго поезда (догоняющего), имеет место:

$$d(b_i^1)p(b_i^2, b_i^1)) \vee (d\left(b_i^2\right)p(b_i^1, b_i^2) = 0.$$

Доказательство закончено.

Таким образом, показано, что произведение функций проходимости несовместимых и враждебных маршрутов равно нулю. Поэтому все множество

маршрутов в дальнейшем будем разделять только на совместимые и несовместимые маршруты.

Следовательно, анализ на совместимость любой пары маршрутов можно делать по соответствующим функциям проходимости. С другой стороны, как будет показано ниже, по логической модели железнодорожной станции сначала находятся логические модели маршрутов в виде функций проходимости, по которым затем определяются маршруты.

5. Множество маршрутов

Нахождение станционных, альтернативных и прочих видов маршрутов возможно непосредственно по модели станции в виде «станционного» графа. Однако процесс нахождения представляется достаточно сложным, так как может потребоваться анализ «станционного» графа при различных состояниях стрелок, число которых может достигать 2^k, где k — количество стрелок графа.

В данном исследовании «станционный» граф (рис. 2) рассматривается как логическая схема со многими выходами. Каждый блок-участок b_j «станционного» графа рассматривается как выходной элемент. При этом остальные блок-участки рассматриваются как входные элементы. Каждый блок-участок b_j имеет два входа — b_j^1, b_j^2, поэтому ему соответствуют две логические схемы. Одна схема имеет выходной элемент b_j^1, другая — b_j^2. Итого имеем $2n$ логических схем, где n — количество блок-участков станции. Каждая схема представляет собой подграф «станционного» графа, который оканчивается соответствующим выходным элементом. Для каждой такой схемы строим формулу логической функции. Всякую формулу можно рассматривать как представление логической функции, аргументами которой являются переменные данной формулы. По логической функции каждой схемы находится множество потенциально возможных маршрутов, оканчивающихся выходным элементом рассматриваемой схемы. Построенные по всем блок-участкам станции логические схемы содержат полные данные о количестве и структуре всех станционных маршрутов.

Для нахождения формулы логической функции по заданной логической схеме применим процедуру, известную в теории анализа логических схем [5], как процедуру “от выхода логической схемы — к ее входам”. Напомним процедуру получения функции, реализуемой логической схемой, на простом примере (рис. 3).

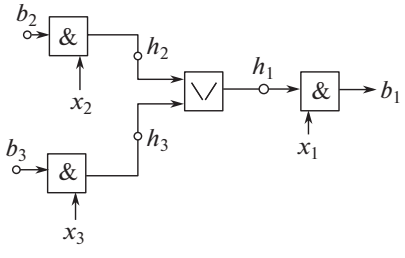


Рис. 3. Логическая схема из функциональных элементов.

Задать логическую схему — значить задать входные и выходные переменные, структуру связей между элементами схемы и логические функции, реализуемые этими элементами. В данном примере входные переменные x_1, x_2, x_3, b_2,b_3, выходная переменная — b_1.

Процедура получения логической функции по заданной схеме является многошаговой.

Шаг 1. Получение логической функции выходного элемента:

$$b_1 = x_1 h_1,$$

где h_1 — внутренняя переменная, она задает связь между выходным элементом схемы (элементом И) и связанного с ним элементом ИЛИ.

Шаг 2. Получение логической функции h_1:

$$h_1 = h_2 \vee h_3,$$

где h_2, h_3 — внутренние переменные, они задают связи между элементом ИЛИ и связанных с ним элементами И.

Шаг 3. Получение логических функций h_2, h_3:

$$h_2 = b_2 x_2, \quad h_3 = b_3 x_3.$$

Шаг 4. Получение логической функции схемы: последовательно заменяем внутренние переменные связи h_1, h_2, h_3 их правыми частями. В итоге имеем:

$$b_1 = x_1(b_2 x_2 \vee b_3 x_3) \quad \text{или} \quad b_1 = x_1 b_2 x_2 \vee x_1 b_3 x_3,$$

что можно рассматривать как два пути от входов схемы до выхода, если переменные x рассматривать как сигналы разрешения.

Рассмотрим процесс построения логических функций «станционного» графа (рис. 2), применяя основные положения вышеизложенной процедуры. При этом логическую функцию схемы, которая оканчивается элементом b_i^1 или b_i^2 блок-участка b_i будем называть логической функцией проходимости схемы с выходным элементом b_i^1 или b_i^2 и обозначать как $F(b_i^1)$ или $F(b_i^2)$ соответственно.

В качестве примера построим функцию проходимости схемы с выходным элементом b_4^1 блок-участка b_4 (рис. 2), т.е. построим функцию проходимости $F(b_4^1)$.

Шаг 1. Построение логической функции выходного элемента b_4^1.

Заменим выходной элемент b_4^1 на логическую функцию разрешения приема поезда (формула 1), т.е. на функцию

$$e(b_4^1) = x_4^1 \overline{x}_4^2 \overline{y}_4^1 \overline{y}_4^2.$$

В дальнейшем (для краткости) в логических формулах переменные x, y с отрицанием исключим, тогда:

$$e(b_4^1) = x_4^1 h_1,$$

где h_1 — переменная связи элемента b_4^1 и соседнего с ним ориентированного ребра (c_5^1, c_5^2) стрелки c_5.

Шаг 2. Построение функции h_1.

Ребро (c_5^1, c_5^2) стрелки c_5 соответствует прямому направлению от центра стрелки c_5^1. Заменим ребро (c_5^1, c_5^2) на функцию разрешения проезда от центра стрелки c_5^1 по прямому направлению (формула (4)): $q\left(c_5^1, c_5^2\right) = s_5 z_5^{12}$, тогда:

$$h_1 = s_5 z_5^{12} h_2,$$

где h_2 — переменная связи центра c_5^1 стрелки c_5 с центром стрелки c_3.

Шаг 3. Построение функции h_2.

Центр стрелки c_3 соединен с двумя ориентированными ребрами (c_3^2, c_3^1) и (c_3^3, c_3^1).

По формуле (8): $q(c_3^1) = q(c_3^2, c_3^1) \vee q(c_3^3, c_3^1)$, по формуле (9): $q\left(c_3^1\right) = s_3 z_3^{21} \vee \vee \overline{s}_3 z_3^{31}$. В результате имеем:

$$h_2 = s_3 z_3^{21} h_3 \vee \overline{s}_3 z_3^{31} h_4,$$

где h_3 — переменная связи ребра (c_3^2, c_3^1) стрелки c_3 с ребром (c_1^3, c_1^1) стрелки c_1, h_4 — переменная связи ребра (c_3^3, c_3^1) стрелки c_3 с элементом b_5^2 блок-участка b_5.

Шаг 4. Построение функции h_3.

По формуле (7): $q(c_1^3, c_1^1) = \overline{s}_1 z_1^{31}$, тогда:

$$h_3 = \overline{s}_1 z_1^{31} h_5,$$

где h_5 — переменная связи ребра (c_1^3, c_1^1) с элементом b_1^2 блок-участка b_1.

Шаг 5. Построение функции h_4.

Блок-участок b_5 является тупиковым участком станции, поэтому элемент b_5^2 является одним из входов рассматриваемой логической схемы, т.е. он может быть началом маршрута. В соответствие с формулой (2) имеем $d(b_5^2) = = y_5^2 \overline{x}_5^1 \overline{x}_5^2 \overline{y}_5^1$, тогда:

$$h_4 = y_5^2.$$

Шаг 6. Построение функции h_5.

Блок-участок b_1 является вход-выходным блок-участком. Он может быть или началом станционного маршрута (элемент b_1^2) или проходным участком (элемент (b_1^1, b_1^2)) при приеме поездов с участка перегона a_1. Тогда в соответствие с формулами (2) и (3) имеем $d(b_1^2) = y_1^2 \overline{x}_1^1 \overline{x}_1^2 \overline{y}_1^1$, $p(b_1^1, b_1^2) = x_1^1 y_1^2 \overline{x}_1^2 \overline{y}_1^1$, тогда:

$$h_5 = y_1^2 \vee x_1^1 h_6,$$

где h_6 — переменная связи элемента $(b_1^1,\ b_1^2)$ (блок-участок b_1) и участка перегона a_1.

Шаг 7. Построение функции h_6.

Участок перегона a_1 является одним из входов рассматриваемой логической схемы, т.е. он может быть началом маршрута. Тогда в соответствие с формулой (2) имеем

$$h_6 = d(a_1).$$

Шаг 8. Построение функции проходимости $F(b_4^1)$.

Заменим переменные связи $h_1 - h_6$ их правыми выражениями, получим формулу функции проходимости $F(b_4^1)$:

$$F(b_4^1) = x_4^1 s_5 z_5^{12}(s_3 z_3^{21}\overline{s}_1 z_1^{31}(y_1^2 \vee x_1^1 d\,(a_1)) \vee \overline{s}_3 z_3^{31} y_5^2). \tag{10}$$

Дизъюнктивная нормальная форма функции $F(b_4^1)$ имеет вид:

$$F(b_4^1) = x_4^1 s_5 z_5^{12} s_3 z_3^{21}\overline{s}_1 z_1^{31} y_1^2 \vee$$
$$\vee\, x_4^1 s_5 z_5^{12} s_3 z_3^{21}\overline{s}_1 z_1^{31} x_1^1 y_1^2 d(a_1) \vee x_4^1 s_5 z_5^{12}\overline{s}_3 z_3^{31} y_5^2.$$

Значения переменных x, y, z в формуле (10) соответствуют состоянию «зеленый» светофоров соответствующих блок-участков и стрелок, значения переменных типа s — направлению стрелок.

Каждой конъюнкции ДНФ функции $F(b_4^1)$ соответствует маршрут, оканчивающийся элементом b_4^1. Для построения маршрутов логические переменные конъюнкций заменяем (в обратном порядке) соответствующими элементами «станционного» графа:

— конъюнкции $\left(x_4^1 s_5 z_5^{12} s_3 z_3^{21}\overline{s}_1 z_1^{31} y_1^2\right)$ соответствует маршрут

$$m[b_1^2, b_4^1] = [b_1^2, c_1(\overline{s}), c_3(s), c_5(s), b_4^1];$$

— конъюнкции $(x_4^1 s_5 z_5^{12} s_3 z_3^{21}\overline{s}_1 z_1^{31} x_1^1 y_1^2 d(a_1))$ соответствует маршрут

$$m[a_1, b_4^1] = [a^1, (b_1^1, b_1^2), c_1(\overline{s}), c_3(s), c_5(s), b_4^1];$$

— конъюнкции $(x_4^1 s_5 z_5^{12}\overline{s}_3 z_3^{31} y_5^2)$ соответствует маршрут

$$m[b_5^2, b_4^1] = [b_5^2, c_3(\overline{s}), c_5(s), b_4^1].$$

Такой процедурой находятся все логические функции проходимости модели станции и соответствующие множества маршрутов станции:

$$F(b_1^1), F(b_1^2), F(b_2^1), F(b_2^2), \ldots, F(b_n^1), F(b_n^2),$$
$$M(b_1^1), M(b_1^2), M(b_2^1), M(b_2^2), \ldots, M(b_n^1), M(b_n^2).$$

Некоторые свойства функций проходимости:

1) все конъюнкции ДНФ функции $F(b_j^i)$ попарно ортогональны;

2) каждая конъюнкция ДНФ функции $F(b_j^i)$ является логической моделью соответствующего маршрута;

3) количество конъюнкций в ДНФ функции $F\left(b_j^i\right)$ равно количеству маршрутов, оканчивающихся элементом b_j^i;

4) каждая конъюнкция ДНФ функции $F(b_j^i)$ содержит полную информацию о положении стрелок и состояниях светофоров соответствующего маршрута;

5) количество конъюнкций по всем ДНФ функций $F(b_j^i)$, $i = 1$, $j = 1, \ldots, n$, равно количеству потенциально возможных станционных маршрутов.

О вычислительной сложности процедуры получения функций проходимости.

Можно доказать, что количество конъюнкций функции $F(b_j^i)$, и, следовательно, количество маршрутов, оканчивающихся элементом b_j^i, меньше или равно $2k_j^i$. Здесь k_j^i — количество стрелок в подграфе «станционного» графа, который оканчивается выходным элементом b_j^i, $k_j^i \leq k$ (k — количество стрелок станции). При этом предполагается, что два и более последовательно соединенных блок-участков в рельсовых путей «станционного» графа считаются как один. Количество шагов процедуры — меньше или равно $4k_j^i$.

6. Нахождение альтернативных маршрутов

Для получения множества маршрутов с общим началом b_i^r и общим концом b_j^v достаточно построить по вышерассмотренной процедуре функцию проходимости $F(b_j^v)$ и соответствующее множество маршрутов $M(b_j^v)$.

Множество маршрутов с общим началом b_i^r и общим концом b_j^v обозначим как множество $M(b_i^r, b_j^v)$: $M(b_i^r, b_j^v) \subseteq M(b_j^v)$.

Пример. Построим множество альтернативных маршрутов с общим началом b_2^1 и общим концом b_1^2 по рис. 2.

Шаг 1. Построим функцию проходимости $F(b_1^2)$:

$$F(b_1^2) = f_1(b_2^1, b_3^1, b_2^1) \vee f_2(b_2^1, b_4^1, b_2^1) \vee f_3(b_2^1, b_7^1, b_2^1) \vee$$
$$\vee f_4(b_3^1, b_1^2) \vee f_5(b_4^1, b_1^2) \vee f_6(b_6^1, b_1^2) \vee f_7(b_7^1, b_1^2).$$

Шаг 2. Функциям проходимости f_1, f_2, f_3 соответствуют альтернативные маршруты

$$m_1[b_2^1, b_3, b_1^2] = [b_2^1, c_2(s), (b_3^2, b_3^1), c_1(s), b_1^2],$$
$$m_2[b_2^1, b_4, b_1^2] = [b_2^1, c_2(\overline{s}), c_4(s), c_6(s), (b_4^2, b_4^1), c_5(s), c_3(s), c_1(\overline{s}), b_1^2],$$
$$m_3[b_2^1, b7, b_1^2] = [b_2^1, c_2(\overline{s}), c_4(s), c_6(\overline{s}), (b_7^2, b_7^1), c_5(\overline{s}), c_3(s), c_1(\overline{s}), b_1^2].$$

7. Жизненный цикл маршрута

Построим модель управления движением поезда по маршруту в виде автоматного графа переходов (рис. 4).

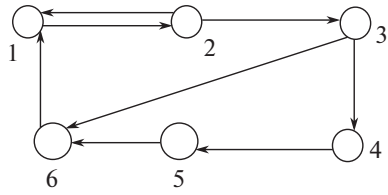


Рис. 4. Типовой жизненный цикл маршрута.

Состояние 1.

Задана модель станции в виде «станционного» графа, а также начало b_i^r и конец b_j^v проектируемого перемещения поезда. Множество действующих маршрутов на момент времени t задано в виде множества функций проходимости $D(t)$: $D(t) = \{d_1, \ldots, d_p\}$.

Состояние 2.

Требуется найти во множестве альтернативных маршрутов такой маршрут, который был бы совместим с действующими маршрутами.

Шаг 1. Построение по логической модели станции функции проходимости $F(b_j^v)$.

Шаг 2. Нахождение альтернативных маршрутов.

Из функции $F(b_j^v)$ находим m функций проходимости альтернативных маршрутов:

$$A(b_j^v) = \{f_1(b_i^r, b_j^v), \ldots, f_m(b_i^r, b_j^v)\}.$$

Шаг 3. Нахождение маршрута $m = [b_i^r, b_j^v]$, совместимого с действующими маршрутами.

Проверяем на ортогональность каждой функции $f_i \in A(b_j^v)$ с каждой функцией множества $D(t)$, либо вычисляем значение логического произведения функций $f_a d_1 d_2 \ldots d_p$, где $a \in \{1, \ldots, m\}$.

Если логическое произведение функций $f_a d_1 d_2 \ldots d_p \neq 0$, то маршрут $m_a(b_i^r, b_j^v)$ совместим с действующими маршрутами.

Переход из состояния 2 в состояние 1 происходит в случае отсутствия маршрута, совместимого с действующими маршрутами, иначе — переход в состояние 3.

Состояние 3.

Сборка маршрута $m_a\left(b_i^r, b_j^v\right)$, совместимого с действующими маршрутами.

По функции проходимости f_a маршрута $m_a(b_i^r, b_j^v)$ система управления движением поездов выдает управляющие воздействия:

— набор управляющих воздействий X, Y на установку входных-выходных светофоров блок-участков маршрута в соответствующие состояния («зеленый» либо «красный»). При этом состояние выходного светофора (состояние разрешения движения поезда) начального блок-участка b_i^r должно сохранятся «красным» $(y(b_i^r) = 0)$;

— набор Z управляющих воздействий на установку входных светофоров стрелок маршрута в соответствующие состояния;

— набор S управляющих воздействий на установку стрелок маршрута в соответствующие направления (прямое либо боковое).

После установки светофоров и стрелок в соответствующие состояния $(X^\circ, Y^\circ, Z^\circ, S^\circ)$ проверяются равенства: $X^\circ = X,\ Y^\circ = Y,\ Z^\circ = Z,\ S^\circ = S$. В случае хотя бы одного неравенства (состояние, хотя б одного светофора или стрелки, не соответствует заданному) происходит переход в состояние 6 графа переходов. Иначе маршрут $m_a(b_i^r, b_j^v)$ считается собранным и помещается в множество действующих маршрутов: $D(t+1) = D(t) \cup \{f_a\}$.

Переход в состояние 4.

Состояние 4.

Система управления или станционный диспетчер разрешают движения поезду: $y(b_i^r) = 1$.

Переход в состояние 5.

Состояние 5.

Движение поезда по маршруту.

Каждый блок-участок и каждая стрелка имеют автоблокировку, т.е. при пересечении поезда границы i-го блок-участка (или стрелки) его входные светофоры устанавливаются в состояние «красный».

Поезд находится на конечном блок-участке b_j^v, если вследствие автоблокировки $x(b_j^v) = 0$, то осуществляется переход в состояние 6.

Состояние 6.

Разборка маршрута.

Система управления движением поездов выдает управляющие воздействия на установку входных-выходных светофоров блок-участков и стрелок маршрута $m_a(b_i^r, b_j^v)$ в состояние «красный».

Из множества $D(t+1)$ действующих маршрутов исключается маршрут $m_a(b_i^r, b_j^v)$:

$$D(t+2) = D(t+1) \backslash \{f_a\}.$$

Переход в состояние 1.

8. Заключение

1. Показано использование методов математической логики при проектировании системы управления групповым движением поездов железнодорожной станции.

2. Разработан типовой жизненный цикл маршрута (проектирование маршрута, сборка, контроль движения поезда по маршруту, разборка маршрута), гарантирующий безопасное групповое движение поездов.

3. На этапе проектирования маршрута разработаны модель маршрута в виде логической функции проходимости, модель железнодорожной станции в виде логической схемы. На основе этих моделей определяются состояния

стрелочных переводов, светофоров маршрута, а также отношения между маршрутами (совместимые, несовместимые).

4. Разработана процедура получения множества потенциально возможных станционных маршрутов, множества альтернативных маршрутов.

СПИСОК ЛИТЕРАТУРЫ

1. *Васильев С.Н.* От классических задач регулирования к интеллектуальному управлению // Изв. АН. Теория и системы управления. 2001. № 1. С. 5–22. № 2. С. 5–21.
2. *Амбарцумян А.А., Потехин А.И.* Событийное логическое управление производственными процессами поточного типа. М.: ИПУ РАН, 2006.
3. *Кузнецов С.К., Потехин А.И.* Современные системы поддержки принятия решений железнодорожным диспетчером // Проблемы управления. 2017. № 1. С. 1–14.
4. *Потехин А.И.* Логические модели объектов железнодорожной стации // Проблемы управления. 2016. № 5. С. 71–79.
5. *Закревский А.Д., Поттосин Ю.В., Черемисинова Л.Д.* Логические основы проектирования дискретных устройств. М.: Изд-во физмат. лит-ры, 2007. 590 с.

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

Поступила в редакцию 17.09.2018

После доработки 15.10.2019

Принята к публикации 28.11.2019

СОДЕРЖАНИЕ

Лазарев А.А. Модели и методы теории расписаний. Танаев В.С. 3
Куприянов Б.В. Оценка и оптимизация производительности рекурсивного конвейера 6
Долгий А.Б., Левин Г.М., Розин Б.М. Структурно-параметрическая оптимизация комплекса пересекающихся множеств операций в условиях нестационарного спроса 26
Ковалёв М.Я., Розин Б.М., Гущинский Н.Н. Математическая модель и алгоритм случайного поиска для задачи оптимального планирования замены традиционного общественного транспорта электрическим 41
Сотсков Ю.Н. Область оптимальности перестановки обслуживания на одном приборе требований с неопределенными длительностями 60
Зиндер Я., Лазарев А.А., Мусатова Е.Г. Корректировка расписания движения на частично заблокированном сегменте железной дороги с разъездом 91
Малах С.А., Сервах В.В. Максимизация удельной приведенной прибыли в системах управления запасами 106
Гафаров Е.Р., Лазарев А.А., Вернер Ф. Минимизация суммарного взвешенного запаздывания на одном приборе с равными продолжительностями обслуживания требований 119
Лущакова И.Н. Геометрические алгоритмы определения точки в пересечении шаров 139
Потехин А.И. Логические основы управления групповым движением поездов . . 156

CONTENTS

Lazarev A.A. Models and methods of scheduling theory. Tanaev V.S. 3
Kupriyanov B.V. Estimation and Optimization of the Efficiency of a Recursive Conveyor 6
Dolgui A.B., Levin G.M., Rozin B.M. Structural-Parametric Optimization of a Complex of Intersecting Sets of Operations under Nonstationary Demand 26
Kovalyov M.Y., Rozin B.M., Guschinsky N.N. Mathematical Model and Random Search Algorithm for the Optimal Planning Problem of Replacing Traditional Public Transport with Electric 41
Sotskov Yu.N. Optimality Region for Job Permutation in Single-Machine Scheduling with Uncertain Processing Times 60
Zinder Y., Lazarev A.A., Musatova E.G. Rescheduling Traffic on a Partially Blocked Segment of Railway with a Siding 91
Malakh S.A., Servakh V.V. Maximization of Unit Present Profit in Inventory Management Systems 106

Gafarov E.R., Lazarev A.A., Werner F. Minimizing Total Weighted Tardiness for Scheduling Equal-Length Jobs on a Single Machine 119
Lushchakova I.N. Geometric Algorithms for Finding a Point in the Intersection of Balls 139
Potekhin A.I. Logical Foundations of Group Traffic Control of Trains 156