Российская Академия Наук

Автоматика и Телемеханика

Журнал основан в 1936 году

Выходит 12 раз в год

11

ноябрь

Москва 2021

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 «Объединённая редакция»

Тематический выпуск[1]

© 2021 г. С.П. АРСЕЕВ (9413serg@gmail.com),
Л.М. МЕСТЕЦКИЙ, д-р техн. наук (mestlm@mail.ru)
(Московский государственный университет им. М.В. Ломоносова)

СКЕЛЕТ СИМВОЛА КАК МОДЕЛЬ СЛЕДА ПЕРА ДЛЯ РАСПОЗНАВАНИЯ ПО ВОССТАНОВЛЕННОЙ ТРАЕКТОРИИ[2]

Статья посвящена исследованию задачи распознавания рукописного текста и сведению задачи распознавания по изображению текста к задаче распознавания по следу пера при его написании. Предложен метод, основанный на использовании медиального представления, восстанавливающий след пера по изображению рукописного текста. Предложено обоснование метода на основе экспериментального исследования распознавания символов по их изображению, по восстановленному и по истинному следу пера.

Ключевые слова: глубокое обучение, сверточные нейронные сети, рекуррентные нейронные сети, скелет, распознавание символов, распознавание рукописного текста.

DOI: 10.31857/S0005231021110015

1. Введение

Одной из активно исследуемых задач компьютерного зрения является распознавание рукописного текста. Эта задача находит множество практических применений, от автоматической обработки документов до работы с архивами. Методы, использующиеся для решения этой задачи, можно разделить на два подхода: offline-распознавание и online-распознавание.

Offline-распознавание рукописного текста представляет собой распознавание по изображению. На практике эта задача встречается чаще всего, и способы ее решения достаточно разнообразны: от классических алгоритмов компьютерного зрения с выделением векторов признаков и последующей классификацией до использования искусственных нейронных сетей со сверточной архитектурой, популярность которых начала быстро расти после того, как в 2012 г. сеть AlexNet [1] превзошла конкурирующие методы. В данный момент нейронные сети превосходят альтернативные методы во многих задачах компьютерного зрения. Классические методы распознавания изображений отличаются разнообразием применяемых дескрипторов: здесь могут

[1] Статьи с 3 стр. по 147 стр. являются окончанием тематического выпуска № 10, 2021.

[2] Работа выполнена при финансовой поддержке Российского фонда фундаментальных исследований (проект № 20-01-00664).

использоваться дескрипторы LBP [2], SIFT [3], SURF [4], HOG [5] и многие другие. Пример подборки классических методов, использующихся для задачи распознавания рукописного текста, можно увидеть в публикации [6], где на примере распознавания индийского письма деванагари исследуются различные классификаторы, оперирующие на гистограмме ориентированных градиентов изображения. Одно из первых применений сверточных нейронных сетей для задачи распознавания символов описывается в [7].

Online-распознавание представляет собой распознавание следа пера в момент написания текста. Так как такой след имеет вид последовательности координат пера, рекуррентные нейронные сети часто применяются в решении этой задачи, пример использования таких сетей для решения задачи распознавания китайских рукописных символов можно увидеть в [8]. Такая задача зачастую проще в решении, чем задача offline-распознавания из-за меньшей размерности данных, но основным ограничением этого подхода является необходимость наличия данных о траектории, что требует использования оборудования для записи данных и существенно ограничивает возможность применять этот подход на практике. Во многих задачах запись следа пера может отсутствовать, тогда для распознавания доступны только изображения текста.

Цель данной статьи заключается в попытке свести задачу распознавания рукописных символов по их изображению (offline-распознавания) к задаче распознавания по траектории пера (online-распознавания) посредством построения по изображению символа возможной траектории пера при его начертании. Такое сведение позволит применять методы online-распознавания ко многим задачам распознавания текста, что позволит пользоваться преимуществами этой группы методов: они не накладывают таких требований на объем обучающей коллекции и могут оказываться более эффективными. Для реконструкции следа пера используется представление символа в виде графовой модели, что позволяет рассматривать также возможность решения задачи с использованием графовых нейронных сетей, которые успешно применяются для детектирования [9] и распознавания различных объектов. Однако первые эксперименты [10] с применением простейших графовых нейронных сетей для распознавания рукописных символов показали отсутствие заметного преимущества таких моделей по сравнению со сверточными нейронными сетями, обрабатывающими изображение.

2. Восстановление следа пера

Основная идея предложенного подхода заключается в том, что для символа строится его модель в формате геометрического графа. Вершины графа близки к положениям пера в определенные моменты при начертании символа, а ребра между вершинами соответствуют отрезкам траектории пера. Полная траектория пера будет представлять собой набор маршрутов в этом графе, где каждый маршрут соответствует куску траектории, написанному без отрыва пера.

2.1. Построение медиального представления

В качестве основы для графа, моделирующего положения пера, используется медиальное представление (скелет) символа на изображении, впервые описанное в [11]. Введем несколько ключевых понятий, также использовавшихся в [10, 12].

Определение 1. Скелетом (медиальным представлением) называется множество центров всех вписанных кругов фигуры [13, раздел 2.4].

В медиальном представлении фигуры сложной формы обычно присутствует множество коротких "шумовых" ветвей, которые могут быть исключены без существенных потерь информации, необходимой для распознавания, а их наличие, наоборот, затрудняет обработку.

Определение 2. Стрижкой скелета или его регуляризацией называется процесс исключения из скелета ветвей, вклад которых в образование формы объекта незначителен [13, 14, глава 10].

В предложенном алгоритме применятся метод стрижки, подробно описанный в [13, раздел 10.3]. Медиальное представление является множеством точек, соединенных отрезками прямых линий и парабол [13, раздел 7.1], но применительно к данной задаче его можно приблизить неориентированным геометрическим графом $S = (V; E)$. Потери при приближении кусков парабол прямолинейными ребрами незначительны из-за малой длины ребер графа. Построение модели производится по алгоритму, подробно описанному в [13]. После бинаризации изображения связные области на его бинарном представлении аппроксимируются многоугольниками [13, главы 5–6], после чего на этих многоугольниках строится морфологическая модель [13, главы 7–10]. Пример выхода алгоритма после регуляризации показан на рис. 1.

Предлагаемый подход основан на предположении, что скелет изображения символа может использоваться в качестве модели следа пера при написании

Рис. 1. Медиальное представление символа (буква "g").

этого символа. В большинстве случаев вершины скелетного графа расположены в точках, близких к середине штриха, и существенные отклонения от этого возможны только в случае сильного шума на краях штриха, не дающего точно построить модель, или же существенного перекрытия близко расположенных штрихов, вызывающего визуальное слияние нескольких штрихов в один. Поэтому задача восстановления траектории пера (одной из возможных) сводится к построению маршрута (последовательности инцидентных вершин и ребер с возможными повторами) или множества в скелетном графе, в которые в совокупности должны входить все ребра графа. Предложенный алгоритм строит маршрут для каждой связной компоненты графа без разбиения на пересекающиеся штрихи. Таким образом, восстанавливается возможная траектория начертания "без отрыва пера". В силу архитектуры рекуррентной нейронной сети, используемой в дальнейшем для распознавания, данные о ребрах не подаются на вход этой сети. Поэтому вместо полного маршрута достаточно сохранять только соответствующую ему последовательность вершин, что и будет являться выводом полного алгоритма.

В общей сложности предложенный в статье алгоритм, сводящий задачу offline-распознавания к задаче online-распознавания, состоит из следующих этапов:

1. Построение медиального представления символа в формате геометрического графа;

2. Конструирование вспомогательного графа (далее — метаграфа) по исходному графу;

3. Обход метаграфа;

4. Построение обхода скелетного графа на основе обхода его метаграфа;

5. Генерализация построенного маршрута посредством удаления близко расположенных вершин и ребер.

2.2. Построение вспомогательного графа

Ребра исходного скелетного графа моделируют фрагменты траектории пера, и обход этих фрагментов определяется направлением движения пера на данном участке. Для последовательности смежных вершин (расположенных на штрихе) степени 2 (не развилок и не терминальных) направление обхода определяется только выбором направления движения пера по всей последовательности. Неоднозначность построения маршрута возникает на развилках (т.е. в вершинах степени 3), а отдельным источником неоднозначности является проблема обхода цикла, так как при обходе требуется обойти все ребра цикла, при этом минимизировав количество повторов. Для облегчения задачи такого обхода графа на его основе строится новый вспомогательный граф.

Определение 3. Метаграфом исходного графа называется вспомогательный граф, где каждому циклу из минимального базиса циклов исходного графа ставится в соответствие единственная вершина. Метаграф строится по алгоритму 1.

Определение 4. Вершина V_1 исходного графа соответствует вершине V_2 его метаграфа, если выполняется одно из следующих условий:

1. *V_1 входит в один цикл исходного графа, которому при построении вершин метаграфа ставится в соответствие V_2. В этом случае одной вершине исходного графа может соответствовать несколько вершин метаграфа и, наоборот, каждой вершине метаграфа, поставленной в соответствие какому-либо циклу, соответствуют все вершины, входящие в этот цикл;*
2. *V_1 не входит ни в один цикл исходного графа, а V_2 ставится ей в соответствие при построении вершин метаграфа. В этом случае соответствие имеет взаимно однозначный характер: каждой такой вершине соответствует ровно одна вершина на другом графе.*

Алгоритм 1 (построение метаграфа).

I. Выделение в исходном графе минимального базиса циклов, т.е. множества циклов, являющегося базисом пространства циклов для графа (любой цикл графа можно представить как комбинацию циклов из этого множества), имеющего при этом минимальную суммарную длину.
II. Построение вершин:
    1. Если вершина не входит ни в один из циклов, она переходит в метаграф напрямую, т.е. в метаграф добавляется одна новая вершина, которая ставится в соответствие этой исходной вершине;
    2. Для каждого цикла из базиса в метаграф добавляется ровно одна новая вершина, которая ставится в соответствие этому циклу.
III. Построение ребер:
    1. Если обе вершины ребра исходного графа не входят ни в один из циклов, это ребро переходит в метаграф напрямую, т.е. между соответствующими им вершинами проводится ребро той же длины;
    2. Если одна вершина ребра не входит ни в один из циклов, а вторая входит в один или несколько циклов, то в метаграф добавляются ребра той же длины, соединяющие вершину, соответствующую первой вершине, с каждой из вершин, соответствующих этим циклам;
    3. Если обе вершины ребра входят в разные циклы графа (но само ребро не является частью цикла), то в метаграф добавляется ребро той же длины, которое соединяет две вершины метаграфа, соответствующие этим циклам;
    4. Если два цикла исходного графа имеют хотя бы одну общую вершину, то в метаграф добавляется ребро нулевой длины, которое соединяет вершины, соответствующие этим циклам;
    5. Берется минимальное остовное дерево получившегося графа.

Построение метаграфа по этому алгоритму однозначно, если однозначен выбор минимального базиса циклов исходного графа. Это всегда выполняется для скелета плоской фигуры, где длина ребра определяется как евклидово расстояние между его концами, и вероятность наличия циклов с

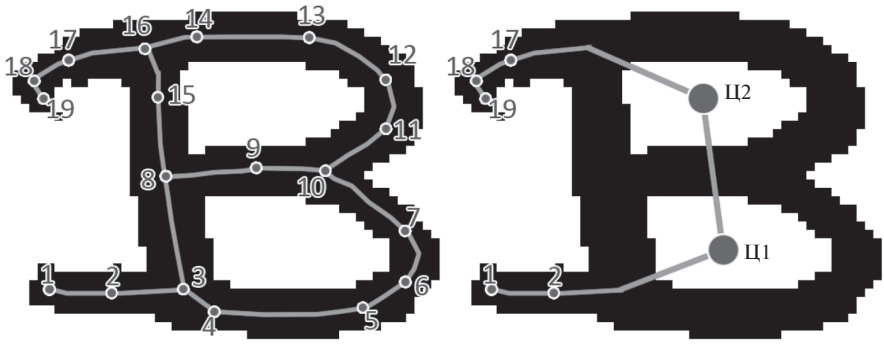

Рис. 2. Пример скелетного графа и его метаграфа.

равной длиной мала. Пример построения этого вспомогательного графа показан на рис. 2, где для символа из тестового набора показан его скелетный граф (большинство вершин не показано в целях удобства восприятия) и соответствующий ему метаграф. В этом примере каждый цикл исходного графа превращается в вершину (Ц1 и Ц2) метаграфа, причем вершина Ц1 метаграфа соответствует вершинам 3–10 исходного графа, а вершина Ц2 — вершинам 8–16. Так как циклы могут иметь общие вершины, получающийся на предпоследнем шаге алгоритма 1 граф не обязательно будет являться деревом. При взятии его остовного дерева в качестве метаграфа ни одно ребро, соответствующее ребру исходного графа, не будет потеряно, так как циклы в метаграфе возникают только в результате применения правил III, п. 2 и III, п. 4 из алгоритма построения метаграфа.

2.3. Обход метаграфа

Так как целью восстановления траектории является ее пригодность для распознавания, соответствие реальной траектории не требуется, но желательно. В качестве эвристического критерия для качественной траектории можно предложить минимизацию повторных проходов через ребра графа. Для графа в виде дерева это достигается посредством рекурсивного обхода поддеревьев от наименее глубокого до наиболее глубокого. Этот метод может быть применен для обхода метаграфа, на основе которого уже строится маршрут в исходном графе.

Первый обход метаграфа и его разметка производятся по алгоритму 2.

Алгоритм 2 (обход метаграфа).

I. Выбор начальной вершины:

1. В исходном графе строится список всех вершин степени 1 (конец штриха);
2. Если список пуст (вершин степени 1 в графе нет), то рассматриваются все вершины графа;
3. Для каждой вершины из списка считается ее сумма координат $S = x + y$;

4. Выбирается вершина N_0 из списка с минимальной суммой S;
5. Если на шаге 2 рассматривались все вершины графа, то в исходный граф добавляется новая вспомогательная вершина N с теми же значениями координат, что и N_0, соединенная ребром длины 0 с вершиной N_0. В метаграф добавляется соответствующая ей вершина с теми же координатами, аналогично соединенная с вершиной, соответствующей N_0. В противном случае $N = N_0$;
6. Вершина N называется стартовой.

II. Обход метаграфа:

1. Метаграф рассматривается как дерево с корнем в вершине N, которая выбирается в качестве текущей вершины;
2. Повторять следующие шаги:
   - Если у текущей вершины есть дочерние вершины, не помеченные как обойденные, спуститься в одну из них, выбранную произвольно (назначить ее текущей вершиной);
   - Иначе, если текущая вершина является терминальной, присвоить ей значение глубины $d = 1$, пометить ее как обойденную и подняться в родительскую вершину;
   - Иначе (если у текущей вершины есть дочерние вершины, но все помечены как обойденные) присвоить ей значение глубины $d = \max(d_{child}) + 1$, где d_{child} – множество меток глубины ее дочерних вершин; пометить ее как обойденную. Если текущая вершина совпадает с вершиной N (обойден весь граф), завершить алгоритм, иначе подняться в родительскую вершину.

Разберем приведенный алгоритм на примере графа с рис. 2. Обход начинается с вершины 17. Метаграф данного графа имеет линейную структуру, что существенно упрощает обход. В качестве стартовой выбирается вершина 19. Алгоритм последовательно спускается в вершину 1 и присваивает ей глубину 1, после чего возвращается, помечая глубину остальных вершин (вершина 2 будет иметь значение 2, вершина Ц1 – значение 3 и т.д.).

2.4. Обход графа

После первого обхода метаграфа производится второй обход, во время которого в соответствии с алгоритмом 3 строится трасса.

Определение 5. Трассой в графе называется последовательность вершин графа, соответствующая последовательности вершин в некотором маршруте, в который входят все ребра и вершины данного графа. Трасса, которая строится данным алгоритмом, соответствует возможной последовательности положений пера при написании символа.

Алгоритм 3 (обход графа).

1. Если текущая вершина не соответствует никакому из циклов исходного графа, то занести ее в трассу и спуститься в еще не обойденную дочернюю вершину с минимальной глубиной.

2. Если текущая вершина соответствует какому-либо циклу исходного графа, то выбрать финальную вершину, где будет производиться выход из цикла. Эта вершина соответствует дочерней вершине с максимальной глубиной. Также выбрать направление обхода цикла. Двигаться по соответствующему циклу исходного графа, занося в трассу пройденные вершины, от точки входа (из предыдущей вершины) до ближайшей точки выхода, соответствующей еще не обойденной вершине метаграфа и не совпадающей с финальной вершиной выхода. Если не обойдена только финальная вершина, то двигаться по циклу, занося пройденные вершины в трассу, до этой вершины.

3. Если дочерних вершин нет (вершина является листом дерева) или все они уже обойдены, то занести текущую вершину в трассу и вернуться на уровень вверх. Для вершины, соответствующей циклу исходного графа, это означает движение по циклу до точки изначального входа в цикл (как и ранее, с занесением в трассу всех пройденных вершин).

Ни одно ребро исходного графа не будет потеряно при таком обходе, так как обход покрывает все вершины и ребра метаграфа. Каждому ребру исходного графа, не входящему ни в какой из циклов, соответствует ребро метаграфа (при выделении остовного дерева на последнем этапе построения метаграфа из него исключаются только дубликаты ребер и те ребра, которые не имеют аналогов на исходном графе). Ребра же, которые входят в циклы исходного графа, обходятся при рассмотрении вершины, соответствующей этому циклу в метаграфе. Такой обход позволяет по метаграфу построить трассу в исходном графе.

Помимо минимизации возвратов и повторных проходов, сортировка дочерних вершин на глубине соответствует специфике задачи: так как короткие ответвления будут обходиться в первую очередь, метод применим не только для построения обхода в отдельных символах, но и для более длинных элементов рукописного текста (слитно написанных слов).

На рис. 2 обход исходного графа начинается с вершины 19 и продолжается до прихода в вершину 16, которая соответствует вершине Ц2 метаграфа. При выборе обхода против часовой стрелки алгоритм выбирает точку выхода в вершине 8, после чего последовательно обходит вершины цикла 15, 8, 9, 10, 11 и далее до возврата в вершину 16. Далее производится переход в вершину 8 через вершину 15 и обход цикла Ц1: точка выхода находится в вершине 3, и алгоритм обходит цикл в следующем порядке: 3, 4, . . . , 7, 10, 9, 8, 3, после чего обход цикла завершается и обходятся вершины 2 и 1. Итоговая трасса будет иметь вид [19, 18, 17, 16, 15, 8, 9, 10, 11, 12, 13, 14, 16, 15, 8, 3, 4, 5, 6, 7, 10, 9, 8, 3, 2, 1].

2.5. Генерализация трассы

В местах большой кривизны штрихов текста его скелетный граф будет иметь большое количество вершин, расположенных близко одна к другой. В трассе, получающейся при обходе такого скелета, также будет много близко расположенных вершин, идущих подряд. Объединение таких вершин (или исключение из трассы лишних вершин) позволяет существенно уменьшить

объем входных данных без существенных потерь важной информации, описывающей форму символа. Это облегчает обучение классификатора при решении задачи распознавания по траектории. Прореживание трассы производится в соответствии с алгоритмом 4.

Алгоритм 4 (прореживание трассы).

I. Вершины начинают рассматриваться со второй вершины трассы (так как трасса строится таким образом, что она всегда начинается в терминальной вершине).
II. Для вершины проверяются следующие критерии:
    1. Вершина имеет степень 2;
    2. Предыдущая или следующая вершина последовательности тоже имеет степень 2;
    3. Расстояние от рассматриваемой вершины до обеих вершин, рассмотренных на предыдущем шаге, не превышает порогового значения.
III. В случае выполнения всех трех критериев вершина удаляется из последовательности.
IV. Рассмотрение переходит к следующей вершине.

В результате всех вышеописанных действий из скелетного графа получается трасса, последовательность его вершин. Эта последовательность объявляется реконструированным следом пера (одним из возможных) при написании рассматриваемого рукописного символа или слова.

3. Распознавание символов

Пригодность восстановленной траектории пера для распознавания рукописного символа по нему обосновывается посредством экспериментального исследования работы алгоритма распознавания на этой восстановленной траектории и сравнения с результатами распознавания по реальной траектории. Сведение задачи offline-распознавания к задаче online-распознавания обосновывается также посредством экспериментального исследования и сравнением качества распознавания символов при помощи предложенных алгоритмов и при помощи традиционных методов offline-распознавания.

Для распознавания символов на основе восстановленной траектории применялся алгоритм, описанный в [15] и основанный на архитектуре двунаправленной рекуррентной нейронной сети, которая состояла из 24 управляемых рекуррентных блоков [16]. Рекуррентная нейронная сеть представляет собой вариант искусственной нейронной сети, предназначенной для работы с данными, имеющими формат последовательности. Структурный элемент такой сети, называемый рекуррентным блоком (управляемый рекуррентный блок представляет собой один из типов таких блоков), выводит два значения: вывод рекуррентного блока и скрытое состояние, а на вход принимает также два значения: текущий элемент последовательности и скрытое состояние с обработки предыдущего элемента последовательности. Размерность каждого блока определяет размерность скрытых состояний.

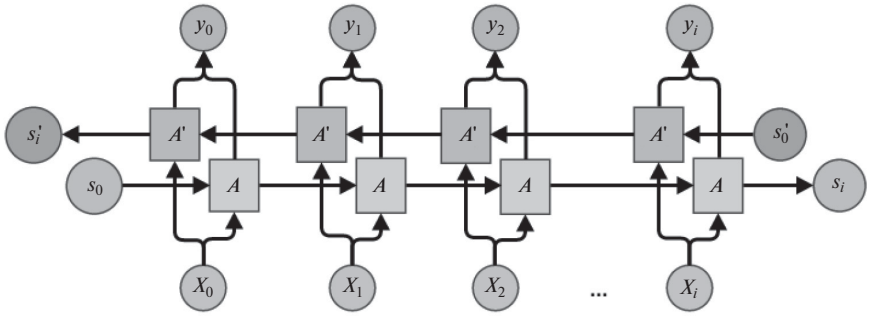

Рис. 3. Двунаправленная рекуррентная нейронная сеть.

Двунаправленная рекуррентная нейронная сеть представляет собой две схожие рекуррентные нейронные сети, которые обрабатывают последовательность в двух противоположных направлениях. Выход этих сетей объединяется для получения итогового результата. Схема такой сети показана на рис. 3, на котором A обозначает рекуррентный блок, x – элементы последовательности, S – скрытые состояния, y – вывод для каждого элемента, i – длина последовательности. Итоговый вывод нейронной сети для всей последовательности получается из y_0 и y_i.

Предобучение модели в данном случае не проводилось; обучающая и тестовая выборки имели соотношение размеров 4 к 1. Обучающая выборка также была расширена посредством случайных сдвигов точек исходных последовательностей.

Для сравнения с алгоритмом offline-распознавания использовалась сверточная нейронная сеть VGG16 [17], предобученная на базе ImageNet. Разбиение на обучающую и тестовую выборки проводилось также в соотношении 4 к 1.

4. Экспериментальное исследование

4.1. Малый объем данных

Первый эксперимент проводился на коллекции данных, представленной в [15]. Эта коллекция данных представляет собой траектории пера, записанные с помощью сенсорного контроллера. Набор данных состоит из семи классов и содержит по 100 примеров для каждого класса.

Для получения изображений символов для offline-распознавания эти траектории были растеризованы, дальше предложенный алгоритм работал с этими изображениями без использования данных о реальной траектории пера. Примеры растеризованных изображений показаны на рис. 4.

Результаты экспериментального исследования

Метод	Точность
RNN-True	0,93
RNN-Offline	0,87
VGG	0,60

Рис. 4. Примеры символов, класс “А”.

Результаты экспериментального исследования приведены в таблице. Методам присвоены следующие обозначения:
RNN-True — рекуррентная нейронная сеть, которая была обучена на истинной траектории пера;
RNN-Offline — рекуррентная нейронная сеть, которая была обучена на восстановленной предложенным алгоритмом траектории;
VGG — сверточная сеть VGG16, предобученная на ImageNet и дообученная на изображениях из набора.

Видно, что предложенный метод отстает от online-распознавания по истинным траекториям, что достаточно предсказуемо, но при этом его точность заметно превосходит точность метода offline-распознавания. Сверточная сеть VGG16 оказалась неспособна обучиться на такой выборке, в отличие от рекуррентной модели.

4.2. Большой объем данных

Второй эксперимент проводился на коллекции данных EMNIST [18], представляющем собой преобразованный набор данных NIST Special Database 19 [19]: по сравнению с оригинальным набором данных размер изображений уменьшен, а сами изображения из бинаризованных переведены в оттенки серого. Следует отметить, что предложенный алгоритм, основанный на построении морфологического скелета и восстановлении траектории, устойчив к вышеописанным преобразованиям, так как один из этапов алгоритма включает в себя (повторную) бинаризацию изображения, координаты узлов скелетного графа являются дробными числами, а алгоритм обхода инвариантен к сдвигу и масштабированию координат узлов графа.

Основными трудностями для обучения сверточных нейронных сетей на наборе EMNIST является высокое число классов (10 цифр, 26 прописных букв латинского алфавита и 26 строчных, итого 62 класса), повышенная схожесть отдельных классов между собой (в частности, для некоторых букв строчные и прописные написания практически идентичны друг другу), а также несбалансированность классов в наборе. В связи с этим многие исследователи упрощают постановку задачи, объединяя часть классов между собой. В [20] набор данных был сокращен до 47 классов, итоговый размер обучающей выборки составил 112 800 изображений, а тестовой— 18 800 изображений. На получившемся наборе искусственная сверточная сеть показала точность классификации, равную 81%.

Предложенный алгоритм проверялся на всех классах набора без их объединения и балансирования. Несмотря на простоту предложенной рекуррентной модели, она показала точность классификации, равную 74%, что сопоставимо с точностью сверточной модели, решавшей более простую задачу. Бóльшая часть ошибок алгоритма связана именно с межклассовым сходством, те символы, которые не имеют аналогов со сходным написанием, распознаются с высокой точностью.

5. Заключение

Предложен метод сведения задачи offline-распознавания рукописных символов к задаче online-распознавания, проиллюстрированный посредством использования рекуррентной искусственой нейронной сети для распознавания на двух коллекциях данных. При малом количестве обучающих примеров модель показала способность к обучению на данных, получаемых предложенным алгоритмом, в отличие от сверточной модели, существенно отставшей по точности.

На большом наборе данных алгоритм показал способность к обучению и классификации несмотря на сложность набора данных, в связи с которой большинство работ, использующих этот набор, оперируют упрощенными данными с объединенными классами. С учетом повышенной сложности задачи, показанная алгоритмом точность сопоставима с методами, основанными на сверточных аналогах.

Дальнейшие направления развития этого исследования могут включать в себя эксперименты с совершенствованием способов восстановления траектории пера, а также эксперименты на различных данных, например слитных последовательностях символов.

СПИСОК ЛИТЕРАТУРЫ

1. *Krizhevsky A., Sutskever I., Hinton G.E.* Imagenet Classification with Deep cCnvolutional Neural Networks // Advances in neural information processing systems. 2012. P. 1097–1105.
2. *Ojala T., Pietikainen M., Hardwood D.* A Comparative Study of Texture Measures with Classification Based on Feature Distributions // Pattern Recognition. 1996. V. 29. No. 1. P. 51–59.
3. *Lowe D.G.* Distinctive image features from scale-invariant keypoints // Inte. J. of Computer Vision. 2004. V. 16. No. 2. P. 91–110.
4. *Bay H., Ess A., Tuytelaars T., Van Gool L.* SURF: Speeded up Robust Features // Computer Vision and Image Understanding. 2008. V. 110. No. 3. P. 346–359.
5. *Dalal N., Triggs B.* Histograms of Oriented Gradients for Human Detection // IEEE Computer Society Conf. on Computer Vision and Pattern Recognition (CVPR 2005). 2005. V. 1. P. 886–893.
6. *Pal U., Wakabayashi T., Kimura F.* Comparative Study of Devnagari Handwritten Character Recognition Using Different Feature and Classifiers // 2009 10th Int. Conf. on Document Analysis and Recognition. 2009. P. 1111–1115.

7. *Ciresan D.C., Meie, U., Gambardella L.M., Schmidhuber J.* Convolutional Neural Network Committees for Handwritten Character Classification // 2011 Int. Conf. on Document Analysis and Recognition. 2011. P. 1115–1139.
8. *Zhang X.Y., Yin F., Zhang Y.M., Liu C.L., Bengio Y.* Drawing and Recognizing Chinese Characters with Recurrent Neural Network // IEEE Trans. Pattern Analysis and Machine Intelligence. 2017. V. 40. No. 4. P. 849–862.
9. *Захаров А.А., Баринов А.Е., Жизняков А.Л., Титов В.С.* Поиск объектов на изображениях с использованием структурного дескриптора на основе графов // Компьютерная оптика. 2018. Т. 42. № 2. С. 283–290.
10. *Ломов Н.А., Арсеев С.П.* Нейронные сети для распознавания формы по медиальному представлению // Тр. Междунар. конф. по компьютерной графике и зрению "Графикон". ФГУ "Федеральный исследовательский центр Институт прикладной математики им. М.В. Келдыша РАН". 2018. № 28. С. 218–221.
11. *Blum H.A.* A Transformation for Extracting New Descriptors of Shape // Proc. Sympos. Models for the Perception of Speech and Visual Form. 1967. V. 4. P. 362–380.
12. *Арсеев С.П., Местецкий Л.М.* Распознавание рукописного текста по восстановленному следу пера с помощью медиального представления // Информационные технологии и нанотехнологии (ИТНТ-2020). 2020. С. 683–689.
13. *Местецкий Л.М.* Непрерывная морфология бинарных изображений: фигуры, скелеты, циркуляры. М.: Физматлит, 2009.
14. *Shaked D., Bruckstein A.M.* Pruning Medial Axes // Computer Vision and Image Understanding. 1998. V. 69. No. 2. P. 156–189.
15. *Fischer T.* "Online Handwritten Character Recognition with capacitive sensors" project // Online: https://github.com/tobiasfshr/online-handwritten-character-recognition-capacitive-sensors. 2018.
16. *Chung J., Gulcehre C., Cho KH., Bengio Y.* Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling // arXiv preprint. 2014. arXiv:1412.3555.
17. *Simonyan K., Zisserman A.* Very Deep Convolutional Networks for Large-scale Image Recognition // arXiv preprint. 2014. arXiv:1409.1556.
18. *Cohen G., Afshar S., Tapson J., Van Schaik A.* EMNIST: Extending MNIST to Handwritten Letters // 2017 Int. Joint Conf. on Neural Networks (IJCNN). 2017. P. 2921–2926.
19. *Grother P., Hanaoka K.* NIST Special Database 19 Hhandprinted Forms and Characters 2nd Edition // National Institute of Standards and Technology, Tech. Rep. Online: http://www.nist.gov/srd/upload/nistsd19.pdf. 2016.
20. *Neftci E.O., Augustine C., Paul S., Detorakis G.* Event-driven Random Backpropagation: Enabling Neuromorphic Deep Learning Machines // Frontiers in Neuroscience. 2017. V. 11. P. 324.

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

Поступила в редакцию 20.01.2021

После доработки 15.06.2021

Принята к публикации 30.06.2021

© 2021 г. А.В. ГРАБОВОЙ (grabovoy.av@phystech.edu)
(Московский физико-технический институт),
В.В. СТРИЖОВ, д-р физ.-мат. наук (strijov@phystech.edu)
(Вычислительный центр им. А.А. Дородницына ФИЦ ИУ РАН, Москва)

БАЙЕСОВСКАЯ ДИСТИЛЛЯЦИЯ МОДЕЛЕЙ ГЛУБОКОГО ОБУЧЕНИЯ[1]

Исследуется проблема понижения сложности аппроксимирующих моделей. Рассматриваются методы, основанные на дистилляции моделей глубокого обучения. Вводятся понятия учителя и ученика. Предполагается, что модель ученика имеет меньшее число параметров, чем модель учителя. Предлагается байесовский подход к выбору модели ученика. Предложен метод назначения априорного распределения параметров ученика на основе апостериорного распределения параметров модели учителя. Так как пространства параметров учителя и ученика не совпадают, предлагается механизм приведения пространства параметров модели учителя к пространству параметров модели ученика путем изменения структуры модели учителя. Проводится теоретический анализ предложенного механизма приведения. Вычислительный эксперимент проводился на синтетических и реальных данных. В качестве реальных данных рассматривается выборка FashionMNIST.

Ключевые слова: выбор модели, байесовский вывод, дистилляция модели, локальные преобразования, преобразования вероятностных пространств.

DOI: 10.31857/S0005231021110027

1. Введение

Исследуется проблема снижения числа обучаемых параметров моделей машинного обучения. Примерами моделей с избыточным числом параметров являются AlexNet [1], VGGNet [2], ResNet [3], BERT [4, 5], mT5 [6], GPT3 [7] и др. В табл. 1 приводится число параметров моделей глубокого обучения, которое с годами растет. Это влечет снижение интерпретируемости моделей. Данная проблема рассматривается в специальном классе задач по состязательным атакам (adversarial attack) [8]. Большое число параметров требует значительных вычислительных ресурсов. Из-за этого данные модели не

[1] Настоящая статья содержит результаты проекта Математические методы интеллектуального анализа больших данных, выполняемого в рамках реализации Программы Центра компетенций Национальной технологической инициативы "Центр хранения и анализа больших данных", поддерживаемого Министерством науки и высшего образования Российской Федерации по Договору МГУ им. М.В. Ломоносова с Фондом поддержки проектов Национальной технологической инициативы от 11.12.2018 №13/1251/2018. Работа выполнена при поддержке Российского фонда фундаментальных исследований (проекты № 19-07-01155, № 19-07-00875).

Таблица 1. Число параметров в моделях машинного обучения

Название	AlexNet	VGGNet	ResNet	BERT	mT5	GPT3
Год	2012	2014	2015	2018	2020	2020
Тип данных	изображение	изображение	изображение	текст	текст	текст
Число параметров, млрд	0,06	0,13	0,06	0,34	13	175

могут быть использованы в мобильных устройствах. Для снижения числа параметров предложен метод дистилляции модели [9–11]. Дистиллируемая модель с большим числом параметров называется *учителем*, а модель, получаемая путем дистилляции, называется *учеником*. При оптимизации параметров модели ученика используется модель учителя с фиксированными параметрами.

Определение 1. Дистилляция модели — снижение сложности модели путем выбора модели в множестве более простых моделей на основе параметров и ответов более сложной фиксированной модели.

Идея дистилляции предложена Дж.Е. Хинтоном и В.Н. Вапником [9–11]. В их публикациях предлагалось использовать ответы учителя в качестве целевой переменной для обучения модели ученика. Поставлен ряд экспериментов, в которых проводилась дистилляция моделей для задачи классификации машинного обучения. Базовый эксперимент на выборке MNIST [12] показал результативную дистилляцию избыточно сложной нейросетевой модели в нейросетевую модель меньшей сложности. Проводился эксперимент по дистилляции ансамбля моделей в одну модель для решения задачи распознания речи. В [9] проведен эксперимент по обучению экспертных моделей на основе одной модели с большим числом параметров при помощи предложенного метода дистилляции на ответах учителя.

В [13] предложен метод передачи селективности нейронов (neuron selectivity transfer), основанный на минимизации специальной функции потерь. Метод основан на вычислении функции максимального среднего отклонения (maximum mean discrepancy) между выходами всех слоев модели учителя и ученика. Вычислительный эксперимент показал эффективность данного метода для задачи классификации изображений на примере выборок CIFAR [14] и ImageNet [15].

В данной статье предлагаются методы, основанные на байесовском выводе. В качестве априорного распределения параметров модели ученика предлагается использовать апостериорное распределение параметров модели учителя. Решается задача приведения пространства параметров модели учителя к пространству параметров модели ученика. Авторы предлагают подход, основанный на последовательном приведении пространства параметров модели учителя.

Определение 2. Структура модели — множество структурных параметров модели, которые задают вид суперпозиции.

Определение 3. Приведение параметрических моделей — изменение структуры модели (одной или нескольких моделей), в результате которого векторы параметров различных моделей лежат в одном пространстве.

В результате приведения параметры модели учителя и модели ученика лежат в одном пространстве. Как следствие, в качестве априорного распределения параметров модели ученика выбирается апостериорное распределение параметров модели учителя. В данной статье в качестве параметрических моделей рассматривается полносвязная нейронная сеть. В качестве структурных параметров модели выбраны число слоев, а также размер каждого скрытого слоя.

В рамках предложенного метода приведения параметрических моделей не оговорен выбор порядка на множестве параметров модели учителя. Для этого предлагается упорядочивать параметры модели учителя на основе их значимости [16]. Первый нейрон является наиболее значимым, а последний нейрон — наименее значимым. Порядок задается на основе отношения плотности распределения упорядочиваемого параметра к плотности распределения параметра в нуле [17] или на основе метода Белсли [18]. В рамках данной статьи порядок на параметрах задается случайный образом.

В рамках вычислительного эксперимента проводится теоретический анализ. Предложенный метод дистилляции анализируется на примере синтетической выборки, а также на реальной выборке FashionMnist [19].

2. Постановка задачи дистилляции

Задана выборка

$$\mathfrak{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{m}, \qquad \mathbf{x}_i \in \mathbb{R}^n, \quad y_i \in \mathbb{Y}, \tag{1}$$

где $\mathbf{x}_i, y_i$ — признаковое описание и целевая переменная i-го объекта, число объектов в обучающей выборке обозначается m. Матрица признаковых описаний обозначается $\mathbf{X} = \left[\mathbf{x}_1^\mathsf{T}, \ldots, \mathbf{x}_m^\mathsf{T}\right]^\mathsf{T}$, а вектор целевых переменных обозначается $\mathbf{y} = [y_1, \ldots, y_m]$. Размер признакового описания объектов обозначается n. Множество $\mathbb{Y} = \{1, \ldots, K\}$ для задачи классификации, где K число классов, множество $\mathbb{Y} = \mathbb{R}$ для задачи регрессии.

Задана модель учителя в виде суперпозиций линейных и нелинейных преобразований:

$$f\big(\mathbf{x}\big) = \boldsymbol{\sigma} \circ \mathbf{U}_T \boldsymbol{\sigma} \circ \mathbf{U}_{T-1} \circ \ldots \mathbf{U}_2 \boldsymbol{\sigma} \circ \mathbf{U}_1 \mathbf{x},$$

где T — число слоев модели учителя, $\boldsymbol{\sigma}$ — функция активации, а $\mathbf{U}_t$ обозначает матрицу линейного преобразования. Матрицы $\mathbf{U}$ соединяются в вектор параметров $\mathbf{u}$ модели учителя f:

$$\mathbf{u} = \mathrm{vec}\big(\mathbf{U}_T, \mathbf{U}_{T-1}, \ldots, \mathbf{U}_1\big), \tag{2}$$

где vec — операция векторизации соединенных матриц. Каждая матрица $\mathbf{U}_t$ имеет размер $n_t \times n_{t-1}$, где $n_0 = n$, а $n_T = 1$ для задачи регрессии и $n_T = K$

для задачи классификации на K классов. Число параметров N_{tr} учителя f

$$N_{\text{tr}} = \sum_{t=1}^{T} n_t n_{t-1}. \tag{3}$$

Для построения вектора параметров $\mathbf{u}$ задается полный порядок на элементах матриц $\mathbf{U}_t$. Для полносвязанной нейронной сети вводится естественный порядок, индуцированный номером слоя t, номером нейрона и номером элемента вектора параметров нейрона: выбирается матрица $\mathbf{U}_t$, строка этой матрицы и элемент строки.

Например, для модели учителя в задаче регрессии:

$$f\big(\mathbf{x}\big) = \boldsymbol{\sigma} \circ \mathbf{U}_3 \boldsymbol{\sigma} \circ \mathbf{U}_2 \boldsymbol{\sigma} \circ \mathbf{U}_1 \mathbf{x} \tag{4}$$

вектор параметров $\mathbf{u}$ принимает вид

$$\mathbf{u} = \Big[u_1^{1,1}, \ldots, u_1^{1,n}, \ldots, u_1^{n_1,1}, \ldots, u_1^{n_1,n}, u_2^{1,1}, \ldots, u_2^{1,n_1}, \ldots \\ \ldots, u_2^{n_2,1}, \ldots, u_2^{n_2,n_1}, u_3^{1,1}, \ldots, u_3^{1,n_2}\Big].$$

Пусть для вектора параметров $\mathbf{u}$ учителя f известно апостериорное распределение параметров $p\big(\mathbf{u}|\mathfrak{D}\big)$.

На основе выборки $\mathfrak{D}$ и апостериорного распределения параметров учителя f требуется выбрать модель ученика из параметрического семейства функций:

$$g\big(\mathbf{x}\big) = \boldsymbol{\sigma} \circ \mathbf{W}_L \boldsymbol{\sigma} \circ \ldots \circ \mathbf{W}_1 \mathbf{x}, \quad \mathbf{W}_l \in \mathbb{R}^{n_l \times n_{l-1}},$$

где L — число слоев модели ученика. Число параметров N_{st} модели ученика g вычисляется аналогично выражению (3). Вектор параметров модели ученика $\mathbf{w}$ строится аналогичным образом (2). Модель g задается своим вектором параметров $\mathbf{w}$. Следовательно, задача выбора модели g эквивалентна задаче оптимизации вектора параметров $\mathbf{w} \in \mathbb{R}^{N_{\text{st}}}$.

Параметры $\hat{\mathbf{w}} \in \mathbb{R}^{N_{\text{st}}}$ оптимизируются при помощи вариационного вывода на основе совместного правдоподобия модели и данных:

$$\mathcal{L}\big(\mathfrak{D}, \mathbf{A}\big) = \log p\big(\mathfrak{D}|\mathbf{A}\big) = \log \int\limits_{\mathbf{w} \in \mathbb{R}^{N_{\text{st}}}} p\big(\mathfrak{D}|\mathbf{w}\big) p\big(\mathbf{w}|\mathbf{A}\big) d\mathbf{w}, \tag{5}$$

где $p\big(\mathbf{w}|\mathbf{A}\big)$ — априорное распределение вектора параметров модели ученика, $\mathbf{A}$ обозначает гиперпараметры априорного распределения. Взятие интеграла (5) является вычислительно сложной задачей. В качестве приближенного решения используется вариационный подход [17, 18]. Для этого задается вариационное распределение параметров модели ученика $q\big(\mathbf{w}|\boldsymbol{\mu}, \boldsymbol{\Psi}\big)$, которое аппроксимирует неизвестное апостериорное распределение $p\big(\mathbf{w}|\mathfrak{D}\big)$

$$q\big(\mathbf{w}|\boldsymbol{\mu}, \boldsymbol{\Psi}\big) \approx p\big(\mathbf{w}|\mathfrak{D}\big),$$

где оптимальные гиперпараметры распределения $\hat{\boldsymbol{\mu}}$ и $\hat{\boldsymbol{\Psi}}$ требуется найти вместе с оптимальными параметрами $\hat{\mathbf{w}}$, решив оптимизационную задачу:

$$\hat{\mathbf{w}}, \hat{\boldsymbol{\mu}}, \hat{\boldsymbol{\Psi}} = \arg \min_{\boldsymbol{\mu}, \boldsymbol{\Psi}, \mathbf{w}} \mathrm{D}_{\mathrm{KL}}\big(q(\mathbf{w}|\boldsymbol{\mu}, \boldsymbol{\Psi}) || p(\mathbf{w}|\mathbf{A})\big) - \sum_{i=1}^{m} \log p\big(y_i|\mathbf{x}_i, \mathbf{w}\big), \tag{6}$$

где D_{KL} обозначает расстояние Кульбака–Лейблера между вариационным распределением $q\big(\mathbf{w}|\boldsymbol{\mu}, \boldsymbol{\Psi}\big)$ и априорным распределением $p\big(\mathbf{w}|\mathbf{A}\big)$. Второе слагаемое формулы (6) является логарифмом правдоподобия $\log p\big(y_i|\mathbf{x}_i, \mathbf{w}\big)$ объекта $(\mathbf{x}_i, y_i) \in \mathfrak{D}$ выборки (1). Выражение (6) не учитывает параметры учителя f. Для использования информации о распределении параметров учителя предлагается рассмотреть параметры априорного распределения $p\big(\mathbf{w}|\mathbf{A}\big)$ как функцию от апостериорного распределения учителя $p\big(\mathbf{u}|\mathfrak{D}\big)$.

3. Построение априорного распределения ученика

Апостериорное распределение параметров модели учителя предполагается нормальным:

$$p\big(\mathbf{u}|\mathfrak{D}\big) = \mathcal{N}\big(\mathbf{m}, \boldsymbol{\Psi}\big), \tag{7}$$

где $\mathbf{m}$ и $\boldsymbol{\Psi}$ — гиперпараметры этого распределения. На основе гиперпараметров $\mathbf{m}$ и $\boldsymbol{\Psi}$ требуется задать параметры $\mathbf{A}$ априорного распределения $p\big(\mathbf{w}|\mathbf{A}\big)$. Когда структура моделей учителя и ученика задается числом слоев и размером этих слоев, возможны следующие варианты: 1) число слоев и размер каждого слоя совпадают; 2) число слоев совпадает, а размеры слоев различаются; 3) не совпадает число слоев.

3.1. Учитель и ученик имеют одну структуру

Рассмотрим следующие условия:

1) число слоев модели ученика равняется числу слоев модели учителя $L = T$;

2) размеры соответствующих слоев совпадают, другими словами, для всех t, l, таких что $t = l$, выполняется $n_l = n_t$, где n_t обозначает размер t-го слоя учителя, а n_l — размер l-го слоя ученика.

В случае выполнения этих условий априорное распределение параметров модели ученика приравнивается к апостериорному распределению параметров учителя $p\big(\mathbf{w}|\mathbf{A}\big) = p\big(\mathbf{u}|\mathfrak{D}\big)$.

3.2. Удаление нейрона в слое учителя

Приведем структуру модели учителя к структуре модели ученика согласно определению 3 при помощи последовательных преобразований вектора параметров $\mathbf{u}$. Рассмотрим преобразование

$$\phi\big(t, \mathbf{u}\big) : \mathbb{R}^{N_{\mathrm{tr}}} \to \mathbb{R}^{N_{\mathrm{tr}} - 2n_t}$$

вектора $\mathbf{u}$, которое описывает удаление одного нейрона из t-го слоя учителя. Обозначим новый вектор параметров $\boldsymbol{\upsilon} = \phi\big(t, \mathbf{u}\big)$, а элементы вектора, которые были удалены, — через $\bar{\boldsymbol{\upsilon}}$. Заметим, что векторы $\boldsymbol{\upsilon}$ и $\bar{\boldsymbol{\upsilon}}$ являются случайными величинами.

Теорема 1. Пусть выполняются следующие условия:

1) *апостериорное распределение* $p\big(\mathbf{u}|\mathfrak{D}\big)$ *параметров модели учителя является нормальным распределением* (7);

2) *число слоев модели учителя равняется числу слоев модели ученика* $T = L$;

3) *размеры соответствующих слоев не совпадают, другими словами, для всех* t, l, *таких что* $t = l$, *выполняется* $n_t \leqslant n_l$.

Тогда апостериорное распределение параметров модели учителя $p\big(\boldsymbol{\upsilon}|\mathfrak{D}\big)$ *также является нормальным.*

Доказательство. Не уменьшая общности, пусть $\phi\big(t, \mathbf{u}\big)$ удаляет j-й нейрон в t-м слое, что является удалением j-й строки матрицы $\mathbf{U}_t$. Заметим, что удаление j-й строки матрицы $\mathbf{U}_t$ влечет удаление j-й компоненты вектора z_{t+1}, где

$$\mathbf{z}_t = \boldsymbol{\sigma} \circ \mathbf{U}_{t-1}\boldsymbol{\sigma} \circ \ldots \mathbf{U}_2\boldsymbol{\sigma} \circ \mathbf{U}_1\mathbf{x}.$$

Удаление j-й компоненты вектора $\mathbf{z}_{t+1}$ эквивалентно занулению j-го столбца матрицы $\mathbf{U}_{t+1}$. Заметим, что тогда предсказание модели не зависит от параметров j-й строки матрицы $\mathbf{U}_t$, а поэтому данными параметрами также можно пренебречь.

Найдем распределение вектора $\boldsymbol{\upsilon}$. Для поиска распределения вектора параметров после зануления j-го столбца матрицы $\mathbf{U}_{t+1}$ воспользуемся формулой условной вероятности $p\big(\bar{\boldsymbol{\nu}}_1|\mathfrak{D}, \boldsymbol{\nu}_1 = \mathbf{0}\big)$, а для удаления j-й строки матрицы $\mathbf{U}_t$ воспользуемся маргинализацией распределения $p\big(\bar{\boldsymbol{\nu}}_1|\mathfrak{D}, \boldsymbol{\nu}_1 = \mathbf{0}\big)$. Обозначим зануляемые параметры модели через $\boldsymbol{\nu}_1$, а удаляемые параметры — через $\boldsymbol{\nu}_2$. Также обозначим все параметры, которые не были занулены, через $\bar{\boldsymbol{\nu}}_1 = [\boldsymbol{\upsilon}^\mathsf{T}, \boldsymbol{\nu}_2^\mathsf{T}]$. Итоговое распределение параметров принимает вид:

$$p\big(\boldsymbol{\upsilon}|\mathfrak{D}\big) = \int\limits_{\boldsymbol{\nu}_2} p\big(\bar{\boldsymbol{\nu}}_1|\mathfrak{D}, \boldsymbol{\nu}_1 = \mathbf{0}\big)d\boldsymbol{\nu}_2.$$

Из свойств нормального распределения следует, что распределение

$$p\big(\bar{\boldsymbol{\nu}}_1|\mathfrak{D}, \boldsymbol{\nu}_1 = \mathbf{0}\big) \tag{8}$$

также является нормальным распределением с параметрами $\boldsymbol{\mu}, \boldsymbol{\Xi}$:

$$\boldsymbol{\mu} = \mathbf{m}_{\bar{\nu}_1} + \boldsymbol{\Psi}_{\bar{\nu}_1,\nu_1}\boldsymbol{\Psi}^{-1}_{\nu_1,\nu_1}\left(\mathbf{0} - \mathbf{m}_{\nu_1}\right),$$
$$\boldsymbol{\Xi} = \boldsymbol{\Psi}_{\bar{\nu}_1,\bar{\nu}_1} - \boldsymbol{\Psi}_{\bar{\nu}_1,\nu_1}\boldsymbol{\Psi}^{-1}_{\nu_1,\nu_1}\boldsymbol{\Psi}_{\bar{\nu}_1,\nu_1},$$

где введенные обозначения $\mathbf{m}_{\bar{\nu}_1}, \mathbf{m}_{\nu_1}$ соответствуют подвектору вектора $\mathbf{m}$, который относится к параметрам $\bar{\boldsymbol{\nu}}_1$ и $\boldsymbol{\nu}_1$ соответственно. Ковариационная

матрица $\boldsymbol{\Psi}_{\bar{\nu}_1,\nu_1}$ обозначает подматрицу матрицы $\boldsymbol{\Psi}$, которая соответствует ковариационной матрицей между параметрами $\bar{\boldsymbol{\nu}}_1$ и $\boldsymbol{\nu}_1$.

Распределение $p\big(\boldsymbol{\upsilon}|\mathfrak{D}\big)$ найдем при помощи маргинализации распределения (8) по параметрам $\boldsymbol{\nu}_2$. Используя свойства нормального распределения, получаем распределение

$$p\big(\boldsymbol{\upsilon}|\mathfrak{D}\big) = \mathcal{N}\big(\boldsymbol{\mu}_{\boldsymbol{\upsilon}}, \boldsymbol{\Xi}_{\boldsymbol{\upsilon},\boldsymbol{\upsilon}}\big), \tag{9}$$

где $\boldsymbol{\mu}_{\boldsymbol{\upsilon}}$ обозначает подвектор вектора $\boldsymbol{\mu}$, который относится к вектору параметров $\boldsymbol{\upsilon}$, а матрица $\boldsymbol{\Xi}_{\boldsymbol{\upsilon},\boldsymbol{\upsilon}}$ является подматрицей матрицы $\boldsymbol{\Xi}$, которая относится к вектору параметров $\boldsymbol{\upsilon}$. Теорема 1 доказана.

Теорема 1 задает апостериорное распределение параметров (9) после зануления нейронов в модели нейросети — учителя. Заметим, что аналогичным образом можно удалить сразу подмножество нейронов в рамках одного слоя. В случае если число нейронов отличается в нескольких слоях модели нейросети учителя, то выполняются последовательно применения отображения $\phi\big(t, \mathbf{u}\big)$ для каждого t-го слоя.

3.3. Удаление слоя учителя

Приведем структуру модели учителя к модели ученика согласно определению 3 при помощи последовательных преобразований вектора параметров $\mathbf{u}$. Рассмотрим преобразование

$$\psi\big(t, \mathbf{u}\big) : \mathbb{R}^{N_{\text{tr}}} \to \mathbb{R}^{N_{\text{tr}} - n_t n_{t-1}}$$

вектора $\mathbf{u}$, которое описывает удаление одного t-го слоя. Обозначим новый вектор параметров $\boldsymbol{\upsilon} = \psi\big(t, \mathbf{u}\big)$, а элементы вектора, которые были удалены, — через $\bar{\boldsymbol{\upsilon}}$.

Теорема 2. Пусть выполняются следующие условия:

1) *апостериорное распределение параметров* $p\big(\mathbf{u}|\mathfrak{D}\big)$ *модели учителя является нормальным распределением* (7);

2) *соответствующие размеры слоев совпадают,* $n_t = n_{t-1}$, *т.е. матрица* $\mathbf{U}_t$ *является квадратной*;

3) *функция активации удовлетворяет свойству идемпотентности* $\boldsymbol{\sigma} \circ \boldsymbol{\sigma} = \boldsymbol{\sigma}$.

Тогда апостериорное распределение также описывается нормальным распределением с плотностью распределения

$$p\big(\boldsymbol{\upsilon}|\mathfrak{D}\big) = \mathcal{N}\big(\mathbf{m}_{\boldsymbol{\upsilon}} + \boldsymbol{\Psi}_{\boldsymbol{\upsilon},\bar{\boldsymbol{\upsilon}}}\boldsymbol{\Psi}^{-1}_{\bar{\boldsymbol{\upsilon}},\bar{\boldsymbol{\upsilon}}}\left(\mathbf{i} - \bar{\boldsymbol{\upsilon}}\right), \boldsymbol{\Psi}_{\boldsymbol{\upsilon},\boldsymbol{\upsilon}} - \boldsymbol{\Psi}_{\boldsymbol{\upsilon},\bar{\boldsymbol{\upsilon}}}\boldsymbol{\Psi}^{-1}_{\bar{\boldsymbol{\upsilon}},\bar{\boldsymbol{\upsilon}}}\boldsymbol{\Psi}_{\boldsymbol{\upsilon},\bar{\boldsymbol{\upsilon}}}\big), \tag{10}$$

где вектор $\mathbf{i}$ *задается как*

$$\mathbf{i} = [\underbrace{1, 0, \ldots, 0}_{n_t}, \underbrace{0, 1, \ldots, 0}_{n_t}, \underbrace{0, 0, 1, \ldots, 0}_{n_t}, \underbrace{0, \ldots, 1}_{n_t}]^{\mathsf{T}}.$$

Доказательство. Рассмотрим структуру нейронной сети с T слоями и $T+1$ слоем. Не уменьшая общности, для удаления рассматривается t-й слой, для которого выполняются условия этой теоремы. Заметим, что если t-й слой нейронной сети с $T+1$ слоем приравнять к единичной матрице, то он будет эквивалентным архитектуре с T слоями:

$$\begin{aligned} f &= \boldsymbol{\sigma} \circ \mathbf{U}_{T+1}\boldsymbol{\sigma} \circ \mathbf{U}_T \ldots \boldsymbol{\sigma} \circ \mathbf{U}_t\boldsymbol{\sigma} \circ \ldots \mathbf{U}_2\boldsymbol{\sigma} \circ \mathbf{U}_1 = \\ &= \boldsymbol{\sigma} \circ \mathbf{U}_{T+1}\boldsymbol{\sigma} \circ \mathbf{U}_T \ldots \boldsymbol{\sigma} \circ \mathbf{I}\boldsymbol{\sigma} \circ \ldots \mathbf{U}_2\boldsymbol{\sigma} \circ \mathbf{U}_1 = \\ &= \boldsymbol{\sigma} \circ \mathbf{U}_{T+1}\boldsymbol{\sigma} \circ \mathbf{U}_T \ldots \boldsymbol{\sigma} \circ \boldsymbol{\sigma} \circ \ldots \mathbf{U}_2\boldsymbol{\sigma} \circ \mathbf{U}_1 = \\ &= \boldsymbol{\sigma} \circ \mathbf{U}_{T+1}\boldsymbol{\sigma} \circ \mathbf{U}_T \ldots \boldsymbol{\sigma} \circ \ldots \mathbf{U}_2\boldsymbol{\sigma} \circ \mathbf{U}_1. \end{aligned}$$

Получаем, что удаление t-го слоя нейросети эквивалентно приравниванию матрицы параметров t-го слоя к единичной матрице. Распределение параметров после приравнивания к единичной матрице вычисляется при помощи условного распределения. В силу общих свойств нормального распределения условное распределение также является нормальным распределением с параметрами $\boldsymbol{\mu}, \boldsymbol{\Xi}$:

$$\begin{aligned} \boldsymbol{\mu} &= \mathbf{m}_{\boldsymbol{\upsilon}} + \boldsymbol{\Psi}_{\boldsymbol{\upsilon},\bar{\boldsymbol{\upsilon}}}\boldsymbol{\Psi}_{\bar{\boldsymbol{\upsilon}},\bar{\boldsymbol{\upsilon}}}^{-1}\left(\mathbf{i} - \bar{\boldsymbol{\upsilon}}\right), \\ \boldsymbol{\Xi} &= \boldsymbol{\Psi}_{\boldsymbol{\upsilon},\boldsymbol{\upsilon}} - \boldsymbol{\Psi}_{\boldsymbol{\upsilon},\bar{\boldsymbol{\upsilon}}}\boldsymbol{\Psi}_{\bar{\boldsymbol{\upsilon}},\bar{\boldsymbol{\upsilon}}}^{-1}\boldsymbol{\Psi}_{\boldsymbol{\upsilon},\bar{\boldsymbol{\upsilon}}}, \end{aligned}$$

где вектор $\mathbf{m}_{\boldsymbol{\upsilon}}$ является подвектором вектора $\mathbf{m}$ соответствующей параметрам $\boldsymbol{\upsilon}$, а матрица $\boldsymbol{\Psi}_{\boldsymbol{\upsilon},\bar{\boldsymbol{\upsilon}}}$ является подматрицей ковариационной матрицы $\boldsymbol{\Psi}$, соответствующей векторам параметров $\boldsymbol{\upsilon}$ и $\bar{\boldsymbol{\upsilon}}$. Теорема 2 доказана.

Теорема 2 задает апостериорное распределение (10) параметров после удаления слоя нейросети. Полученное распределение $p\big(\boldsymbol{\upsilon}|\mathfrak{D}\big)$ является оценкой апостериорного распределения модели без одного слоя.

3.4. Выполнение последовательных преобразований

Преобразования ϕ, ψ приводят пространство параметров учителя f к пространству параметров ученика g. После приведения параметрических моделей получаем, что параметры модели учителя и модели ученика принадлежат одному семейству 3.1.

4. Вычислительный эксперимент

Вычислительный эксперимент анализирует предложенный метод дистилляции на основе апостериорного распределения параметров модели учителя.

4.1. Синтетические данные

Проанализируем модель на синтетической выборке. Выборка построена следующим образом:

$$\begin{aligned} \mathbf{w} &= \big[w_j : w_j \sim \mathcal{N}\big(0,1\big)\big]_{n\times 1}, \quad \mathbf{X} = \big[x_{ij} : x_{ij} \sim \mathcal{N}\big(0,1\big)\big]_{m\times n}, \\ \mathbf{y} &= \Big[y_i : y_i \sim \mathcal{N}\big(\mathbf{x}_i^{\mathsf{T}}\mathbf{w}, \beta\big)\Big]_{m\times 1}, \end{aligned}$$

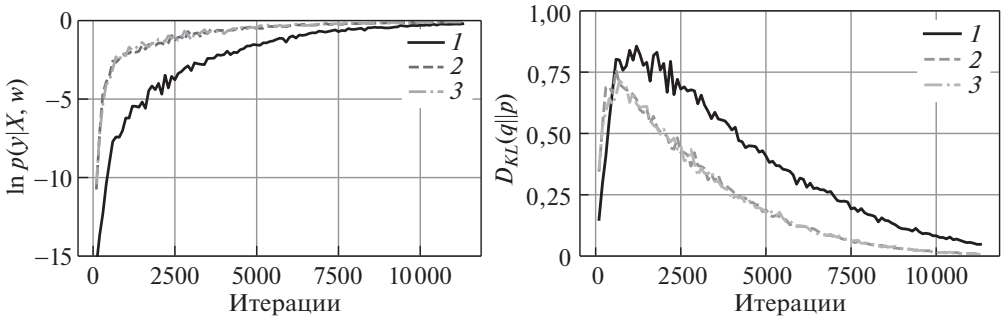

Рис. 1. Структура (11) модели ученика g. Слева: правдоподобие выборки в зависимости от номера итерации при обучении. Справа: дивергенция Кульбака–Лейблера между вариационным и априорным распределениями параметров модели.

где $\beta = 0{,}1$ — уровень шума в данных. В эксперименте число признаков $n = 10$, для обучения и тестирования было сгенерировано $m_{\text{train}} = 900$ и $m_{\text{test}} = 124$ объекта.

В качестве модели учителя рассматривалась модель — многослойный перцептрон с двумя скрытыми слоями (4). Матрицы линейных преобразований имеют размер:

$$\mathbf{U}_1 \in \mathbb{R}^{100\times 10}, \quad \mathbf{U}_2 \in \mathbb{R}^{50\times 100}, \quad \mathbf{U}_3 \in \mathbb{R}^{1\times 50}.$$

В качестве функции активации была выбрана функция активации ReLu. Модель учителя предварительно обучена на основе вариационного вывода (6), где в качестве априорного распределения параметров выбрано стандартное нормальное распределение.

В качестве модели ученика были выбраны две конфигурации. Первая конфигурация получается путем удаления нейронов в модели учителя:

$$g(\mathbf{x}) = \boldsymbol{\sigma} \circ \mathbf{W}_3\boldsymbol{\sigma} \circ \mathbf{W}_2\boldsymbol{\sigma} \circ \mathbf{W}_1\mathbf{x}, \tag{11}$$

где $\boldsymbol{\sigma}$ является нелинейной функцией активации, а матрицы линейных преобразований имеют размер:

$$\mathbf{W}_1 \in \mathbb{R}^{10\times 10}, \quad \mathbf{W}_2 \in \mathbb{R}^{10\times 10}, \quad \mathbf{W}_3 \in \mathbb{R}^{1\times 10}.$$

В качестве функции активации была выбрана функция активации ReLu.

На рис. 1 сравниваются модели ученика со структурой (11). Представлено сравнение разных моделей: модель без дистилляции (график *1*), где в качестве априорного распределения выбирается стандартное нормальное распределение; модель с частичной дистилляцией (график *2*), где в качестве среднего значения параметров выбираются параметры согласно (9), а ковариационная матрица была приравнена к единичной матрице; модель с полной дистилляцией (график *3*) согласно (9). Видно, что модели ученика, где в качестве априорного распределения выбраны распределения, основанные

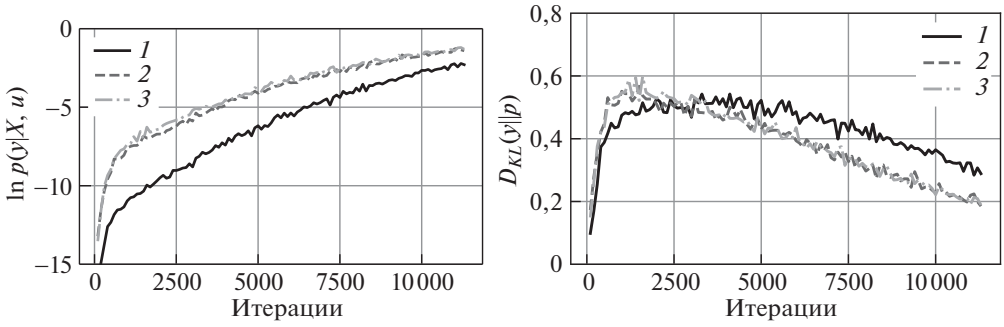

Рис. 2. Структура (12) модели ученика g. Слева: правдоподобие выборки в зависимости от номера итерации при обучении. Справа: дивергенция Кульбака–Лейблера между вариационным и априорным распределениями параметров модели.

на апостериорном распределении учителя, имеют большее правдоподобие, чем модель, где в качестве априорного распределения выбрано стандартное нормальное. Также заметим, что использование параметра среднего из апостериорного распределения дает основной вклад при дистилляции, так как качество моделей без дистилляции и с полной дистилляцией совпадает.

Вторая конфигурация получается путем удаления слоя модели учителя:

$$g(\mathbf{x}) = \boldsymbol{\sigma} \circ \mathbf{W}_2 \boldsymbol{\sigma} \circ \mathbf{W}_1 \mathbf{x}, \tag{12}$$

где $\boldsymbol{\sigma}$ является нелинейной функцией активации, а матрицы линейных преобразований имеют размер:

$$\mathbf{W}_1 \in \mathbb{R}^{50\times 10}, \quad \mathbf{W}_2 \in \mathbb{R}^{1\times 50}.$$

В качестве функции активации была выбрана функция активации ReLu.

На рис. 2 сравниваются модели ученика со структурой (12). Аналогично рис. 1 на рис. 2 представлено сравнение модели без дистилляции (график *1*, модели с дистилляцией параметра среднего значение (график *2*) и модели с полной дистилляцией (график *3*). В рамках данного эксперимента по дистилляции модели учителя в модель ученика с меньшим числом параметров получены результаты, которые подтверждают, что задание априорного распределения параметров ученика позволяет улучшить число итераций при выборе оптимальных параметров модели ученика.

4.2. Выборка FashionMnist

В рамках данного эксперимента проводился анализ байесовского подхода к дистилляции на реальных данных. В качестве реальных данных выбрана выборка FashionMnist [19], которая является задачей классификации изображений на 10 классов.

В качестве модели учителя рассматривалась модель многослойный перцептрон с двумя скрытыми слоями (4). Матрицы линейных преобразований

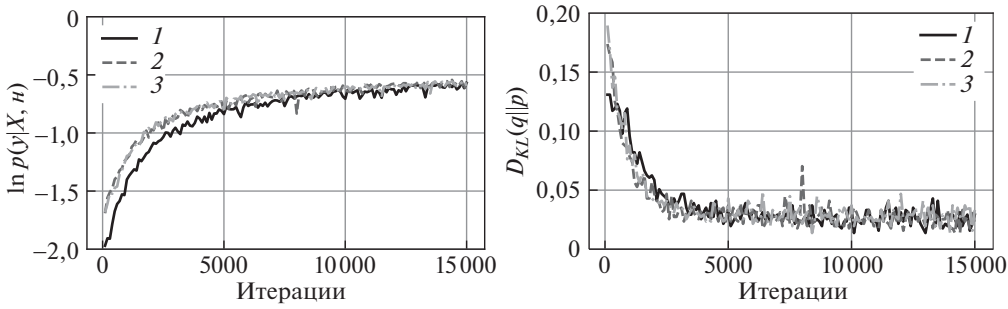

Рис. 3. Слева: правдоподобие выборки в зависимости от номера итерации при обучении. Справа: дивергенция Кульбака–Лейблера между вариационным и априорным распределениями параметров модели.

имеют размер:

$$\mathbf{U}_1 \in \mathbb{R}^{800\times 784}, \quad \mathbf{U}_2 \in \mathbb{R}^{50\times 800}, \quad \mathbf{U}_3 \in \mathbb{R}^{10\times 50}.$$

В качестве функции активации была выбрана функция активации ReLu. Модель учителя предварительно обучена на основе вариационного вывода (6), где в качестве априорного распределения параметров выбрано стандартное нормальное распределение.

В качестве модели ученика были выбрана конфигурация с одним скрытым слоем (12), где матрицы линейных преобразований имеют размер:

$$\mathbf{W}_1 \in \mathbb{R}^{50\times 784}, \quad \mathbf{W}_2 \in \mathbb{R}^{50\times 10}.$$

В качестве функции активации была выбрана функция активации ReLu.

На рис. 3 сравниваются модели ученика с разными априорными распределениями параметров. Аналогично синтетическому эксперименту модель, где в качестве априорного распределения использовалось стандартное нормальное распределение, сравнивалась с моделью, где параметры распределения определялись на основе формулы (10). Видно, что у моделей с заданием априорного распределения на основе апостериорного распределения параметров учителя правдоподобие выборки выше, чем у модели, где в качестве априорного распределения выбрано стандартное нормальное распределение.

В табл. 2 представлен результат вычислительного эксперимента. Для численного сравнения качества моделей выбрана разность площадей графика $p\big(\mathbf{y}|\mathbf{X},\mathbf{u}\big)$ между моделью без дистилляции и моделями с частичной дистилляцией и полной дистилляцией соответственно:

$$S = \sum_s p\big(\mathbf{y}|\mathbf{X},\mathbf{u}_{\mathrm{s}}^s\big) - p\big(\mathbf{y}|\mathbf{X},\mathbf{u}_{\mathrm{ds}}^s\big), \tag{13}$$

где $\mathbf{u}_{\mathrm{s}}^s, \mathbf{u}_{\mathrm{ds}}^s$ обозначают параметры модели ученика и модели дистиллированного ученика после s-й итерации оптимизационного процесса. Заметим, что площадь S имеет знак: чем большее положительное число, тем дистиллированная модель лучше, чем модель, построенная без учителя. Если площадь S

Таблица 2. Сводная таблица результатов вычислительного эксперимента

	Учитель	Модель ученика без дистилляции	Модель ученика с частичной дистилляцией	Модель ученика с полной дистилляцией
Эксперимент на синтетической выборке (удаление нейрона)				
Структура	$[10, 100, 50, 1]$	$[10, 10, 10, 1]$	$[10, 10, 10, 1]$	$[10, 10, 10, 1]$
Число параметров	6050	210	210	210
Разность площадей S	–	0	16 559	16 864
Эксперимент на синтетической выборке (удаление слоя)				
Структура	$[10, 100, 50, 1]$	$[10, 50, 1]$	$[10, 50, 1]$	$[10, 50, 1]$
Число параметров	6050	550	550	550
Разность площадей S	–	0	23 310	25 506
Эксперимент на выборке FashionMnist				
Структура	$[784, 800, 50, 10]$	$[784, 50, 10]$	$[784, 50, 10]$	$[784, 50, 10]$
Число параметров	667 700	39 700	39 700	39 700
Разность площадей S	–	0	1165	1145

принимает отрицательное значение, то, значит, модель без дистилляции является лучше, чем модель с дистилляцией. В рамках вычислительного эксперимента видно, что площадь S под графиками принимает положительные значения, т.е. модели ученика, полученные при помощи дистилляции, являются лучше, чем модель ученика без дистилляции.

Код вычислительного эксперимента доступен по ссылке `https://github.com/andriygav/BayesianDistilation`.

5. Заключение

В данной статье проанализирована байесовская дистилляция модели учителя в модель ученика на основе вариационного вывода. В рамках данной статьи дистилляция основывается на задании априорного распределения параметров модели ученика. Априорное распределение параметров модели ученика задается на основе апостериорного распределения параметров модели учителя. Механизм преобразования структуры модели учителя в структуру модели ученика представлен в теореме 1 и теореме 2.

Теорема 1 описывает механизм приведения пространства параметров модели учителя к пространству параметров модели ученика в случае, если число слоев совпадает, но размер слоев различается. Теорема 2 описывает механизм приведения пространства параметров модели учителя к пространству параметров модели ученика в случае, если число слоев различается.

В вычислительном эксперименте сравнивается модель ученика, которая обучена без использования распределения параметров учителя, и модель ученика, где в качестве априорного распределения параметров выбрано апо-

стериорное распределение параметров модели учителя после приведения. В табл. 2 показано, что модели ученика с заданием априорного распределения параметров на основе апостериорного распределения параметров учителя сходятся быстрее, что подтверждается положительным значением метрики (13), которая введена для численного сравнения модели без дистилляции с дистиллированной моделью.

СПИСОК ЛИТЕРАТУРЫ

1. *Krizhevsky A., Sutskever I., Hinton G.* ImageNet Classification with Depp Convolutional Neural Networks // Proc. 25th Int. Conf. on Neural Information Procescing Systems. 2012. V. 1. P. 1097–1105.
2. *Simonyan K., Zisserman A.* Very Deep Convolutional Networks for Large-Scale Image Recognition // Int. Conf. on Learning Representations. San Diego. 2015.
3. *He K., Ren S., Sun J., Zhang X.* Deep Residual Learning for Image Recognition // Proc. IEEE Conf. on Computer Vision and Pattern Recognition. Las Vegas. 2016. P. 770–778.
4. *Devlin J., Chang M., Lee K., Toutanova K.* BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding // Proc. 2019 Conf. North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Minnesota. 2019. V. 1. P. 4171–4186.
5. *Vaswani A., Gomez A., Jones L., Kaiser L., Parmar N., Polosukhin I., Shazeer N., Uszkoreit J.* Attention Is All You Need // In Advances in Neural Information Processing Systems. 2017. V. 5. P. 6000–6010.
6. *Al-Rfou R., Barua A., Constant N., Kale M., Raffel C., Roberts A., Siddhant A., Xue L.* mT5: A Massively Multilingual Pre-trained Text-to-text Transformer // Proc. 2021 Conf. North American Chapter of the Association for Computational Linguistics: Human Language Technologies. 2021. P. 483–498.
7. *Brown T., et al.* GPT3: Language Models are Few-Shot Learners // Advances in Neural Information Processing Systems. 2020. V. 33. P. 1877–1901.
8. *Zheng T., Liu X., Qin Z., Ren K.* Adversarial Attacks and Defenses in Deep Learning // Engineering. 2020. V. 6. P. 346–360.
9. *Hinton G., Dean J., Vinyals O.* Distilling the Knowledge in a Neural Network // NIPS Deep Learning and Representation Learning Workshop. 2015.
10. *Vapnik V., Izmailov R.* Learning Using Privileged Information: Similarity Control and Knowledge Transfer // J. of Machine Learning Research. 2015. V. 16. P. 2023–2049.
11. *Lopez-Paz D., Bottou L., Scholkopf B., Vapnik V.* Unifying Distillation and Privileged Information // Int. Conf. on Learning Representations. Puerto Rico. 2016.
12. *Burges C., Cortes C., LeCun Y.* The MNIST dataset of handwritten digits. 1998. http://yann.lecun.com/exdb/mnist/index.html
13. *Huang Z., Naiyan W.* Like What You Like: Knowledge Distill via Neuron Selectivity Transfer // arXiv:1707.01219. 2017.
14. *Hinton G., Krizhevsky A., Nair V.* CIFAR-10 (Canadian Institute for Advanced Research) // http://www.cs.toronto.edu/ kriz/cifar.html
15. *Deng J., et al.* Imagenet: A Large-scale Hierarchical Image Database // Proc. IEEE Conf. on Computer Vision and Pattern Recognition. Miami. 2009. P. 248–255.

16. *LeCun Y., Denker J., Solla S.* Optimal Brain Damage // Advances in Neural Information Processing Systems. 1989. V. 2. P. 598–605.
17. *Graves A.* Practical Variational Inference for Neural Networks // Advances in Neural Information Processing Systems. 2011. V. 24. P. 2348–2356.
18. *Grabovoy A.V., Bakhteev O.Y., Strijov V.V.* Estimation of Relevance for Neural Network Parameters // Informatics and Applications. 2019. V. 13 No. 2. P. 62–70.
19. *Rasul K., Vollgraf R., Xiao H.* Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms // arXiv preprint arXiv:1708.07747. 2017.

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

Поступила в редакцию 20.01.2021

После доработки 25.06.2021

Принята к публикации 30.06.2021

© 2021 г. С.П. ГРАЧЕВ (sergey@grachev.me),
А.А. ЖИЛЯЕВ (zhilyaev.alexey@gmail.com),
В.Б. ЛАРЮХИН (vladimir.larukhin@live.ru),
Д.Е. НОВИЧКОВ (dmitriy.novichkov@gmail.com),
В.А. ГАЛУЗИН (galuzin@.kg.ru)
(Самарский государственный технический университет),
Е.В. СИМОНОВА, канд. техн. наук (simonova@kg.ru)
(Самарский национальный исследовательский университет им. академика С.П. Королева),
И.В. МАЙОРОВ, канд. техн. наук (imayorov@kg.ru),
П.О. СКОБЕЛЕВ, д-р. техн. наук (petr.skobelev@gmail.com)
(Самарский федеральный исследовательский центр РАН, Институт проблем управления сложными системами РАН, Самара)

МЕТОДЫ И СРЕДСТВА ПОСТРОЕНИЯ ИНТЕЛЛЕКТУАЛЬНЫХ СИСТЕМ ДЛЯ РЕШЕНИЯ СЛОЖНЫХ ЗАДАЧ АДАПТИВНОГО УПРАВЛЕНИЯ РЕСУРСАМИ В РЕАЛЬНОМ ВРЕМЕНИ[1]

Формулируется постановка задачи адаптивного управления ресурсами предприятий в реальном времени. Приводятся примеры современных задач адаптивного управления ресурсами в различных областях применений, показывающие размерность и другие особенности решаемых задач. Формулируются требования к решению рассматриваемых задач и предлагаются общие принципы, повышающие адаптивность управления ресурсами. Проводится краткий анализ существующих подходов и показываются их ограничения. Предлагается новая методология решения рассматриваемых задач, дается обзор разработок и обсуждается первый опыт ее применения при создании интеллектуальных систем управления ресурсами. Показывается возможность решения экстремально сложных задач управления ресурсами за счет модифицированной на основе онтологий концепции сети потребностей и возможностей, в рамках которой расписание строится как «конкурентное равновесие» на виртуальном рынке унифицированной мультиагентной системы, настраиваемой на конкретное предприятие с использованием прикладных онтологий. Даются пример построения прототипа интеллектуальной системы управления ресурсами многоспутниковой группировки ДЗЗ и результаты ее экспериментальных исследований. Предлагается подход к оценке качества и эффективности разрабатываемых моделей и методов адаптивного построения расписаний при управлении ресурсами в реальном времени. Показываются перспективы дальнейшего развития подхода для решения сложных задач адаптивного управления ресурсами.

[1] Работа поддержана Российским фондом фундаментальных исследований (проект № 20-37-90052).

Ключевые слова: задача адаптивного управления ресурсами, онтологии, мультиагентные технологии, интеллектуальные системы, сети потребностей и возможностей, виртуальный рынок, конкурентное равновесие, реальное время.

DOI: 10.31857/S0005231021110039

1. Введение

Новые потребности предприятий в повышении эффективности бизнеса в условиях современной сетевой экономики и стремительного прогресса компьютерных технологий приводят к появлению новых постановок задач управления ресурсами в сравнении с традиционными, которые было принято рассматривать в исследовании операций [1–3]. Одна из таких новых задач связана с переходом к управлению ресурсами в реальном времени, когда от самого момента времени принятия решения напрямую зависят качество и эффективность создания продукта или реализации оказываемой услуги.

Целый ряд таких задач появляется в связи с переходом к юбер-подобному управлению ресурсами в реальном времени, где есть высокая сложность, априорная неопределенность и турбулентная динамика изменений спроса и предложения, например, когда на предприятие часто приходят новые заказы с индивидуальными требованиями или подключаются и выходят из доступа ресурсы, уже запланированные заказы приостанавливаются или отменяются, изменяются или запускаются новые технологические процессы, меняются задачи, поставщики комплектующих и т.д.

Примеров предприятий, где рассматриваемая постановка задачи оказывается актуальна и значима, становится все больше: это он-лайн такси для пассажиров и перевозки грузов, заводы и фабрики, работающие по индивидуальным заказам, доставка товаров интернет-магазинов и т.п. Можно утверждать, что в связи с усиливающимися тенденциями «юберизации», т.е. с переходом к использованию разделяемых ресурсов и экономике реального времени, число таких задач будет продолжать расти.

В этих условиях традиционное «пакетное» планирование на большой горизонт времени, базирующееся, как правило, на классических методах математического программирования или разнообразных эвристиках и метаэвристиках, включая табу-поиск, муравьиные алгоритмы, генетическое программирование и др. [4–6], не дает существенного выигрыша, так как построенный план в условиях высокой неопределенности и динамики возникновения событий может существовать лишь короткое время и далее безвозвратно устаревает, а полное перестроение планов заново по каждому событию требует долгих вычислений, в ходе которых, в свою очередь, могут приходить все новые события.

В этой связи становятся востребованными новые модели и методы адаптивного управления ресурсами в реальном времени, предназначенные для решения задач распределения, планирования, оптимизации, прогнозирования, согласования, мониторинга и контроля использования ресурсов по событиям,

без полного перестроения планов, а только той их части, которая прямо или косвенно задета событием.

При этом планирование ресурсов должно предполагать не столько «инкрементальное» изменение плана, когда новые заказы планируются на свободные ресурсы, добавляясь в открытые интервалы времени («в хвост») уже имеющихся в расписании заказов, сколько выявление и разбор конфликтов между ранее запланированными заказами при условии, что еще есть время для принятия таких решений.

Во втором разделе формулируется содержательная постановка задачи адаптивного управления ресурсами предприятий в реальном времени и рассматриваются примеры такого рода практических задач в различных сферах управления ресурсами. При этом показывается, что решение любой такой сложной задачи – это всегда поиск согласия или консенсуса между всеми участниками процессов управления. К числу таких участников следует относить не только людей, но и любые другие сущности: заказы и ресурсы, продукты и др., что во многом связано с развитием бережливой модели производства. Выделяются типовые критерии, предпочтения и ограничения, показывающие внутренний конфликтный характер решаемых практических задач и необходимость разработки методов и средств, позволяющих находить баланс интересов такого рода участников. Выявляются и показываются ограничения существующих методов и средств планирования и оптимизации, затрудняющих их практическое использование.

В третьем разделе проводится краткий анализ существующих подходов и формулируются их ограничения. Показывается эволюция представлений, моделей и методов планирования и оптимизации для построения оптимального плана в интересах одного центра (консонанс ценностей) к теории игр (конфликт ценностей) и сетевым рыночным моделям конкуренции и кооперации множества участников с юберизацией ресурсов (компромиссу ценностей). Формулируется задача создания интеллектуальных систем управления ресурсами (ИСУР), приходящих на смену классическим Enterprise Resource Planning (ERP) системам, для работы в условиях новой экономики.

В четвертом разделе обобщается методология решения рассматриваемых задач и дается обзор разработок ИСУР в Самарской школе мультиагентных систем. Прослеживается развитие предложенной в [7, 8] концепции сетей потребностей и возможностей, показываются поэтапное расширение состава основных классов агентов и переход к виртуальному рынку агентов со взаимными уступками и компенсациями, в рамках которого расписание строится как «конкурентное равновесие» (динамический останов). Приводятся требования к программной среде для реализации мультиагентных систем в составе ИСУР. Показывается опыт разработки ИСУР для различных применений, которые создавались как новые системы «с нуля».

В пятом разделе рассматривается возможность решения сложных задач управления ресурсами с использованием модифицированной, построенной на основе базовой онтологии управления ресурсами, концепции ПВ-сетей. Выделяются основные классы понятий и отношений в предметной области управ-

ления ресурсами, которые позволяют описывать особенности каждого предприятия, что в свою очередь позволяет создать унифицированную мультиагентную систему, настраиваемую посредством прикладных онтологий и онтологических моделей предприятий на каждое конкретное применение. Дается формализованное описание метода адаптивного построения расписания для решения сложных задач управления ресурсами, включая новые классы агентов, их целевые функции и протоколы переговоров.

В шестом разделе представлена структура инструментального комплекса для создания ИСУР, включающего базу знаний на основе онтологий для построения онтологических моделей предприятий и унифицированную мультиагентную систему для планирования, мониторинга и контроля исполнения планов.

В седьмом разделе рассматривается пример создания прототипа ИСУР для управления многоспутниковыми группировками дистанционного зондирования Земли. Показаны функции системы, представлены результаты моделирования для исследования качества и эффективности разработанного метода адаптивного планирования в сравнении с классическими и эвристическими методами. Обсуждаются особенности коллективного поиска агентами согласованных решений в ходе самоорганизации расписаний за счет выявления и разрешения конфликтов между агентами и проведения переговоров с взаимными уступками для достижения консенсуса.

В восьмом разделе по результатам проведенных компьютерных экспериментов делаются выводы и даются рекомендации по созданию и исследованию ИСУР для практических применений. Формулируется подход к оценке качества и эффективности моделей и методов адаптивного построения расписаний, который может стать основой расчета экономического эффекта по внедрению ИСУР.

В заключение показываются перспективы дальнейшего развития подхода для решения сложных задач адаптивного управления ресурсами.

Результаты настоящего исследования впервые дают системное изложение и обобщают ранее полученные результаты, представленные в [7–13].

2. Задача адаптивного управления ресурсами предприятий в реальном времени

Управление ресурсами предприятия предполагает связанное решение задач организации деятельности, планирования и оптимизации ресурсов, обеспечения производственных процессов работы всем необходимым, мониторинга и контроля хода исполнения задач, а также развития предприятия [2, 14].

Рассмотрим примеры практических задач управления ресурсами в реальном времени, связанных с грузовыми перевозками и мобильными бригадами, машиностроительным производством, цепочками поставок и другими областями, чтобы характеризовать требования к управлению ресурсами и размерность пространства поиска решений.

— **Управление грузоперевозками:** от 500 грузовиков на дороге, 100 заказов приходит в день в заранее не известные моменты времени, горизонт поездок от одного до 30 дней и более, 5000 точек доставки, продолжительность обработки заказа 3–15 с, критерии: время и стоимость перевозки, комфортность водителей, равномерность нагрузки на машины и водителей, социальная справедливость, соблюдение режимов работы водителей, необходимость учета возможных задержек на погрузке или разгрузке и штрафов за опоздание, различные типы грузов и оборудования, требования возврата тары, время работы промежуточных складов и магазинов, порядок загрузки и центровка грузов в кузове, износ шин, наличие рекламных обязательств, качество услуг для потребителей, риски, прибыль компании.

— **Управление фабриками:** сборка двигателя самолета – 50 тыс. деталей, 30–40 цехов на предприятии, каждый цех – 150 рабочих и 300 единиц оборудования, ежедневно в цех поступает 10 новых заказов, одно сменно-суточное задание рабочего может включать до 100 операций, для обработки каждой деталесборочной единицы требуется от 10 до 300 технологических операций, горизонт стратегического планирования составляет 3–5 лет, оперативного планирования и диспетчеризации – от одного дня до 6 мес, план одного цеха может включать до 500 тыс. связанных задач, желаемое время обработки события – от 2 с до 5 мин, учет особенностей технологических процессов, компетенций и опыта рабочих, возможность усиления технологических операций за счет привлечения дополнительных рабочих, возникновение брака, наличие на складе комплектующих и материалов и т.д.

— **Управление цепочками поставок:** 5–7 фабрик в сети, 300 складов промежуточного хранения, сотни каналов транспортировки, 1000 магазинов, изменяющих свои заказы на горизонте в 7–10 дней, 10 000 товаров, движущихся в сети с разной скоростью, сотни и тысячи новых заказов, поступающих ежедневно, одно событие требуется обрабатывать от 20 с до 15 мин, горизонт планирования от 3 дней до 3 мес с критериями минимизации себестоимости производства заказов, включая стоимость производства, транспортировки и хранения, а также минимальных штрафов за отклонение от заданных сроков доставки с учетом изменяющегося прогноза продажи продукции.

— **Управление мобильными бригадами:** число событий в день: от 50 до 250 заявок, в среднем около 100 заявок; 43 бригады для обслуживания региона в 3,3 млн человек; горизонт планирования — 8–12 ч (длительность смены); требуемое время обработки события – до 1 мин; критерии принятия решений: максимизация количества устраненных аварий по заявкам при минимальном пробеге и простое, компетенция и оснащенность техникой бригады и др.

— **Управление грузопотоком Международной космической станции:** номинальный план полета рассчитывается на 6 мес (2 экспедиции), грузопоток и программа полета – на год, долгосрочная программа – на 3 года, используется 3500 типов грузов, в числе которых топливо, вода, продовольствие для космонавтов, приборы и оборудование для научных экспериментов, запасные инструменты и принадлежности, необходим учет состава экспедиций, типов грузов, норм потребления, объема складов на МКС и т.д. Критерия-

ми планирования являются своевременность доставки грузов, максимальное заполнение кораблей и т.п.

— **Управление оперативным движением поездов:** полигон Москва-Санкт-Петербург содержит 49 станций, 48 перегонов, 3500 блок-участков, в день проходит около 800 поездов; горизонт планирования – сутки, время начального планирования – не более 4 мин. План каждого поезда включает 40–50 крупных операций при ограничениях на число путей на станциях, по соблюдению интервалов движения, времени хода, продолжительности стоянок, разгона и торможения, а также приоритетов поездов, опасности грузов и др. В течение дня может произойти до 50 непредвиденных событий (поломки путей, задержки проездов и т.д.), обработка события – до 3 мин.

— **Управление проектами:** на подразделение в 150 инженеров в год может приходиться 25–40 проектов, каждый проект предполагает не менее 300–500 задач. В день на исполнителя поступает 2–5 новых задач и возникает от 5 до 20 событий: при задержке финансирования требуется маневр по проектам, больше времени отнимает согласование технической документации, уточняются задачи, подводят смежники и т.п. Горизонт планирования составляет от 2–3 мес до 3 лет, расписание может содержать около 10 000 связных задач, требуемое время обработки события — от 2 с до 5 мин. Требуется учитывать приоритеты, трудоемкость задач и связи между ними, индивидуальный календарь каждого исполнителя, его производительность и опыт.

Ключевым этапом сквозного процесса управления ресурсами в реальном времени является адаптивное планирование, под которым в рамках разработанной методологии предлагается понимать не традиционное централизованное нахождение одного «глобально-оптимального» плана в интересах центра, а согласованную выработку и синхронизацию «по ситуации» (событиям) множества «локально-оптимальных» планов участников в их взаимодействии, с учетом того, что у каждого имеются свои собственные интересы, критерии принятия решений, предпочтения и ограничения.

Методология такого распределенного подхода может быть естественным путем распространена на интересы, предпочтения и ограничения не только людей, но и любых физических или абстрактных сущностей, таких как подразделения и предприятие в целом, заказы, станки и другое оборудование, грузовики, комплектующие и материалы и т.п. Можно, конечно, задать вопрос: “А зачем награждать «интересами» неживые сущности?” Ответ дан в методологии «бережливого производства» — нужна «забота» об эффективном использовании каждой такой сущности в производственной системе, чтобы постараться не допустить избытка, простоя или дефицита использования каждого ресурса.

Более того, опыт практического решения представленных выше сложных задач управления ресурсами показывает, что под «хорошим» планом следует понимать такое производственное расписание использования ресурсов, которое обеспечивает баланс интересов («консенсус») между участниками процессов управления в каждой конкретной ситуации с учетом актуального для

всех на этот момент состава и значений критериев, предпочтений и ограничений.

Такие собственные критерии, предпочтения и ограничения есть у заказчиков, финансистов, производственников, логистов, инженеров и рабочих, водителей и других сотрудников предприятия, для которых типичны следующие критерии:

— обеспечить качество выполнения работ;

— выполнить все заказы в срок;

— минимизировать себестоимость работ;

— минимизировать риски срыва заказов;

— обеспечить равномерность загрузки ресурсов;

— минимизировать стоимость исполнения заказов;

— гарантировать выплату зарплаты рабочим в конце месяца;

— вовремя выполнить ремонт оборудования и т.д.

Наличие участников со своими интересами делает рассмотренные задачи не только многокритериальными по своей природе, но и, по определению, конфликтными, причем одним из равных участников может быть и само предприятие («центр»), отстаивающий интересы объединенного «целого» в поиске компромисса с интересами отдельных частей.

3. Краткий анализ существующих подходов к решению сложных задач адаптивного управления ресурсами

Традиционные информационно-управляющие системы класса Enterprise Resource Planning (ERP) разработки таких известных компаний как SAP, BAAN, Oracle, 1С, Галактика и многих других хотя и называются системами планирования ресурсов предприятий, но в реальности остаются в основном учетными системами с весьма ограниченными возможностями пакетного планирования.

Под «капотом» указанных систем обычно используются известные зарубежные программы CPLEX, ILOG, MOSEK, Gurobi, Knitro, Xpress Optimizer и др., основанные на использовании классических моделей, методов и алгоритмов решения задач управления ресурсами. В большинстве таких систем считается, что заказы и ресурсы заранее известны и не изменяются в ходе вычислений, так что даже небольшое изменение ситуации вызывает необходимость полного пересчета планов с существенными затратами времени.

С ростом сложности решаемых задач бизнеса и в условиях динамического изменения спроса и предложения на рынке практическое использование классических методов и средств становится затруднительным вследствие ситуационного характера принимаемых решений по управлению ресурсами с учетом индивидуальных особенностей заказов и ресурсов, а также специфических предметных знаний, известных лишь исполнителям «на земле», и других требований участников процессов управления. Также не решают указанные выше проблемы и известные эвристические подходы, предназначенные для

поиска допустимого решения, такие как жадные алгоритмы локального поиска, генетические алгоритмы, муравьиная оптимизация и т.п. [4–6, 16–19].

Кроме того, до сих пор принято считать, что все сведения о предприятии и данные о его работе должны быть сосредоточены в центральном компьютере, где выполняются расчеты планов на основе заранее заданных и неизменяющихся заказов и ресурсов, а в результате должен быть создан один общий глобально-оптимальный план всего предприятия в целом. При этом предполагается, что обработка должна вестись в пакетном режиме на основе единственной целевой функции, обычно отражающей интересы центра (собственника), при ограничениях, задаваемых только в виде неравенств, расчет вариантов в течение десятков часов вполне приемлем, а исполнители только ждут указаний (отметим, что в ряде случаев требуется задать критерии работы исполнителей, причем ограничения могут задаваться таблично, в виде правил или алгоритмами).

Но на практике ситуация обычно совершенно другая – окружающие нас социально-технические системы становятся все более сложными и распределенными, причем «части» системы приобретают все большую автономность в принятии решений, т.е. изначально признается, что исполнители обладают собственными интересами и предпочтениями и главным принципом управления становится согласованное взаимодействие между всеми участниками процессов управления как по «вертикали», так и по «горизонтали».

Можно утверждать, что классические математические методы комбинаторной оптимизации и различных эвристик с точки зрения процессов принятия коллективных решений со многими участниками по-прежнему отвечают **модели консонанса** (единства) ценностей, в которой всем участникам присущи одни и те же критерии (ценности), в данном случае пусть сколько угодно сложная, но одна целевая функция, как это было, например, во времена Госплана СССР.

Этой модели противостоит модель принятия коллективных решений **в условиях конфронтации**, как, например, во время войны или спортивных состязаний, а также жесткой конкуренции, когда у каждого участника есть свои собственные ценности и кто-то из сторон в результате своих планов действий и взаимодействия обязательно должен выиграть, а кто-то проиграть, что находит выражение и реализуется в теории игр.

Однако в последнее время в бизнесе и технологиях на первый план выходят идеи **поиска согласия (консенсуса)**, которое может строиться как по горизонтали, так и по вертикали, в ходе выработки и согласования коллективных решений, когда у всех участников ценности могут быть разные, но в рамках диалога стороны могут выявлять и разрешать конфликты и находить компромиссы путем переговоров и взаимных уступок. Это отвечает требованиям растущей сетевой экономики с совместным использованием ресурсов (shared economy), что ведет к тотальной юберизации ресурсов.

В этой новой модели экономики и поиска согласия сочетаются возможности как конкуренции, так и кооперации заказов и ресурсов на рынке, что отражается в недавно появившемся термине «coopetition» (от англ.

«Competition» — конкуренция и «Cooperation» — кооперация, здесь их сочетание). Например, небольшие грузовые компании могут конкурировать друг с другом на рынке, но при поступлении большого заказа объединять свои ресурсы. Если одна из таких компаний, имея все грузовики на юге, неожиданно получает заказ на севере, где сосредоточены простаивающие грузовики другой компании, разумнее было бы разделить данный заказ со второй компанией и получить хотя бы 50% возможной прибыли, чем полностью потерять такой заказ. Тенденция перехода от одного «глобального» плана к «распределенному» множеству, причем непрерывно самосинхронизируемых планов, актуальна и для одного предприятия, где подразделения также имеют собственные интересы и планы. В условиях неопределенности и высокой динамики «оптимальные» производственные планы, построенные в центре управления крупного предприятия, оказываются нежизнеспособными уже на уровне цехов из-за неучтенных важных особенностей производственной ситуации, которая могла измениться в ходе расчетов, что требует ручной «доводки» планов на рабочих местах с участием экономистов и диспетчеров, мастеров и рабочих в цехах. Это совершенно не означает, что нет общего плана на уровне завода в целом — такой план просто имеет приблизительную точность, а точный план может быть собран на момент выдачи запроса на основе поступающих планов подразделений. Однако такой план в ту же секунду устареет — в подразделениях будут происходить новые события, вызывающие перестройку их планов и ресинхронизацию с другими.

Важным шагом в развитии моделей новой экономики является теория активных систем (ТАС) [20], развивающая идеи программно-целевого планирования, в рамках которой кроме «центра» свои интересы и свободу выбора могут иметь и «исполнители» на местах, например, при выполнении комплексных проектов. При этом исполнители стремятся к выбору таких своих состояний, которые являются наилучшими с точки зрения их предпочтений при заданных управляющих воздействиях центра, а управляющие воздействия, в свою очередь, зависят от состояний управляемых субъектов. Однако и в данном случае основные решения принимает центр, а горизонтальные переговоры между самими исполнителями для выработки решений не предусмотрены.

В этой связи наиболее перспективными для решения сложных задач адаптивного управления ресурсами становятся модели и методы самоорганизации множества агентов при построении расписаний, где каждый агент сам принимает решения, но готов идти на переговоры и уступки ради общей цели или партнера. Такого рода модели должны предусматривать как вертикальные (центр-исполнители), так и горизонтальные (исполнитель-исполнитель) взаимодействия для достижения коллективного баланса интересов («консенсуса»).

На сегодня проблемы разработки и применения такого рода методов и средств остаются малоизученными, но являются весьма перспективными как для создания интеллектуальных систем управления ресурсами нового поколения, так и построения сетецентрических платформ и цифровых эко-систем

для решения задач управления ресурсами [20, 21]. Разработанные модели и методы могут найти применение и для решения других типов сложных задач: от проектирования сложных технических изделий до понимания текстов, распознавания образов или извлечения знаний.

В то же время проведенный анализ позволяет сформулировать следующие ключевые особенности и ограничения, затрудняющие применение классических методов комбинаторной оптимизации и различных эвристик:

— необходимо учитывать многочисленные особенности предметной области предприятия: заказов, задач и технологических операций, станков, рабочих и т.д.;

— требуется обеспечивать решение задач высокой размерности пространства решений (сотни ресурсов и тысячи заказов на большой горизонт планирования);

— каждый участник имеет множество индивидуальных предпочтений, ограничений и критериев, которые еще и могут изменяться с течением времени;

— планирование заказов редко осуществляется в пакетном режиме, но все чаще в «скользящем режиме», с наложением на исполнение уже построенных планов;

— при появлении непредвиденных событий (поломка ресурсов или приход новых заказов) все расписание не должно пересчитываться заново с «нуля», а требуется адаптивное перепланирование с разбором конфликтов;

— качество решений по управлению ресурсами зависит от момента времени и любые задержки сразу приводят к затовариванию, простою или дефициту ресурсов;

— необходимо уметь ситуативно балансировать интересы, чтобы добиться взвешенного (гармоничного) решения, учитывающего интересы всех участников;

— необходимо уметь объяснить решение пользователю и дать ему возможность вмешаться в процесс планирования на любой стадии принятия решений;

— необходимо обеспечить пользователю возможность интерактивно исправить только часть расписания без полного пересчета и т.п.

Таким образом, для решения поставленной задачи требуется создание нового класса интеллектуальных систем управления ресурсами (ИСУР), которые реализуют цикл Деминга по управлению ресурсами, дополняя, а в перспективе и заменяя, менеджеров в этой роли [15].

В отличие от традиционных ERP систем ИСУР должны управляться целями, задаваемыми на основе критериев, предпочтений и ограничений, состав и важность которых у каждого участника могут меняться по ходу развития ситуации. Например, для предприятия в начале года наивысший приоритет имела прибыль от выполнения имеющихся заказов, а затем — крупный заказ важного постоянного клиента или более равномерная загрузка рабочих для

своевременной выплаты зарплаты, и такая смена критериев должна также вызывать адаптивное перепланирование ресурсов.

Эти системы призваны расширить и дополнить функции классических ERP систем, которые на сегодня выполняют лишь функции информационно-учетных систем, в редких случаях реально охватывая контур стратегического и оперативного управления.

4. Предлагаемая методология решения задач адаптивного управления ресурсами в реальном времени

Рассмотренные задачи построения ИСУР вызывают растущий интерес исследователей и разработчиков интеллектуальных систем к альтернативным подходам, моделям и методам, продуктам и технологиям в исследовании операций. Начиная с 1980–1990 гг. можно отметить возрастание количества исследований, направленных на использование мультиагентных технологий для моделирования процесса поиска решений по построению расписаний во взаимодействии его участников [23–25]. В начале 2000-х годов были предприняты попытки применения мультиагентных технологий для перехода к распределенному решению задач управления ресурсами (от англ. Distributed Problem Solving). Данный подход основан на разделении исходной сложной задачи на несколько подзадач с последующим объединением полученных частных решений. В это же время был выполнен ряд исследований, в которых теория игр применялась к мультиагентным системам распределения ресурсов. В развитие существующих комбинаторных подходов в рамках направления по исследованию операций сформировалась целая новая область решения задач распределенной оптимизации в системах с ограничениями Distributed Constraint Optimization Problem (DCOP) [26]. Несмотря на то что при этом продолжает доминировать централизованный подход «сверху-вниз», в области планирования и оптимизации ресурсов разработаны распределенные модели, методы и алгоритмы с использованием мультиагентных технологий. К ним можно отнести методы Asynchronous Distributed Constraint Optimization (ADOPT), Optimal Asynchronous Partial Overlay (OptAPO), Distributed pseudo-tree optimization (DPOP), Asynchronous Backtracking (ABT), предназначенные для решения задач управления в системах, имеющих сетевую структуру, методы роевой оптимизации (Particle Swarm Optimization), в которых несколько иначе трактуется понятие «роя» агентов, и ряд других [27–29].

Однако предложенные подходы во многом остаются «централизованными» в то время, когда сами «исполнители» могли бы выявлять конфликты и договариваться между собой путем переговоров со взаимными уступками.

Переход к использованию моделей, методов и алгоритмов самоорганизации при построении расписаний был сделан в начале 2000-х годов в работах Самарской школы мультиагентных систем в связи с разработкой концепции сетей потребностей и возможностей (ПВ-сетей) [7–9]. Был предложен набор базовых классов и функций агентов ПВ-сети (в том числе агентов потреб-

ностей и возможностей), разработан набор методов взаимодействий агентов для формирования расписаний в виде «конкурентных равновесий» на виртуальном рынке системы, получаемых в ходе взаимных уступок и компенсаций, реализованы различные приложения по управлению ресурсами.

Суть разработанной мультиагентной технологии построения расписаний на базе концепции ПВ-сети состоит в формировании решения любой сложной задачи управления ресурсами в ходе конкуренции и кооперации агентов потребностей и возможностей на виртуальном рынке системы.

Например, в мультиагентной системе управления грузовыми перевозками роль потребностей могут играть заказы, которые ищут себе грузовики, но, в свою очередь, каждому грузовику требуются водитель и топливо, а водителю — ночлег и питание. Можно продолжить этот ряд и ввести новые потребности и возможности следующей степени детализации процессов согласованного принятия решений, например ввести потребности на техническое обслуживание и ремонт грузовиков, смену водителей с учетом наличия у них международных паспортов или подбор шин с учетом их пробега и состояния трассы для предстоящей поездки.

В качестве базовых агентов, согласно холоническому подходу PROSA, были выбраны агенты заказов, продуктов и ресурсов, штабной агент [30]. На этапе поиска потребностей и возможностей было предложено использовать протоколы аукционно-подобных переговоров, построенные по типу Contract-net протокола [31, 32].

В 2009–2010 гг. были опубликованы две монографии [33, 34], которые обобщили первые результаты исследования моделей, методов и алгоритмов работы агентов на виртуальном рынке.

Модель виртуального рынка была сформулирована следующим образом: имеется множество агентов $A = \{A_i\}$, множество целевых функций агентов $C = \{C_i\},\ i = 1, \ldots, n$, где n – количество агентов, и набор задач T. Для любого набора задач T, функция $C_i(T)$ определяется как стоимость выполнения агентом A_i всех задач T. Первоначально каждый агент выбирает некоторый произвольный набор задач, для которого сумма всех расходов агентов не является минимальной, что приводит к неоптимальному распределению. Далее агенты вступают в переговоры с целью улучшения распределения, вследствие чего рано или поздно на виртуальном рынке устанавливается «конкурентное равновесие», которое считается решением задачи с минимальной стоимостью, так как ни одно другое решение не приведет к улучшению результата. Переговоры можно рассматривать как итеративный процесс взаимных уступок и перестановок, на каждом шаге которого агенты заключают «контракт» для обмена задачами и виртуальной валютой.

В [33] была доказана возможность получения глобального оптимума при использовании предлагаемого метода для решения задачи о назначениях и высказаны аргументы в пользу перспективности дальнейшего развития и применения этих методов при решении более сложных NP полных задач планирования и оптимизации. Был отмечен также ряд важных преимуществ данных моделей и методов в случаях, когда классические методы оказыва-

ются не применимы: простота и понятность для разработчиков, устойчивость к изменениям требований, возможность частичного адаптивного изменения планов, органичность распараллеливания и масштабирования и т.д.

В России эта же методология применяется для решения задачи о балансировке нагрузки в грид-сети вычислителей [35, 36]. На примере расчетов грид-сети из 1024 вычислителей показана возможность практического применения разработанных методов и алгоритмов в задачах большой размерности, причем в условиях действия помех, когда другие методы оказываются в принципе не применимы.

В ходе проведенных разработок был введен ряд новых классов агентов и для каждого агента — собственные функции удовлетворенности и бонусов-штрафов, регулирующих их эластичность при уступках для достижения баланса интересов. Например, цель заказа — максимально дешевое или быстрое исполнение, цель ресурса — максимальная загрузка, а цель продукта — меньше пролеживать, но в зависимости от ситуации приходится идти на компромиссы и разменивать стоимость на время и т.п.

В основе разработанной технологии, поддерживающей конкуренцию и кооперацию агентов, лежит возможность для агентов непрерывно, асинхронно и параллельно разрешать возникающие конфликтные ситуации, когда несколько заказов или задач претендуют на использование одного и того же ресурса или продукта (или наоборот), посредством предложения выплаты компенсаций за освобождение слота времени.

Получение компенсации позволяет уступающей стороне конфликта найти себе новое место без потери удовлетворенности или с минимально возможной потерей удовлетворенности при получаемой дополнительной сумме на счет для поиска вариантов улучшения своего состояния в будущем.

Методология построения мультиагентных ПВ-сетей в интеллектуальных системах управления ресурсами предполагает реализацию следующих принципов:

1. Автономность агентов, т.е. наличие у каждого экземпляра агента индивидуальных целей, критериев их достижения, предпочтений и ограничений, а также состояний, отражающих контекст ситуации.
2. Наличие у агента собственных сценариев, методов и средств для достижения целей, что может включать и более традиционные методы, например методы ветвей и границ или машинного обучения.
3. Наличие прямых взаимодействий агентов в виде переговоров на виртуальном рынке системы, в которых выявляются конфликты, ищутся и согласовываются варианты их разрешения путем взаимных уступок, в частности, за счет аукционно-подобных протоколов.
4. Наряду с прямыми переговорами могут использоваться и косвенные, опосредованные через общую сцену, содержащую контекст ситуации и формируемое решение задачи построения расписания. При этом роль сцены как семантической сети состоит в том, чтобы фиксировать отношения между объектами предметной области, что позволяет использовать «топологию» такой сети для сокращения перебора вариантов и

быстрого выяснения того, с какими ближайшими локальными соседями следует разговаривать.

5. Наличие специальной среды, где каждый агент в любой момент времени может стать активным и старается улучшить свое состояние, реагируя на события и обладая проактивностью. В такой среде должен поддерживаться параллельный и асинхронный характер работы каждого экземпляра агента. В этих целях среда может предлагать диспетчер агентов и желтые страницы или «меш»-коммуникацию (от англ. mesh), без общего единого центра, где каждый агент может понять, с кем ему взаимодействовать, проводя локальный анализ отношений семантической сети.
6. Решение проблемы в виде искомого расписания выдается не в виде множества вариантов, полученных в ходе перебора вширь или вглубь и упорядоченных по убыванию значений единственной заданной целевой функции, но как одно, «наилучшее» в данной ситуации, рациональное решение, отражающее баланс целевых функций или интересов (консенсус) агентов участников, достигнутый в виде неулучшаемого «конкурентного равновесия» («динамического останова»), когда агенты продолжают работать, но больше ни один агент не может улучшить свое состояние так, чтобы при этом не ухудшить состояние других агентов и системы в целом.
7. Важной особенностью архитектуры построения системы является высокая «модульность» агентов, дающая возможность быстро дорабатывать логику отдельных агентов или вводить новые классы агентов, чтобы последовательно наращивать сложность системы для отражения сложности выявляемых разнообразных факторов, влияющих на качество услуг или эффективность предприятий.
8. Важной задачей развития мультиагентных ПВ-сетей является создание унифицированной мультиагентной системы, поддерживающей решение задач планирования ресурсов в различных предметных областях, и конструктора онтологий, позволяющего строить онтологические модели любых предприятий на основе базовой онтологии «Управление ресурсами» и прикладных онтологий для управления ресурсами в машиностроении, авиастроении, судостроении, приборостроении, микроэлектронике и т.д.

Новизной технологии является возможность адаптивно пересматривать решения, ранее принятые в ходе планирования, что в особенности важно для работы по событиям в режиме реального времени, в сравнении с обычной практикой first-in/first-out, когда, например, новые заказы размещаются инкрементально в свободные слоты времени на ресурсах, т.е. идет планирование в «хвост» расписания.

Более подробно разработанные модели и методы рассмотрены в [12].

Разработанный подход был впервые использован в 1999 г. при построении мультиагентной системы управления поставками деревянных частей для оформления салонов автомобилей класса «люкс» на фабриках компании

Фольксваген (Германия) и в дальнейшем к 2008 г. позволил создать первое поколение промышленных мультиагентных систем для управления танкерами (2002), консолидацией грузовых перевозок (2004), корпоративным такси (2006), рекламными баннерами (2008) и т.д.

В 2010 г. началась разработка нового инструментария, на базе которого были реализованы специализированные модели и методы построения ПВ-сетей, включая гибридную комбинацию классических и мультиагентных методов для управления контейнерными грузовыми перевозками — Full Truck Load (FTL) и частично заполненными грузовиками — Less Truck Load (LTL) (2010), движением поездов (2012), фабриками (2014) и проектами (2015), грузопотоком Международной космической станции (2016), мобильными бригадами (2017) и рядом других приложений.

В 2017–2020 гг. был сделан следующий шаг в развитии предлагаемого МАС подхода и предложены новые классы программных агентов для решения сложных задач управления производственными системами, в частности предложены агенты технологического процесса и каждой задачи, введена двухуровневая микроэкономика виртуального рынка, а также разработаны онтологический подход и структура унифицированной мультиагентной системы для настройки на специфику предметной области и бизнеса предприятий.

5. Формализация задачи и метода адаптивного планирования в интеллектуальных системах управления ресурсами предприятий

5.1. Онтологическая модель предприятия для настройки унифицированной мультиагентной системы на специфику предметной области предприятия

В целях настройки унифицированной МАС на специфику предметной области предприятия выделены основные концепты и построена базовая онтология управления ресурсами в форме семантической сети, состоящей из классов понятий и отношений, а также предложена методика построения онтологической модели цифрового двойника (ЦД) предприятия [37]. Общее определение онтологии имеет вид: $O =< C, R, \Phi >$, где C – множество понятий, R – множество атрибутов и отношений (n-местных предикатов), Φ – множество функций семантической обработки (интерпретации), заданных на понятиях и отношениях. Онтологии все чаще используются при создании интеллектуальных систем [38–40], но для управления ресурсами требуемые онтологии не были известны. Для построения онтологических моделей предприятий предлагается использовать базовую онтологию управления ресурсами O_{plan}, в которой на основе анализа различных производственных задач выделены наиболее общие и повторно используемые понятия (табл. 1), в то время как детали, зависящие от предметной области, предложено специфицировать в прикладных онтологиях O_{domain}, расширяющих базовую:

$$O_{domain} \supseteq O_{plan}.$$

Таблица 1. Основные понятия базовой онтологии управления ресурсами

Понятие	Краткое описание
Заказ	Заявка на выпуск продукта, специфицирующая его количество и директивные сроки получения
Продукт	Объект, поступающий на вход или являющийся результатом выполнения задачи
Задача	Групповая или атомарная работа (набор связанных работ), выполнение которых необходимо для получения продукта
Ресурс	Средства производства, необходимые для выполнения задачи

Онтология O_{plan} используется реализованными в МАС классами агентов, которые через функции интерпретации Ф получают возможность взаимодействовать с базой знаний. Часть понятий и отношений из O_{domain} являются производными от базовых понятий и отношений из O_{plan}, что позволяет объяснить МАС, как работать с онтологией предметной области, связав ее понятия и отношения с уже известными и интерпретируемыми системой, обработка которых встроена в ее программный код. При этом O_{domain} может также включать понятия и отношения, не являющиеся производными от базовых, которые будут использоваться МАС при сопоставлении свойств ресурсов и продуктов с требованиями со стороны задач.

В предлагаемой формализованной онтологии управления ресурсами C_{plan} заказы ($Order$) определяют количество и сроки создания продукта ($Product$), задачи ($Task$) задают необходимую последовательность действий для его получения и специфицируют необходимые для своего выполнения ресурсы ($Resource$):

$$C_{plan} = \{Order, Product, Task, Resource\}.$$

Каждый заказ требует появления продукта («создает» – *create*), который, в свою очередь, связан с задачей, в результате выполнения которой он появляется:

$$\forall x \exists y (Order(x) \rightarrow Product(y) \wedge create(x, y)).$$

Продукты могут поступать на вход задачи, а также являться результатом ее выполнения, и в зависимости от роли в технологическом процессе их предложено декомпозировать на «Производимые» (*Produced Product*) и «Потребляемые» (*Consumed Product*). Между задачей и соответствующим видом продукта предложены отношения «производит» (*produce*) и «потребляет» (*consume*):

$$\forall x \exists y (ProducedProduct(x) \rightarrow Product(x) \wedge Task(y) \wedge produce(y, x)),$$
$$\forall x \exists y (ConsumedProduct(x) \rightarrow Product(x) \wedge Task(y) \wedge consume(y, x)).$$

Множество задач предложено разбивать на подмножества: «Групповые» (*Group Task*) и «Атомарные» (*Atomic Task*). Задачи связаны между собой посредством отношений вложенности («является частью» – *part of*) и упорядоченности («следует за» – *follow*). Эти отношения позволяют агенту найти

Таблица 2. Базовые типы задач

Вид задачи		Параметры
Атомарная	Фиксированная длительность	Продолжительность задана фиксированной нормой времени
	Фиксированный объем работы	Продолжительность зависит от состава и характеристик используемых ресурсов и/или объема выпускаемого продукта
	Гамак	Выполняется строго между задачами предшественниками и задачами последователями
Групповая		Продолжительность «покрывает» интервалы выполнения дочерних задач

Таблица 3. Базовые типы ресурсов

Вид ресурса	Параметры
Преобразуемый	Тратится при выполнении задачи (в количестве, определенном ее требованиями), может быть восполнен согласно графику поставок
Обеспечивающий	Становится доступным для повторного использования в прежнем количестве сразу после завершения задач, на которые был выделен. Может иметь график зависимости располагаемого объема от времени.

предыдущую и следующую задачу для запроса о перемещении в расписании или сообщения о возникшей задержке в выполнении:

$$\forall x, y(partOf(x, y) \rightarrow Task(x) \wedge Task(y)),$$
$$\forall x, y(follow(x, y) \rightarrow Task(x) \wedge Task(y)),$$
$$\forall x \exists y(GroupTask(x) \leftrightarrow Task(x) \wedge Task(y) \wedge partOf(x, y)),$$
$$\forall x(AtomicTasks(x) \leftrightarrow \neg\, GroupTasks(x)).$$

В зависимости от способа определения длительности предложено атомарные задачи подразделять на задачи с фиксированной длительностью, с фиксированным объемом работ и на задачи вида «гамак» (табл. 2). Ресурсы обеспечивают выполнение задач, с точки зрения участия в технологическом процессе предложено подразделять их на преобразуемые и обеспечивающие (табл. 3).

Отношение «требует» (*require*) указывает типы ресурсов, необходимые для выполнения задачи. Для выбора вариантов выполнения задачи на различных ресурсах вводится понятие «Требование к ресурсам» (*Resource requirement*):

$$\forall x, y((reuire(x, y) \rightarrow Task(x) \wedge (ResourceRequirenment(y) \vee Resource(y))).$$

Продукты могут требовать размещения (*stored*):

$$\forall x, y((stored(x, y) \rightarrow Product(x) \wedge ReusableResource(y)).$$

Таким образом, на уровне базовой онтологии планирования фиксируется множество отношений R, которые должны поддерживаться МАС:

$$R_{plan} = \{create, consume, produce, partOf, follow, require, stored\}.$$

Для работы с онтологиями управления ресурсами и построения баз знаний предприятий были выделены и реализованы в виде библиотеки средств агентов следующие функции семантической обработки (интерпретации) Ф:

1. $Concepts = \phi_1(c)$ – получить множество всех понятий $Concepts \subseteq C$, являющихся производными от указанного понятия $c \in C$.
2. $Relations = \phi_2(r)$ – получить множество всех отношений $Relations \subseteq R$, являющихся производными от указанного отношения $r \in R$.
3. $Instances = \phi_3(c)$ – получить множество всех экземпляров $Instances$ заданного класса $c \in C$ (включая экземпляры производных классов).
4. $AreRelated = \phi_4(c_1, c_2)$ – проверить, является ли понятие $c_1 \in C$ производным от понятия $c_2 \in C$.
5. $AreRelated = \phi_5(r_1, r_2)$ – проверить, является ли отношение $r_1 \in R$ производным от отношения $r_2 \in R$.
6. $IsPart = \phi_6(i, set)$ – определить, принадлежит ли экземпляр i заданному множеству set, путем сравнения атрибутов и отношений экземпляра с атрибутами и отношениями, определяющими это множество (учитывая возможность замещения базового класса или отношения производным).
7. $Tasks = \phi_7(p)$ – определить множество задач, результатом которых является получение указанного продукта $p \in ProducedProduct$.
8. $Resources = \phi_8(t)$ – определить подходящие для выполнения задачи $t \in Task$ ресурсы.
9. $Products = \phi_9(t)$ – определить подходящие для выполнения задачи $t \in Task$ продукты.

В отличие от базовой онтологии управления ресурсами онтология предметной области может дополняться новыми элементами без необходимости последующего внесения изменений в состав и логику работы агентов. Для работы унифицированной МАС добавляемые новые понятия, атрибуты и отношения должны быть достижимы относительно базовых понятий, атрибутов и отношений при применении к ним функций семантической обработки Ф. Например, на уровне онтологии машиностроения в качестве продуктов рассматриваются детали ($Component$), сборочные единицы ($AssemblyElement$) и изделия ($FinalProduct$), в качестве задач – технологические процессы ($Process$) и операции ($Operation$), ресурсы представлены оборудованием ($Equipment$), оснасткой ($Tool$) и персоналом ($Employee$):

$$\forall x(Product(x) \to Component(x) \vee AssemblyElement(x) \vee FinalProduct(x)),$$
$$\forall x(Task(x) \to Process(x) \vee Operation(x),$$
$$\forall x(Resource(x) \to Equipment(x) \vee Tool(x) \vee Employee(x)).$$

На основе прикладной онтологии строится онтологическая модель предприятия:

$$M = \{O_{domain}(O_{plan}, I\},$$

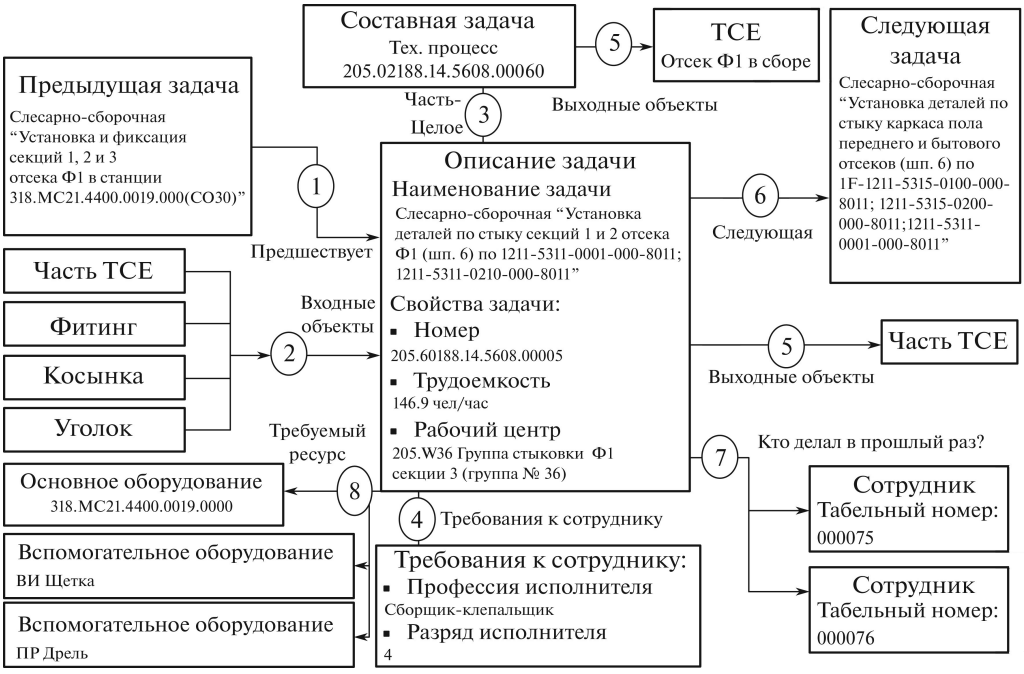

Рис. 1. Пример онтологической спецификации одной операции по сборке изделия (цифрами указаны отношения: 1 и 6 – «предыдущая/последующая задача», 2 и 5 – «вход/выход задачи», 3 – «часть/целое», 4 – «требует», 5 – «связана», 7 – «исполнитель»).

в которую, кроме понятий и отношений базовой онтологии O_{plan} и прикладной онтологии O_{domain}, рассмотренных выше, включаются дополнительные понятия уровня предприятия, а также экземпляры I введенных ранее понятий, например добавляются единицы оборудования с инвентарными номерами, а также рабочие с табельными номерами, навыками и компетенциями.

На основе онтологической модели строится сцена предприятия S, которая описывает состояние предприятия и содержит значения атрибутов всех экземпляров понятий и отношений онтологической модели предприятия для заданного момента времени t : $S = M(t)$. В результате применения предлагаемого подхода расписание работы предприятия представляется в виде семантической сети связанных между собой отношениями понятий (рис. 1), что позволяет не только учитывать специфику каждой задачи, но и использовать «топологию» расписания при принятии решений агентами, например, быстрее определять состав участников переговоров, существенно сокращая перебор вариантов и время вычислений.

Для создания ИСУР предложена методика, позволяющая формализовать знания предметной области в онтологии и на этой основе создавать онтологические модели предприятий. Она состоит в выполнении нижеследующих этапов:

1. Описать номенклатуру используемых и производимых продуктов или сервисов (сырья, полуфабрикатов, готовых изделий, сборочных единиц и др.).
2. Описать состав и структуру производственных ресурсов.
3. Задать технологические процессы получения продуктов, представляющие собой упорядоченный список задач (операций).
4. Определить критерии, предпочтения и ограничения для адаптивного планирования заказов на ресурсы предприятия при возникновении событий рассогласования плана и факта.
5. В качестве входных данных для такой модели могут подаваться перечень заказов, содержащих сведения об изготавливаемом продукте, его количестве и сроках выполнения, а также выделенные события (новый заказ, поломка станка, задержка исполнения задачи и т.д.).

В результате предприятию ставится в соответствие его онтологическая модель, отражающая текущее состояние его заказов и ресурсов, а также планы и показатели работы на любой горизонт времени, ограниченный сроками крайнего заказа.

5.2. Унифицированная мультиагентная система для управления ресурсами

Рассмотрим метод работы унифицированной мультиагентной системы управления ресурсами предприятий в составе ИСУР при переходе из текущего в новое состояние при появлении очередного события.

Состояние ИСУР предприятия S_{Π} предлагается определить композицией состояний $s_{i,i} = \overline{(1,n)}$ объектов, участвующих в производственном процессе (заказов, продуктов, ресурсов и задач):

$$S_{\Pi} = \{s_i\},$$
$$s_i = \{model_i, plan_i, kpi_i\},$$

где: $model_i$ – онтологическая модель объекта; $plan_i$ – план работы объекта, kpi_i – показатели эффективности его работы. Если обозначить как S_{real} состояние реального предприятия, то необходимо, чтобы состояние реального предприятия и состояние его ИСУР в каждый момент времени k максимально совпадали:

$$D\Big(S_{real}^{(k)}, S_{\Pi}^{(k)} \to 0\Big),$$

где D – функция, показывающая степень различия онтологической модели, планов и показателей объектов ИСУР с реальным предприятием. Тогда при появлении нового события $Event^{(k)}$ в реальном предприятии его ИСУР должен максимально быстро перейти в новое состояние за счет переходного процесса по адаптивному перепланированию задач и ресурсов, задетых событием:

$$S_{\Pi}^{(k+1)} = F(S_{\Pi}^{(k)}, Event^{(k)}),$$

Таблица 4. Цели, предпочтения и ограничения основных классов агентов

Тип	Цели и предпочтения	Ограничения
Агент заказа	Быть выполненным с минимальной задержкой (c) и стоимостью (p): $Y_i = w_1\left(1 - \frac{c}{c_{kp}}\right) + w_2\left(1 - \frac{p}{p_{kp}}\right)$	Сроки, объем, предельная стоимость
Агент задачи: – групповой – атомарной	Быть выполненным на подходящем ресурсе в указанные сроки за минимальное время ($\tau_i = finish_i - start_i$): $Y_i = \begin{cases} 1, & \tau_i < \tau_{opt} \\ \frac{\tau_i - \tau_{kp}}{\tau_{opt} - \tau_k p} & \text{иначе,} \end{cases}$	Характеристики требуемых ресурсов и продуктов, сроки начала и окончания, взаимосвязи с другими задачами
Агент ресурса	Быть максимально загруженным, минимизировать простои и переналадки: $Y_i = \begin{cases} 0, & u_i < u_{kp} \\ \frac{u_i - u_{kp}}{u_{opt} - u_k p} & \text{иначе,} \end{cases}$ где u_i – утилизация ресурса i	Календарь работы, интервалы недоступности, правила обслуживания и переналадки, производительность
Агент продукта	Обеспечить свое хранение, минимизировать время между производством и потреблением (ϵ): $Y_i = 1 - \frac{\epsilon_i}{\epsilon_{kp}}$	Требования по хранению, время поставки или производства, время потребления
Агент системы (предприятия в целом)	Выявление «узких мест» в расписании, управление активностью агентов системы, взаимодействие с внешними системами	Время, отводимое на планирование, глубина цепочек перестановок в расписании

где F – функция, адаптивно перестраивающая план работы предприятия в ответ на поступившее событие, которую и должна реализовывать МАС. Для решения поставленной задачи выделен модифицированный набор базовых объектов ПВ-сети и каждому такому объекту s_i поставлен в соответствие программный агент a_i, реализующий заданное для его класса поведение (табл. 4).

Определим цели каждого агента через функцию удовлетворенности $Y_i(Plan_i)$, представляющую собой взвешенную сумму M компонент, соответствующих различным критериям – показателям kpi_i и рассчитываемых на основе текущего плана работы $plan_i$ связанного с агентом объекта:

$$Y_i = \sum_{j=1}^{M} w_{i,j} y_{i,j},$$

где: $y_{i,j}$ – компонент функции удовлетворенности по критерию $j = \overline{(1, M)}$, $w_{i,j}$ – весовой коэффициент, такой что $0 \leqslant w_{i,j} \leqslant 1$ и $\sum_{(j=1)}^{M} w_{i,j} = 1 \; \forall i$.

Для автоматизации принятия решений будем использовать виртуальный рынок ПВ-сети системы, на котором агенты заказов могут покупать время ресурсов, и решать конфликтные ситуации, когда несколько заказов или задач претендуют на использование одного и того же ресурса или продукта, посредством выплаты компенсаций за освобождение слота времени.

В зависимости от достигнутой удовлетворенности агенту начисляется премия (штраф), размер которой определяется через заданную для него функцию бонусов и штрафов: $B_i(Y_i)$. Ожидаемый бонус и текущий бюджет могут быть потрачены агентом на выплату компенсаций агентам, которые согласились на уступки, но чье состояние было ухудшено при изменениях.

Функция удовлетворенности агента связывается с оценкой состояния объекта, а функция бонусов и штрафов – с возможностями агента перестроить расписание для удовлетворения своих интересов. Вид функций выбирается таким образом, чтобы приближение состояния агента к его идеалу показателей kpi_i повышало удовлетворенность и размер бонуса агента.

Агенты ресурсов дополнительно характеризуются функцией стоимости $W_i(plan_i, kpi_i)$, определяющей стоимость размещения задач. Модификация модели ПВ-сети связана с введением неоднородных классов агентов с собственными функциями удовлетворенности и функциями бонусов и штрафов, а модификация метода сопряженных взаимодействий позволяет использовать онтологическую модель предприятия в работе унифицированной МАС, в которой для каждого экземпляра объекта модели будут создаваться собственные экземпляры агентов для анализа ситуации, планирования ресурсов и контроля их использования (рис. 2):

1. В соответствии с текущим состоянием ИСУР $S_{\text{П}}$ создаются экземпляры агентов заказов, ресурсов и продуктов, которые получают разрешение от агента системы начать активность.
2. Агент активного заказа a_k считывает из базы знаний технологический процесс изготовления связанного с ним продукта и порождает агентов задач, соответствующих технологическому процессу и его дочерним операциям, связанных между собой отношениями вложенности и очередности.
3. Агент задачи верхнего уровня проверяет наличие используемых при выполнении задачи продуктов, оценивает требования по ресурсам и подбирает их комбинацию на основе оценки своей продолжительности.
4. Процедура поиска вариантов размещения включает анализ требуемых ресурсов, сопоставление требований задач и возможностей ресурсов, согласование времен доступности всех ресурсов, выбор лучшей комбинации исполнителей на основе метода ветвей и границ.
5. По мере подбора ресурсов определяется множество заказов $\{a_i \mid i \neq k, plan'_k \cap plan_i \neq 0\}$, мешающих размещению на выбранных ресурсах (конфликтное множество), что определяет направленный характер проводимого перебора и существенно сокращает число вариантов.

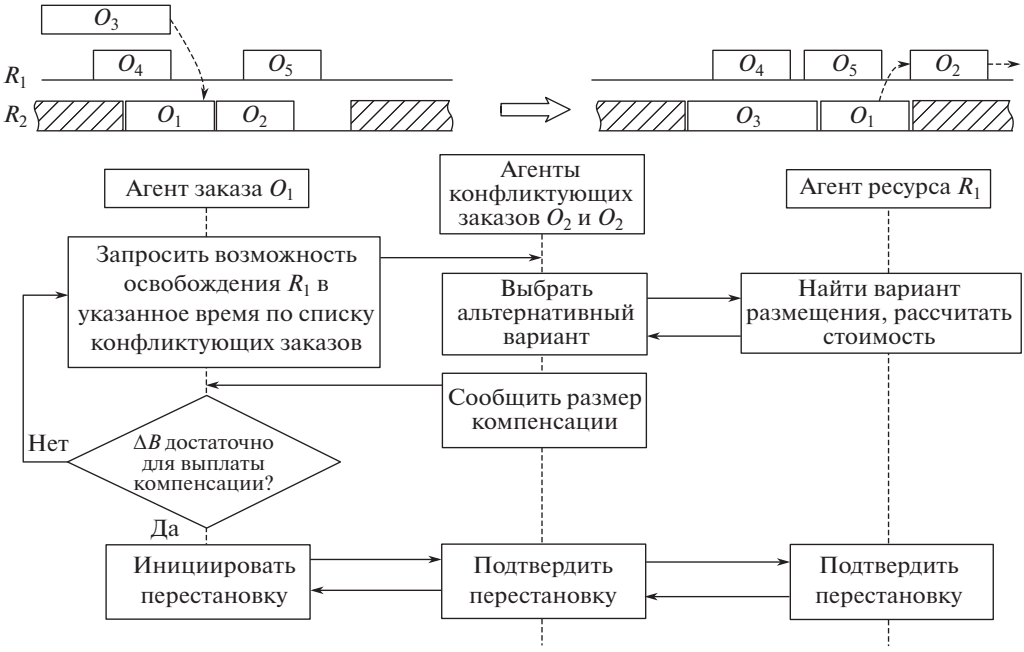

Рис. 2. Фрагмент протокола переговоров по разрешению конфликтов.

6. После выбора варианта размещения агент групповой задачи отправляет запрос на планирование агентам дочерних задач.
7. Агенты дочерних задач рекурсивно проводят поиск вариантов размещения с учетом установленных родительской задачей ограничений. Результаты планирования сообщаются агенту родительской задачи верхнего уровня, который уточняет свое размещение или предлагает задачам найти другое размещение.
8. Агент верхней задачи сообщает агенту заказа о выбранном размещении.
9. Агент заказа предлагает конфликтующим заказам найти себе другое место в расписании, сообщив потери, которые им пришлось понести, по сравнению с базовым (отправным для текущей версии плана) вариантом расписания (рис. 1). В результате определяется цепочка перестановок, рассчитываются потери агентов ΔB_i, которых затронули изменения. Цепочка перестановок успешна, если агент заказа может компенсировать потери конфликтующим заказам благодаря достигаемому приросту функции бонусов и штрафов ΔB_k :

$$\Delta B_k \geqslant \sum_{i \neq k}^{n} \Delta B_i.$$

В этом случае изменение утверждается, иначе – ищутся другие варианты.

10. Агент заказа проверяет наличие связанных с ним отношением «Производится» продуктов и оповещает их агентов о сроках поставки на склад.

11. Процесс завершается, если вышло время, отводимое на построение расписания, или достигнуто условие **«конкурентного равновесия» (консенсуса)**, которое состоит в том, что для любого агента a_k больше не находится такого изменения плана работы $plan'_k$, которое привело бы к приросту удовлетворенности ΔY_k и, как следствие, увеличению значения функции бонусов и штрафов ΔB_k, что смогло бы компенсировать суммарные потери остальных агентов a_i, затронутых этим изменением и нашедших другой вариант размещения $plan'_i$, минимизирующий их потери и согласующийся с ранее принятыми изменениями:

$$\Delta B_k + \sum_{i \neq k}^{n} \Delta B_i < 0 \ \forall k.$$

12. По достижению консенсуса ИСУР приостанавливает свою работу, выдает построенный новый план исполнителям и переходит в режим ожидания новых событий.

Существенным фактором, снижающим вычислительную сложность алгоритма, является использование направленного поиска вариантов перестановок при адаптивной перестройке расписания, при котором взаимодействие происходит в основном между конфликтующими заказами.

6. Инструментальный комплекс для создания ИСУР

Для автоматизации процессов построения ИСУР разработаны функции и архитектура комплекса инструментальных средств. В состав комплекса включен конструктор онтологий (КО) и баз знаний (КБЗ) предприятия (включает как классы, так и экземпляры понятий), конструктор сцен, унифицированная МАС управления ресурсами в реальном времени, очередь сообщений и средства взаимодействия с пользователем (рис. 3).

Основное назначение КО – построение, редактирование и хранение базовой и прикладных онтологий, а также предоставление программного доступа к этим данным. КБЗ предназначен для формирования онтологических моделей предприятий на основе онтологии выбранной предметной области.

Унифицированная МАС управления ресурсами обеспечивает создание и настройку виртуального мира агентов под заданную онтологическую модель каждого предприятия. На основе онтологической модели в виртуальном мире ИСУР предприятия создаются экземпляры классов агентов для каждой сущности конкретного предприятия (заказа, станка, технологии, изделия, сотрудника и т.д.) и обрабатываются события, поступающие через очередь событий из мира реального предприятия. Основными функциями унифицированной МАС являются адаптивное перепланирование расписания выполнения заказов с достижением нового консенсуса агентов по каждому событию, а также последующий мониторинг и контроль их исполнения. Исходные данные, результаты перепланирования и показатели работы МАС размещаются в сцене

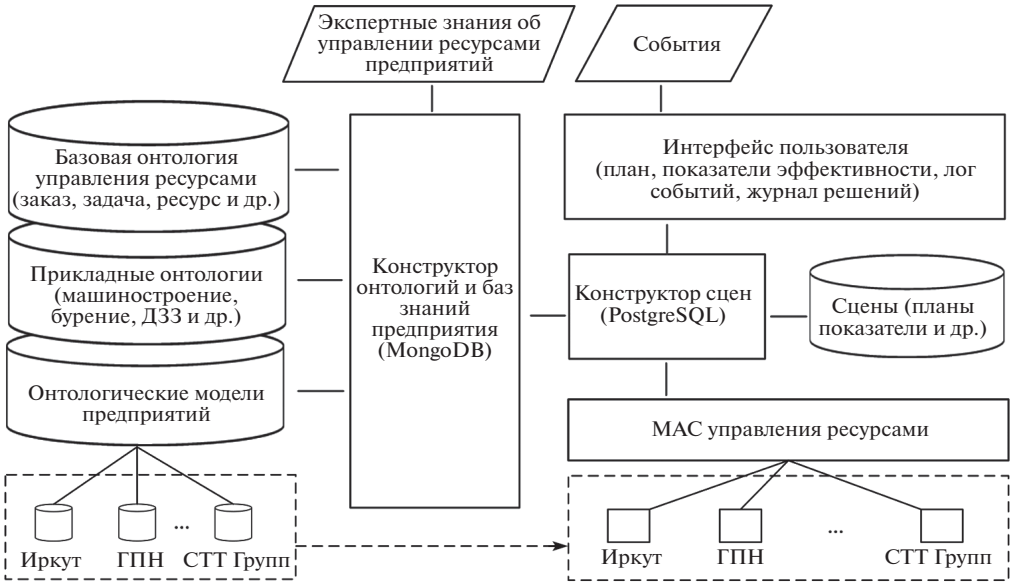

Рис. 3. Структура инструментального комплекса создания ИСУР.

мира агентов, которая содержит массивы данных истории, текущего состояния и планов работы ИСУР, показатели эффективности.

Веб-интерфейс пользователя включает в себя набор визуальных компонент для задания исходных параметров и визуализации результатов.

При загрузке онтологической модели предприятия создается виртуальный мир агентов ИСУР, содержащий экземпляры основных агентов, и сцену (набор массивов) данных, отражающую параметры состояния объектов предприятия.

В любой момент времени ИСУР предприятия может быть скопирована в отдельную версию для перехода к режиму моделирования «Что будет, если...?», чтобы предсказать реакцию ИСУР на события, которые ожидаются, но пока еще не случились (приход нового крупного заказа, ввод в действие новых ресурсов, смена технологических процессов и т.д.).

Разработанный комплекс был использован для решения прикладных задач управления ресурсами:

— агрегатно-сборочного производства самолетов МС-21 для ОАО «Иркут»;

— сборочного производства грузовых электромобилей для ООО «ТРА»;

— бурения нефтедобывающих скважин для ООО «Газпромнефть-Ямал»;

— выращивания посевов растений для ОАО «Рассвет»;

— целевого применения группировки КА ДЗЗ для «СТТ-Групп».

Исследование применения разработанного комплекса показало существенное снижение сложности и трудоемкости разработки ИСУР. При этом для каждого созданного ИСУР оценивался объем изменений, вносимых в базовую

Таблица 5. Результаты применения комплекса для создания ИСУР предприятий

Задача	Число классов понятий и отношений			Кол-во агентов	Время на разработку (чел/м)	
	O_{plan}	O_{domain}	M		БЗ	МАС
Сборка самолетов	61	152	925	>350	3	3.5
Сборка грузовиков	61	89	382	>520	1	2
Бурение скважин	61	89	382	>520	1	2
ЦД посевов	61	42	236	>100	1	1
Группировка КА	61	112	304	> 450	1	4

онтологию, а также объем доработок основных классов агентов под задачу (табл. 5).

Представленные данные показывают, что базовая онтология управления ресурсами O_{plan} оказалась построена из примерно 60 классов основных понятий и отношений. Прикладные онтологии расширяют состав понятий и отношений примерно в 2–3 раза. Онтологические модели предприятия, включающие экземпляры, имеют от 236 до 925 экземпляров. Именно эти модели и загружаются в ИСУР, позволяя автоматически создать требуемое число агентов. Трудоемкость доработки унифицированной ИСУР для каждого из указанных применений составила в среднем 2–3 мес, что по экспертным оценкам позволяет в 3–4 раза сократить сроки и стоимость создания ИСУР в сравнении с традиционным подходом.

7. Пример построения ИСУР для решения прикладной задачи

7.1. Постановка задачи для управления спутниковой группировкой ДЗЗ

Рассмотрим применение разработанного подхода для создания ИСУР управления многоспутниковой группировки космических аппаратов (КА) для дистанционного зондирования Земли (ДЗЗ). В основе управления группировкой КА лежит задача адаптивного планирования полетных операций по заказам на съемку объектов ДЗЗ, поступающим в заранее не известные моменты времени, или по событиям неисправностей КА или каналов связи, также возникающим в заранее не известные моменты времени. Пусть задана упрощенная модель космической системы (КС) для решения задач ДЗЗ, представляющая собой совокупность двух сегментов: космического комплекса и наземного комплекса. Космический комплекс выполняет функции получения и передачи информации, наземный комплекс – функции приема-передачи и обработки передаваемой информации. Космический комплекс состоит из множества КА $S = s_i, i = \overline{1, L}$. Каждый КА s_i характеризуется набором элементов орбиты и параметрами установленного на нем бортового оборудования. Наземный комплекс представлен множеством наземных станций приема информации

(НСПИ) $G_R = \{g_r\}, r = \overline{1,R}$. Каждая станция g_r характеризуется географическим местоположением и параметрами установленной антенны. Для НСПИ и КА могут быть указаны ограничения в виде графика работы и интервалов недоступности. КС должна обеспечить выполнение множества заявок на съемку точечных и площадных объектов наблюдения (ОН) $O = \{o_p\}$, $p = \overline{1,P}$. Для заявки на съемку o_p может быть указан ее приоритет pr_p (заявка с низким приоритетом не должна мешать оптимальному расположению более высокоприоритетной заявки) и множество ограничений: момент времени, до которого необходимо получить снимки t_p^end, коэффициент баланса между оперативностью и качеством получаемой информации c_p (задается в диапазоне от 0 до 1), минимальное и желаемое линейное разрешение полученного снимка $\min Rp$ и $\max Rp$. В рассматриваемой модели КС КА выполняет две операции:

— съемка некоторой области $sa_j shoot_j$, характеризующаяся интервалом выполнения $t_j^{shoot} = [t_j^{shootStart}; t_j^{shootEnd}]$ и углом крена КА $sAngle_j$;

— проведение сеанса связи КА с НСПИ с целью передачи полученных данных на Землю $drop_j$, характеризующегося интервалом выполнения $t_j^{drop} = [t_j^{dropStart}; t_j^{dropEnd}]$.

НСПИ в свою очередь выполняет одну операцию – получение данных с КА $receiv_j$, характеризующуюся интервалом выполнения $t_j^{recive} = [t_j^{reciveStart}, t_j^{reciveEnd}]$.

Для реализации космической съемки ДЗЗ на основе заявок, поступающих от потребителей, требуется сформировать комплексный план выполнения операций на заданный горизонт планирования, составленный в соответствии с критерием минимизации времени доставки снимков потребителям, а также максимизации их качества. Таким образом, целевая функция системы имеет вид: $t_j^{receiv} = [t_j^{receivStart}; t_j^{receivEnd}]$.

$$OF = \frac{1}{M}\sum_{k=1}^{N} OF_k \to \max,$$

$$OF_k = c_k F_1^k + (1 - c_k)F_2^k \to \max,$$

$$F_1^k = \frac{t_k^{end} - t_k^{dropEnd}}{t_k end - t_k^{start}},$$

$$F_2^k = \begin{cases} \dfrac{\min R_k - r_k}{\min R_k - \max R_k}, & \text{если } r_k \geqslant \max R_k \\ \dfrac{r_k}{\max R_k} & \text{иначе,} \end{cases}$$

где

OF – целевая функция системы,

OF_k – целевая функция k-й задачи,

N – количество запланированных съемок,

F_1^k – оценка критерия оперативности получения информации для k-й задачи,

F_2^k – оценка критерия качества полученного снимка для k-й задачи,

t_k^{start}, t_k^{end} – горизонт планирования для k-й задачи,

r_k – фактическое линейное разрешение полученного снимка для k-й задачи.

Примем, что на полученное решение накладывается ряд ограничений:

— выполнение условия наблюдаемости между КА и ОН при съемке;

— наличие радиовидимости между КА и НСПИ при передаче результатов съемки;

— наличие свободного места в бортовом запоминающем устройстве (ЗУ) КА;

— выполнение условия приоритезации заявок;

— согласованность последовательности моментов времени проведения операций;

— КА и НСПИ могут одновременно выполнять не более одной операции.

7.2. Функциональные возможности прототипа ИСУР для решения задач ДЗЗ

Разработанный прототип системы предназначен для составления и адаптивного перестроения локально-оптимального плана выполнения задач по съемке точечных и площадных районов средствами группировки КА при заданных критериях эффективности и технических характеристиках КА и НСПИ, принятой модели обстановки и внешней среды, а также для моделирования КС при изменении ее состава и конфигурации.

Система обеспечивает следующие основные функции:

— загрузка исходных данных о составе и параметрах элементов КС;

— загрузка заявок на съемку точечных и площадных ОН;

— составление локально-оптимального плана работы КС (группировки КА и НСПИ) при заданном составе технических характеристик;

— адаптивное перестроение локально-оптимального плана работы КС по событиям, изменяющим характеристики КС (состав КА и НСПИ, технические характеристики систем КА и НСПИ, добавление ограничений работы КА и НСПИ, изменение критериев планирования), изменяющим исходные данные по съемке точечных и площадных районов;

— визуальное моделирование процесса выполнения заявок ДЗЗ, приема и передачи данных на наземные станции;

— формирование графиков и диаграмм, отображающих результаты планирования;

— выгрузка полученного локально-оптимального плана работы КС.

Система создается для эксплуатации в профильных подразделениях Центра управления полетами (ЦУП) и обеспечения поддержки принятия решений по планированию использования группировки КА.

7.3. Экспериментальные исследования ИСУР для решения задач ДЗЗ

Для проведения экспериментальных исследований и оценки степени пригодности разработанной системы к решению задачи управления группировками КА в реальном времени использовалась модель КС, в состав космического комплекса которой входит группировка из 30 идентичных КА, а наземный комплекс системы представлен сетью из 10 НСПИ. Моделировался случайный поток заявок на съемку объектов ДЗЗ, и результаты планирования подвергались автоматизированной обработке с участием экспертов. Эксперименты проводились на персональном компьютере с центральным процессором Intel Core i7-3770 (4 ядра/8 потоков, 3.4ГГц) и оперативным запоминающим устройством 8Гб под управлением операционной системы Windows 10.

7.3.1. Анализ хода и результатов планирования в ИСУР для ДЗЗ

В данном исследовании производился сбор статистической информации с целью оценки качества полученного расписания и анализа хода его построения. На ее основе построены ряд графиков и гистограмм. На рис. 4 изображены графики с историей изменения значения текущей и предельной целевой функции системы с момента начала планирования, при помощи которых можно оценить отклонение текущего значения целевой функции (ЦФ) системы от ее максимально возможного значения:

$$limOF = \frac{1}{M}\sum_{k=1}^{N} OF_k(i\dot{w}o_k),$$

где

$limOF$ – максимально возможная ЦФ системы, M – общее число задач, N – число рассмотренных задач, $(i\dot{w}o_k)$ – вариант размещения, расположенный в точке глобального оптимума ЦФ k-й задачи.

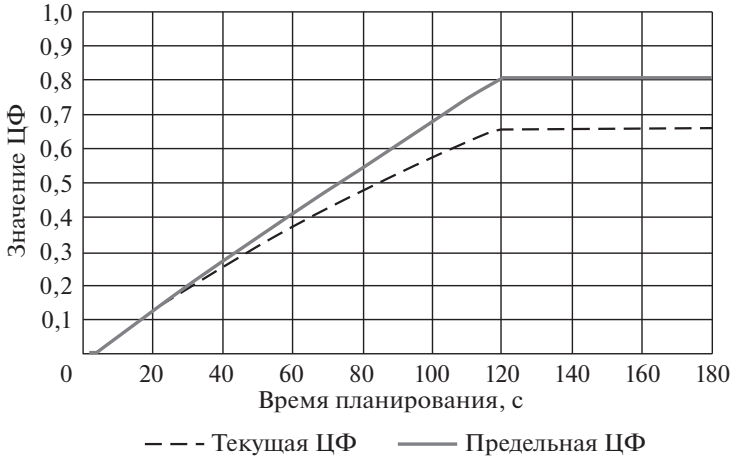

Рис. 4. Графики изменения значения текущей и предельной ЦФ в ходе планирования.

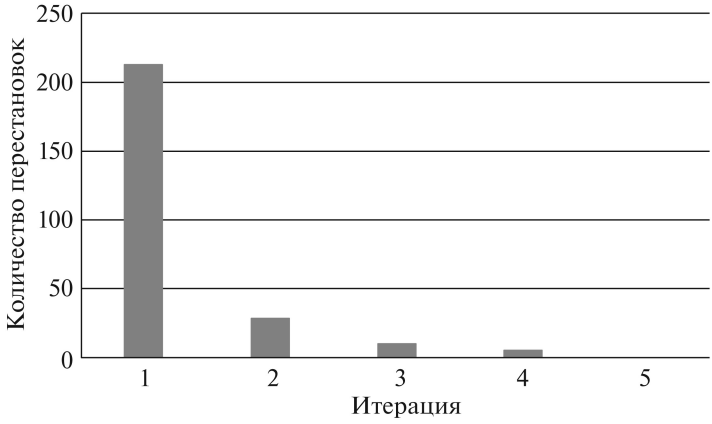

Рис. 5. Число перестановок задач на каждой итерации проактивного планирования.

Рис. 6. Распределение числа вариантов размещения задач лучше текущего, на котором она запланированы.

На рис. 5 изображена диаграмма распределения числа вариантов размещения задач лучше текущего, на котором они запланированы, т.е. вариантов размещения, найденных в ходе планирования, для которых ЦФ задачи больше ее текущего значения, но на которых задача не смогла разместиться в результате конфликтов с другими задачами. Из анализа данной диаграммы видно, что около 1200 задач запланированы на наиболее подходящем для них ресурсе, а более половины оставшейся части задач расположены в 30% лучших вариантов, что говорит об относительно «хорошем» качестве построенного плана.

Столбиковая диаграмма распределения числа перестановок задач на каждой итерации проактивного планирования (рис. 6) демонстрирует его быструю сходимость, так, число перестановок уже на второй итерации планирования меньше числа перестановок на первой итерации в 7 раз.

7.3.2. Исследование адаптивности получаемого в ИСУР расписания ДЗЗ

В данном исследовании оценивалась способность метода к адаптации расписания по «разрушающим» событиям, в частности по выходу из строя одного из КА.

В качестве исследуемого параметра здесь рассматривалось время, затраченное на перепланирование, и качество полученного расписания. Для этого была проведена серия из 10 экспериментов, в ходе которых вначале планировалось выполнение 3000 заявок на съемку ОН, сгенерированных случайным образом по равномерному закону распределения. После того как все заявки были успешно размещены в расписании, из системы исключался один из КА и измерялись время, затраченное на перестроение расписания, изменения значения ЦФ системы и количество запланированных заявок. Результаты экспериментов представлены в табл. 6.

Результаты эксперимента показывают, что выход из строя одного из КА привел к резкому падению значения ЦФ системы в среднем на 0,1 и необходимости поиска новых вариантов размещения для 409 заявок. В ходе перестроения расписания на другие КА было перепланировано 403 заявки, что составляет 98% от числа заявок, которые были запланированы на удаленный КА. В результате восстановления расписания значение ЦФ повысилось до 0,69, что меньше исходного всего на 0,04.

Среднее время перепланирования при этом составило около 9 с.

Таким образом, применение мультиагентного подхода при планировании позволяет оперативно, гибко и эффективно парировать возникновение внешних событий, приводящих к изменению условий решаемой задачи.

Таблица 6. Результаты экспериментов по исследованию способности системы к адаптации расписания

№	Время перепланирования, с	После выхода из строя КА		После перестроения расписания	
		Кол-во распланированных заявок	ЦФ	Кол-во перепланированных заявок	ЦФ
1	9	422	–0,11	418	0,07
2	8	379	–0,10	371	0,04
3	10	468	–0,11	465	0,08
4	11	411	–0,10	406	0,07
5	9	407	–0,11	396	0,06
6	10	425	–0,11	422	0,09
7	7	397	–0,11	395	0,06
8	8	388	-0,10	376	0,05
9	8	377	–0,07	372	0,05
10	9	419	–0,10	417	0,06

7.3.3. Анализ эффективности в сравнении с алгоритмами планирования, основанными на традиционных методах оптимизации

В данном исследовании проводился анализ эффективности разработанного метода в сравнении с алгоритмами планирования, основанными на традиционных методах оптимизации, таких как алгоритм имитации отжига, алгоритм Late Acceptance Hill Climbing и алгоритм Tabu Search, по качеству полученного расписания и временным затратам, необходимым на его составление.

В ходе исследования проведена серия экспериментов, в которых количество заявок на съемку ОН изменялось от 100 до 20 000. При этом измерялось время, затраченное на составление плана, и значение ЦФ системы.

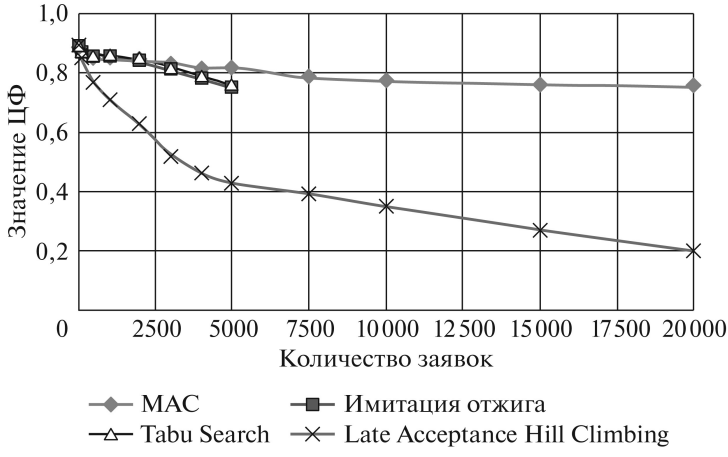

Рис. 7. Число перестановок задач на каждой итерации проактивного планирования.

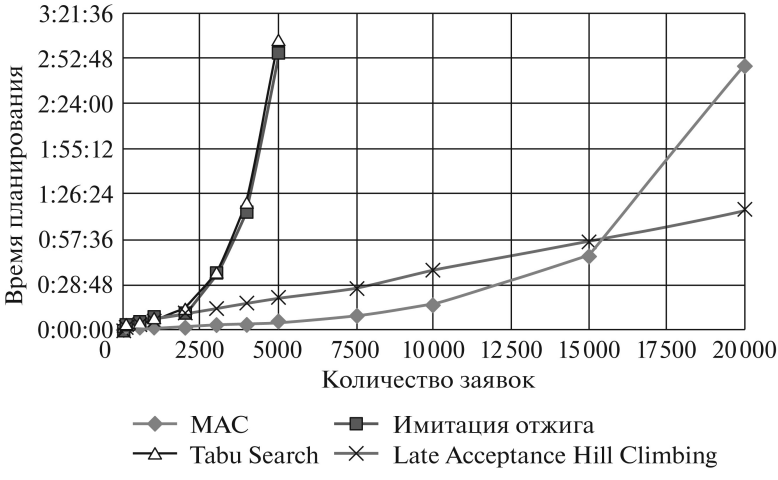

Рис. 8. График зависимости времени планирования от количества заявок.

По результатам проведенных экспериментов построены графики зависимости значения ЦФ системы (рис. 7) и времени планирования (рис. 8) от количества заявок для различных алгоритмов планирования. Для алгоритма имитации отжига и алгоритма Tabu Search результаты получены лишь до 5000 заявок на съемку, так как далее наблюдался экспоненциальный рост времени их работы и потребляемых ресурсов.

Результаты экспериментов показывают, что в данном случае уже на задачах малой размерности ДЗЗ для группировки КА предлагаемый мультиагентный метод не уступает традиционным эвристическим алгоритмам, а с ростом числа заявок демонстрирует более высокую скорость построения расписания без потери в качестве, продолжая работать там, где классические методы уже работают плохо или вовсе не работают.

8. Выводы по результатам

При разработке любых новых методов и средств управления ресурсами традиционно возникает ряд вопросов: “А чем отличается метод? При каких условиях следует его применять? Как новый метод дополняет существующие решения? Что с чем сравнивать и при каких условиях? Какой выигрыш получается в результате и т.д.?”

Ответы на эти вопросы не всегда очевидны и в ходе проведенных исследований по оценке качества и эффективности работы ИСУР по сравнению с классическими и эвристическими системами был выявлен ряд особенностей, которые могут рассматриваться как довольно характерные для разработки и внедрения всего класса рассматриваемых интеллектуальных систем принятия решений в реальном времени:

— Оценка качества и эффективности результатов работы традиционных методов пакетного планирования и оптимизации ресурсов, когда все заказы и ресурсы известны заранее и не меняются в ходе вычислений, в сравнении с адаптивными методами ИСУР, работающими по событиям в реальном времени. Напрямую сделать такое сравнение часто затруднительно, так как традиционное решение оптимизирует одну целевую функцию, а предлагаемый подход в ИСУР позволяет искать баланс интересов между многими функциями. В этих целях критерии принятия решений, предпочтения и ограничения всех участников должны быть сопоставимы, но в ИСУР много больше число участников, принимающих решения, которые при этом еще и должны быть согласованы. В любом случае пользователям важно понимать, насколько далеко получаемое решение ИСУР от глобального «оптимума», предлагаемого имеющимися методами как «лучшее решение». Но главная проблема состоит в том, что для адаптивных методов качество и эффективность решения зависят от самого момента времени получения события и той ситуации, которая сложилась на этот момент в группировке КА, цехе машиностроительного предприятия, парке грузовиков и т.д. В следующий момент ситуация может измениться и может быть востребовано уже другое решение, а решение, выработанное в этом моменте, может уже в следующую минуту быть не лучшим

и даже может не быть применимым вовсе, что требуется учитывать в оценке и выводах.

— Учет семантики предметной области в ИСУР выражается в возможности вводить «на лету» (без перепрограммирования) новые понятия и отношения предметной области, в терминах которых формулируются критерии, предпочтения и ограничения всех участников; при сравнении различных методов целесообразно показать, какова будет трудоемкость встраивания в систему новых критериев, предпочтений и ограничений предметной области. Учет семантики предметной области позволяет на практике сразу отсекать лишние варианты и повышать «разумность» получаемых решений для конечных пользователей.

— Кроме качества и эффективности решения, для такого рода систем важно сравнивать возможности масштабирования решения с ростом размерности решаемой задачи. Здесь важно не только число обрабатываемых заказов, но сама возможность привлечения дополнительного числа вычислительных ресурсов и использования возможностей естественного распараллеливания задачи на современных серверах.

— При сравнении и выборе методов важно показывать, что предлагаемые адаптивные методы виртуального рынка, где переговоры как аукционы идут параллельно и асинхронно, позволяют формировать решение итерационно, причем можно управлять процессом вычислений, включая качество и скорость вычислений, используя механизмы «направляемой самоорганизации» («управляемого хаоса»), в зависимости от желаемых показателей, наиболее приоритетных событий или заданного интервала времени, отведенного на принятие решений.

— Благодаря переговорам, в ходе которых выявляются причины отказов агентов, предлагаемые ИСУР потенциально могут быть способны к «преодолению ограничений» в случае, когда решение не находится в заданных условиях, а также для выработки рекомендаций получения допустимого рационального результата в условиях даже полного отсутствия решения.

— Важнейшим аспектом сравнения обсуждаемых методов является трудоемкость разработки и развития ИСУР, понятность и удобство работы с создаваемыми моделями и методами как для опытных разработчиков методов планирования и оптимизации, так и, в особенности, для новичков-программистов, с целью оперативной доработки системы под специфические и постоянно изменяющиеся условия решения задачи или новые применения.

— Работа методов должна анализироваться и сравниваться в условиях растущей сложности, неопределенности и динамики изменений в среде, связываемой с постоянным желанием пользователей внести все большее число факторов в принятие решений.

— Несомненным преимуществом методов, использующих онтологии предметной области, является устойчивость к ошибкам в наборах исходных данных, так как в реальном времени у пользователей нет времени на анализ таких ситуаций.

— Важно обеспечивать пользователям объяснение найденных решений и возможность удобной интерактивной доработки расписаний совместно с пользователем, изменения им любых частей расписания по своему усмотрению, добавления новых, еще не формализованных, факторов.

Указанные особенности должны обеспечивать более рациональный выбор современными предприятиями применяемых методов и средств для управления ресурсами и разработку комплексных подходов к оценке эффективности внедрений систем искусственного интеллекта.

9. Заключение

До настоящего времени не предложено единого подхода к созданию ИСУР для решения сложных NP полных задач многокритериального адаптивного управления ресурсами в реальном времени.

Разработанная методология построения ИСУР на основе мультиагентной технологии может рассматриваться как один из возможных практических подходов к решению такого рода задач, который не дает глобального оптимума, но позволяет на общих принципах находить приближенные локально-оптимальные решения в приемлемые сроки для повышения эффективности ресурсов предприятий и создавать такие системы в относительно короткие сроки.

Областью применения разработанной технологии в ИСУР являются сложные задачи управления ресурсами, где требуется перестраивать связанные между собой планы работы многих исполнителей по различным событиям, возникающим в заранее не известные моменты времени, причем «здесь и сейчас», в реальном времени. При отсутствии неопределенности и динамики и наличии заранее известных и неизменяющихся потоков заказов и пулов ресурсов предлагаемые методы и средства могут комбинироваться с классическими и эвристическими подходами, что позволяет создавать гибридные ИСУР.

Повышение качества решения поставленных сложных задач в ИСУР по сравнению с традиционными методами связывается с переходом от поиска глобального оптимума одной целевой функции системы к поиску баланса интересов всех участников бизнеса в каждой конкретной ситуации, причем как людей (заказчиков, исполнителей и партнеров), так и любых физических или абстрактных сущностей, также наделяемых своими интересами (предприятие в целом, грузовик, заказ, станок или другое оборудование и т.д.).

Эффективность решения поставленной задачи обеспечивается переходом при создании ИСУР от последовательного комбинаторного перебора вариантов к параллельным и асинхронным переговорам агентов всех участников с собственными целевыми функциями, что позволяет агентам взаимно сужать размерность поиска вариантов решений непосредственно в ходе разрешения конфликтов, а решению сложной задачи – фактически, самоорганизовываться в ходе последовательных улучшений.

Предложенные модели, методы и средства создания ИСУР адаптивного управления ресурсами в реальном времени могут стать основой открытой инструментальной платформы для решения задач указанного класса с целью сокращения трудоемкости, сроков, стоимости разработки и эксплуатации такого рода систем.

Перспективы развития направления связываются с созданием ИСУР как интеллектуальных киберфизических систем, а также построением цифровых экосистем ИСУР масштаба крупного предприятия и даже отрасли для поддержки цепочек кооперации предприятий при решении сложных производственных задач.

СПИСОК ЛИТЕРАТУРЫ

1. *Таха Х.* Введение в исследование операций. М.: Изд. дом «Вильямс», 2005.
2. *Новиков Д.А.* Классификации систем управления // Проблемы управления. 2019. № 4. С. 27–42.
3. *Лазарев А.А., Гафаров Е.Р.* Теория расписаний. Задачи и алгоритмы. М.: Изд-во МГУ, 2011.
4. Handbook of Scheduling: Algorithms, Models and Performance Analysis / Leung J. (Ed.). Chapman & Hall, 2004.
5. *Vos S.* Meta-heuristics: The State of the Art // Local Search for Planning and Scheduling / Nareyek A. (Ed.). Germany: Springer-Verlag. 2001.
6. *Pinedo M.* Scheduling Theory, Algorithms, and Systems. Springer. 2008.
7. *Скобелев П.О.* Открытые мультиагентные системы для оперативной обработки информации в процессах принятия решений // Автометрия. 2002. № 6. С. 45–61.
8. *Виттих В.А., Скобелев П.О.* Мультиагентные модели взаимодействия для построения сетей потребностей и возможностей в открытых системах // АиТ. 2003. № 1. С. 177–185.

   Vittikh V.A., Skobelev P.O. Multiagent Interaction Models for Constructing the Needs-and-Means Networks in Open Systems // Autom. Remote Control. 2003. V. 64. No. 1. P. 162–169. https://doi.org/10.1023/A.1021836811441
9. *Виттих В.А., Скобелев П.О.* Метод сопряженных взаимодействий для управления распределением ресурсов в реальном масштабе времени // Автометрия. 2009. № 2. С. 78–87.
10. *Скобелев П.О.* Мультиагентные технологии в промышленных применениях: к 20-летию основания Самарской научной школы мультиагентных систем // Мехатроника, автоматизация, управление. 2010. № 12. С. 33–46.
11. *Skobelev P.* Bio-Inspired Multi-Agent Technology for Industrial Applications. Multi-Agent Systems – Modeling, Control, Programming, Simulations and Applications / Faisal Alkhateeb F., et al. (Eds.). Austria: InTech Publishers, 2011. P. 495–522.
12. *Rzevski G., Skobelev P.* Managing Complexity. London-Boston: WIT Press, 2014.
13. *Skobelev P.* Towards Autonomous AI Systems for Resource Management: Applications in Industry and Lessons Learned // Y. Demazeau et al. (Eds.). Proceedings of the 16th International Conference on Practical Applications of Agents and Multiagent Systems (PAAMS 2018). 2018. LNAI 10978. P. 12–25.

    https://doi.org/10.1007/978-3-319-94580-4_2

14. *Мескон М., Альберт М., Хедоури Ф.* Основы Менеджмента / пер. с англ. М.: Дело, 1997.

15. *Деминг Э.* Менеджмент нового времени: Простые механизмы, ведущие к росту, инновациям и доминированию на рынке / пер. с англ. М.: Альпина Паблишер. 2019.

16. *Merdan M., Moser T., Sunindyo W., Biffl S., Vrba P.* Workflow scheduling using multi-agent systems in a dynamically changing environment // J. Simulation. 2012. Vol. 7(3). P. 144–158.

17. *Deng L., Lin Y., Chen M.* Hybrid ant colony optimization for the resource-constrained project scheduling problem // J. Syst. Engineer. Electron. 2010. Vol. 21. Issue 1. P. 67–71.

18. *Jaberi M.* Resource Constrained Project Scheduling Using Mean Field Annealing Neural Networks // Int. J. Multidisciplin. Sci. Engineer. 2011. Vol. 2. No. 7. P. 6–12.

19. *Zhang H., Xu H., Peng W.* A Genetic Algorithm for Solving RCPSP // International Symposium on Computer Science and Computational Technology (ISCSCT '08). 2008. Vol. 2. P. 246–249.

20. *Бурков В.Н., Новиков Д.А.* Теория активных систем: состояние и перспектива. М.: Синтег, 1999.

21. *Ivanyuk V., Abdikeyev N., Patshenko F., Grineva N.* Network-Centric Methods Management // Management Science. M.: Financial University under the Government of Russian Federation. 2017. Vol. 7(1). P. 26–34.

22. *Chigani A., Jamed D., Bohner S.* Architecting Network-Centric Software Systems: A Style-Based Beginning // 31st IEEE Software Engineering Workshop (SEW '07). P. 290–299. https://doi.org/10.1109/SEW.2007.95

23. *Wooldridge M., Jennings N.* Intelligent agents: Theory and practice // The Knowledge Engineering Review. 1995. Vol. 10(2). P. 115–152.

24. *Городецкий В.И, Грушинский М.С., Хабалов А.В.* Многоагентные системы // Новости искусственного интеллекта. 1998. № 2. С. 64–116.

25. *Пантелеев М.Г., Пузанков Д.В.* Интеллектуальные агенты и многоагентные системы. СПб.: Изд-во СПбГЭТУ «ЛЭТИ», 2016.

26. *Pearce J., Tambe M., Maheswaran R.* Solving multiagent networks using distributed constraint optimization // AI Magazine. 2008. 29(3).

    Available from: http://teamcore.usc.edu/papers/2008/K-optimality

27. *Modi P.J., Shen W., Tambe M., Yokoo M.* ADOPT: Asynchronous distributed constraint optimization with quality guarantees // Artificial Intelligence Journal. 2005. 161 (1–2). P. 149–180.

28. *Petcu A.* A class of Algorithms for Distributed Constraint Optimizations. Frontiers in Artificial Intelligence and Applications. IOS Press. 2009. 194(1).

    https://doi.org/10.5075/epfl-thesis-3942

29. Evolutionary Multiobjective Optimization Theoretical Advances and Applications. Abraham A., Jain L. C., Goldberg R. (Eds.). Springer. 2005.

    https://doi.org/10.1007/1-84628-137-7

30. *Brussel H., Wyns J., Valckenaers P., et al.* Reference Architecture for Holonic Manufacturing Systems: PROSA // Computers in Industry. 1998. No. 37. P. 255–274.

31. *Sandholm T., Lesser V.* Issues in Automated Negotiation and Electronic Commerce: Extending the Contract Net Framework – In Proc. of the First International Conference on Multiagent Systems (ICMAS-95). 1995. P. 328–335.

32. *Sandholm T., Lesser V.* Leveled-commitment contracting: a backtracking instrument for multiagent system // AI Magazine. 2002. Vol. 23(3). P. 89–100.

33. *Shoham Y., Leyton-Brown K.* Multi-agent systems: Algorithmic, Game Theoretic and Logical Foundations. Cambridge University Press. 2009. Available from: http://www.masfoundations.org

34. *Easley D., Kleinberg J.* Networks, Crowds, and Markets: Reasoning about a Highly Connected World. Cambridge University Press. 2010. Available from: http://www.cs.cornell.edu/home/kleinber/networks-book/

35. *Amelina N., Fradkov A., Jiang Y., Vergados D.* Approximate Consensus in Stochastic Networks With Application to Load Balancing // IEEE Transactions On Information Theory. 2015. Vol. 61. No. 4. P. 1739–1752.

36. *Ivanskiy Y., Amelina N., Granichin O., et al.* Optimal Step-Size of a Local Voting Protocol for Differentiated Consensuses Achievement in a Stochastic Network with Cost Constraints // IEEE Conference on Control Applications (CCA). 2015. P. 1367–1372.

37. *Жиляев А.А.* Онтологии как инструмент создания открытых мультиагентных систем управления ресурсами // Онтологии проектирования. 2019. Т. 9. № 2(32). С. 261–281.

38. *Minhas S.* Ontology Based Environmental Knowledge Management – A System to Support Decisions in Manufacturing Planning // 6th International Conference on Knowledge Engineering and Ontology Development. 2014. P. 397–404.

39. *Järvenpää E., Siltala N., Hylli O., Lanz M.* The development of an ontology for describing the capabilities of manufacturing resources // Journal of Intelligent Manufacturing. 2019. Vol. 30. P. 959–978.

40. *Хорошевский В.Ф.* Проектирование систем программного обеспечения под управлением онтологий: модели, методы, реализации // Онтология проектирования. 2019. Т. 9. № 4. С. 429–448.

41. *Rojko A.* Industry 4.0 Concept: Background and Overview // International Journal of Interactive Mobile Technologies. 2017. Vol. 11(5). P. 77–90.

42. *Leitao P., Karnouskos S., Ribeiro L., et al.* Smart agents in industrial cyber-physical systems // Proc. IEEE. 2016. Vol. 104(5). P. 1086–1101.

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

Поступила в редакцию 27.10.2020

После доработки 15.05.2021

Принята к публикации 30.06.2021

© 2021 г. Ю.А. ДОРОФЕЮК, канд. техн. наук (dorofeyuk_julia@mail.ru),
В.А. ЛАПТИН (straqker@bk.ru)
(МГУ им. М.В. Ломоносова),
А.С. МАНДЕЛЬ, д-р техн. наук (almandel@yandex.ru)
(Институт проблем управления им. В.А. Трапезникова РАН, Москва)

ОЦЕНКА ПАРАМЕТРОВ И СТРУКТУРЫ СИСТЕМ МАССОВОГО ОБСЛУЖИВАНИЯ С ПЕРЕКЛЮЧЕНИЕМ КАНАЛОВ

Рассматривается задача оценки структуры и параметров системы массового обслуживания при реализации системы оптимального переключения основных каналов обслуживания в моменты контроля, отстоящие друг от друга на фиксированный временной шаг. Для решения задачи оценки структуры и параметров модели предложены процедуры экспертно-классификационного анализа, структурного прогнозирования и экспертно-статистической обработки данных.

Ключевые слова: управляемые системы массового обслуживания, марковский входящий поток, оценивание структуры и параметров, экспертно-классификационный анализ, структурное прогнозирование, экспертно-статистическая обработка данных.

DOI: 10.31857/S0005231021110040

1. Введение

Рассматривается задача оценки структуры и параметров модели оптимального управления системой массового обслуживания (СМО), описанной в [1], где сделано предположение о скачкообразном марковском пошаговом изменении интенсивности простейшего входящего потока. В результате сделан вывод о том, что классические методы статистического оценивания не вполне пригодны для исследования структуры (числа элементов множества и значений различных интенсивностей входящего потока), а также вероятностей перехода из состояния в состояние. В качестве альтернативы предлагается набор процедур экспертно-классификационного анализа [2, 3], структурного прогнозирования [4] и экспертно-статистической обработки данных [5].

2. Модель системы массового обслуживания

Как отмечено в [1], в исследуемой СМО число рабочих каналов обслуживания может изменяться в моменты контроля, отстоящие друг от друга на фиксированное время — шаг контроля. При этом считается, что в СМО поступает простейший входящий поток, интенсивность которого $\lambda(t)$ на протяжении шага постоянна, а в моменты контроля претерпевает скачкообразные

марковские изменения, принимая конечное число k значений λ_i из дискретного множества $\Lambda = \{\lambda_i, i \in \overline{1,k}\}$. Задача заключается в том, чтобы сформировать стратегию переключения рабочих каналов (отключения части работающих каналов или введение в действие дополнительных резервных каналов), которая минимизирует средние затраты СМО на заданном N-шаговом периоде планирования. Задана матрица вероятностей перехода соответствующей однородной марковской цепи $P = \|p_{ij}\|$, где p_{ij} – это вероятность перехода (в момент контроля) от интенсивности λ_i, $i \in \overline{1,k}$, на предыдущем шаге к интенсивности λ_j, $j \in \overline{1,k}$, на следующем шаге.

В [1] показано, что решение задачи о выборе оптимальной стратегии переключения каналов сводится к решению следующей системы уравнений дискретного динамического программирования:

$$C_1^*(\lambda_i, m) = \min_{u \geqslant \underline{u_i}} C^{(1)}(\lambda_i, m, u), \tag{1}$$

$$C_n^*(\lambda_i, m) = \min_{u \geqslant \underline{u_i}} \left\{ C^{(1)}(\lambda_i, m, u) + \alpha \sum_{j=1}^{l} p_{ij} C_{n-1}^*(\lambda_j, u) \right\}, \quad n \in \overline{2,N}, \tag{2}$$

где $C_n^*(\lambda_i, m)$ – минимальное возможное значение суммарных средних затрат на n последних шагах процесса управления, когда математическое ожидание берется по траектории изменения интенсивности входящего потока, интенсивность которого совершает марковские скачки. Переменная u в уравнениях (1)–(2) определяет текущее (за n шагов до конца процесса) значение управляющего решения о числе включаемых рабочих каналов.

3. Постановка задачи

Главное, что предстоит оценить в процессе практического использования модели (1), (2), – это точки множества значений интенсивностей входящего потока $\Lambda = \{\lambda_i\}$, $i \in \overline{1,k}$ (структура системы) и значения вероятностей перехода (параметры системы) соответствующей однородной марковской цепи $P = \|p_{ij}\|$, где p_{ij} – это вероятность перехода (в момент контроля) от интенсивности λ_i, $i \in \overline{1,k}$, на предыдущем шаге к интенсивности λ_j, $j \in \overline{1,k}$, на следующем шаге. При этом важно понимать, что модель (1)–(2) является лишь приближением к описанию реальной ситуации, в которой четких скачков не наблюдается (процесс, конечно, размыт во времени, да и сами значения λ_i из множества Λ могут "ползти", постепенно изменяясь от шага к шагу). Поэтому довольно сложно представить себе процедуру статистического оценивания указанных параметров и вероятностей перехода в рамках классической статистической теории.

В связи с этим для решения проблемы оценки структуры системы и ее параметров в работе была избрана концепция структурного прогнозирования в понимании публикации [4] с привлечением методов экспертно-классификационного анализа [2, 3] и элементов экспертно-статистической обработки данных [5].

3.1. Результаты наблюдений для управляемых СМО

Естественно, что результаты наблюдений сильно связаны с предметной областью, к которой относятся соответствующие СМО. При этом эти данные могут быть чрезвычайно разнородны, что представляет собой дополнительную проблему [6]. Как минимум, они представляют собой данные о средних интенсивностях входящих потоков на каждом шаге, сведения о конкретных СМО (бюджет, положение на рынке, приоритеты обслуживания различных требований, конъюнктура рынка и т.д.). Например, если СМО представляет собой кассовый зал кинотеатра, то привлекательность услуги по продаже билета может обусловливаться жанром фильма, его бюджетом, названием студии-производителя, именами режиссера и актеров и другими признаками. При этом в качестве разных объектов могут выступать отдельные отрезки времени с результатами наблюдений за одной и той же системой массового обслуживания. Каждый раз задача выбора набора наблюдений решается в зависимости от указанных обстоятельств.

Следует понимать, что последующие применения результатов обработки этих данных представляют собой предложение вариантов управляющих решений для некой системы поддержки принятия решений (СППР), что также порождает необходимость учитывать различные особенности СППР. Прежде всего приходится иметь в виду наличие объектов нечисловой природы [7], обязательность выполнения экспертных оценок [8], включая проведение в некоторых случаях многовариантной экспертизы [9], и, наконец, особенности инженерии информационно-управляющих систем [10].

В результате решения задачи выбора наблюдений образуется совокупность объектов. Каждый объект описывается набором параметров и представляется отрезком траектории в пространстве состояний. Соответствующий набор отрезков траекторий назовем обучающей выборкой. Изучается поведение этого множества объектов (отрезков траекторий) в дискретные моменты времени. Вводится в рассмотрение m-мерное (по числу параметров) пространство параметров X, в котором j-й объект в момент времени n представляется точкой $\mathbf{x_j}(n) = \{x_j^{(1)}(n), x_j^{(2)}(n), \ldots, x_j^{(m)}(n)\}$. Совокупность векторов $\{\mathbf{x_j}(1), \mathbf{x_j}(2), \ldots, \mathbf{x_j}(n)\}$ представляет отрезок траектории движения (динамики) j-го объекта.

3.2. Формирование начального приближения к структуре кластеров

Начальное приближение к структуре кластеров, в которую отображается рассматриваемая совокупность объектов, строится с использованием комплексного алгоритм автоматической классификации [4].

Критериальной функцией при построении начального кластерного разбиения является функционал вида

$$J_1 = \sum_{i=1}^{r} \frac{p_i}{p} K(A_i, A_j) \quad (r \ - \text{число классов}), \tag{3}$$

где через A_i обозначены кластеры точек, r – число этих кластеров, а

$$K(A_i, A_j) = \frac{2}{p_i(1-p_i)} \sum_{i=1}^{p_i} \sum_{j>i} K(x_i, x_j) \tag{4}$$

— мера близости между собой отдельных кластеров, которая рассчитывается с помощью классической потенциальной функции близости двух точек [11]:

$$K(x, y) = \frac{1}{1 + \alpha R^m(\mathbf{x}, \mathbf{y})}, \tag{5}$$

где $R^m(\mathbf{x}, \mathbf{y})$ – некая метрика в пространстве X, а α и m – настраиваемые параметры алгоритма.

Определение начального числа кластеров r может быть в зависимости от контекста и степени формализации конкретной прикладной проблемы выполнено автоматически или с использованием процедур экспертно-классификационного анализа. Соответствующие процедуры могут включать в свой состав процедуры перекрестной экспертизы [9], рассчитанные на то, чтобы добиться согласования мнений задействованных в процедуру принятия окончательных решений экспертов. При этом среди экспертов почти всегда присутствует лицо, принимающее решения (ЛПР), которого в рамках лексики, что используется в экспертно-статистических процедурах принятия решений [5], называют главным экспертом.

4. Этапы решения задачи

4.1. Процедура оценки начального приближения к вероятностям перехода объектов

После того как в соответствии с рекомендациями подраздела 3.2 выделена начальная кластерная структура $\{A_i\}_{(i=1)}^{r}$, определяются "центры"[1] кластеров начального разбиения $\mathbf{a_i}(1)$ всех выделенных кластеров $R_{ji}^{(1)} = R(\mathbf{x_j}(1), \mathbf{a_i}(1))$, $i = 1, \ldots, k$; $j = 1, \ldots, p$. При этом, как отмечается в [12], в некоторых случаях для повышения степени корректности решений задачи выделения центра может понадобиться отказаться от классических евклидовых метрик в пользу метрик более экзотических.

При этом значения интенсивностей входящего потока λ_i, которые соответствуют объектам $\mathbf{a_i}(1)$, формируют результирующее множество возможных состояний идентифицируемой конечной цепи Маркова $\Lambda = \{\lambda_i\}, i \in \overline{1, r}$.

Затем при движении объектов по их индивидуальным траекториям, т.е. в динамике, начинаются переходы объектов из кластера в кластер. В рамках идеологии структурного прогнозирования [4] можно рассчитать вероятности перехода каждого из объектов в каждый из выделенных кластеров.

[1] Центры кластеров иногда называют эталонами кластера.

Элементы матрицы вероятностей перехода объекта j в кластер i[2], значения $p_{ji}^{(1)} = p_{ji}(1)$, могут быть рассчитаны по формуле

$$p_{ji}^{(1)} = \frac{\alpha_j^{(1)}}{R_{ji}^{(1)}}, \tag{6}$$

где $R_{ji}^{(1)}$ – расстояние по выбранной метрике от точки (объекта) j до центра i-го кластера $\mathbf{a_i}(1)$ в начальный момент времени, а нормирующий множитель $\alpha_j^{(1)}$ определяется выражением

$$\alpha_j^{(1)} = \frac{\prod_{i=1}^{r} R_{ji}^{(1)}}{\sum_{j=1}^{p} \frac{1}{R_{ji}^{(1)}} \prod_{i=1}^{r} R_{ji}^{(1)}}. \tag{7}$$

4.2. Процедура оценки последующих приближений

После выделения начальной структуры и оценки начального приближения к вероятностям перехода объектов на следующих шагах элементы матрицы вероятностей перехода объекта j в кластер i модифицируются при помощи процедуры соотнесения между собой расстояний между точкой (объектом) j и центром кластера i в два соседних момента времени, т.е. в начале и конце каждого шага. Обозначим соответствующее изменение расстояний через $\Delta R_{ji}^{(n)} = R_{ji}^{(n)} - R_{ji}^{(n-1)}$.

Если окажется, что j-я точка (объект) в момент n совпадала с центром какого-либо кластера (под номером i_0), т.е., что $R_{ji_0}^{(n)} = 0$, то принимается соглашение, что вероятность для данного объекта остаться в этом кластере равна 1, а стало быть, вероятность перехода в другой кластер равна 0:

$$p_{ji}^{(n)} = \begin{cases} 1, & \text{если } i = i_0, \\ 0, & \text{если } i = 1, \ldots, k, \quad i \neq i_0. \end{cases}$$

Для случая, когда $R_{ji_0}^{(n)} \neq 0$, происходит модификация всех переходных вероятностей по формуле, которая несколько отличается от ее аналога в [4]:

$$p_{ji}^{(n)} = \gamma \left\{ p_{ji}^{(n-1)} + \left[\frac{1 + \operatorname{sign}(\Delta R_{ji}^{(n)})}{2} - p_{ji}^{(n-1)} \operatorname{sign}(\Delta R_{ji}^{(n)}) \right] R_{ji}^{(n)} \right\}, \tag{8}$$

где, как обычно, $\operatorname{sign}(z) = \begin{cases} 1, & \text{если } z \geqslant 0, \\ 0, & \text{если } z < 0, \end{cases}$ а γ – нормирующий множитель, определяемый условием нормировки $\sum_{i=1}^{k} p_{ji}^{(n)} = 1$:

$$\gamma = \frac{1}{1 + \left[\frac{1+\operatorname{sign}(\Delta R_{ji}^{(n)})}{2} - p_{ji}^{(n-1)} \operatorname{sign}(\Delta R_{ji}^{(n)}) \right] \Delta R_{ji}^{(n)}}. \tag{9}$$

[2] Заметим, что эти вероятности не совпадают с искомыми вероятностями из алгоритма (2) (см. ниже подраздел 4.3).

4.3. Способы оценки искомых вероятностей перехода p_{ij} из формулы (2)

Поскольку в (2) используются вероятности перехода p_{ij} рассматриваемой СМО из кластера i в кластер j, а в двух предыдущих пунктах приводятся прогнозные оценки вероятностей перехода конкретного объекта в конкретный кластер, то в зависимости от характера обучающей выборки можно поступать следующими двумя способами.

Способ 1. Если рассматриваемая СМО – только один из объектов обучающей выборки с номером s, тогда для оценки вероятностей p_{ij} разумно использовать соотношение

$$p_{ij} = p_{sj}^{(L)}, \quad \text{объект с номером } s \text{ принадлежит кластеру } A_i, \tag{10}$$

где L – длина отрезка временного ряда в обучающей выборке, A_i – кластер объектов с номером i, а оценки p_{sj}^{L} рассчитываются по формулам (8)–(9).

Способ 2. Если рассматриваемая СМО — единственный объект обучающей выборки, иначе говоря обучающая выборка построена по предыстории движения этой СМО в различных ее отрезках, тогда для оценки вероятностей p_{ij} можно использовать соотношение

$$p_{ij} = \frac{1}{p_i} \sum_{s \in A_i} p_{sj}^{(L)}, \tag{11}$$

где L – длина отрезка временного ряда в обучающей выборке, A_i – кластер объектов с номером i, оценки p_{sj}^{L} рассчитываются по формулам (8)–(9), а p_i – это число объектов в кластере A_i.

Можно предложить и другие способы оценки искомых вероятностей p_{ij} из алгоритма (2) с возможным привлечением экспертов для осуществления экспертно-статистической обработки результатов оценивания [5].

5. Применения

Предложенная методика была применена для расчета параметров модели многоканальной системы массового обслуживания, которой описывалось функционирование операционного отделения Института нейрохирургии им. Н.А. Бурденко.

6. Заключение

Для решения задачи оценки структуры и параметров модели выбора оптимальной стратегии переключения каналов в управляемой системе массового обслуживания (СМО) предложены процедуры экспертно-классификационного анализа, структурного прогнозирования и экспертно-статистической обработки данных.

Предстоит сравнить различные способы представления оценок вероятностных параметров задачи управления СМО через оценки, полученные методом структурного прогнозирования.

При дальнейшем развитии предложенной модели и методов обработки и анализа данных придется, по-видимому, использовать новые возможности, связанные с применениями теории нейронных сетей [13].

СПИСОК ЛИТЕРАТУРЫ

1. *Mandel A., Laptin V.* Channel Switching Threshold Strategies for Multichannel Controllable Queuing Systems // Communicat. Comput. Inform. Sci. 2020. V. 1337. P. 259–270. link.springer.com/chapter.

   https://doi.org/10.1007/978-3-030-66242-4_21

2. *Бауман Е.В., Дорофеюк А.А.* Классификационный анализ данных // Труды Междунар. конф. по проблемам управления. Том 1. М.: СИНТЕГ, 1999. С. 62–77.
3. *Дорофеюк А.А.* Методология экспертно-классификационного анализа в задачах управления и обработки сложноорганизованных данных (история и перспективы развития) // Проблемы управления. 2009. № 3.1. С. 19–28.
4. *Дорофеюк Ю.А., Чернявский А.Л* Интеллектуальные методы динамического структурного анализа данных // Датчики и системы. 2019. № 10. С. 3–8.
5. *Мандель А.С., Дорофеюк Ю.А.* Экспертно-классификационная и экспертно-статистическая обработка информации и принятие решений. М.: Макс Пресс, 2010. С. 165–168.
6. *Воронцов К.В.* Десять открытых проблем вероятностного тематического моделирования // Интеллектуализация обработки информации: Пленарный доклад на 13-й Междунар. конф. ИОИ-2020.
7. *Дорофеюк А.А., Покровская И.В., Чернявский А.Л.* Структуризация объектов нечисловой природы // Информационные технологии и вычислительные системы. 2018. № 1. С. 16–21.
8. *Дорофеюк А.А., Покровская И.В., Чернявский А.Л.* Экспертные методы анализа и совершенствования систем управления // АиТ. 2004. № 10. С. 172–188.

   Dorofeyuk A.A., Pokrovskaya I.V., Chernyavsky A.L. Expert Methods to Analyze and Perfect Management Systems // Autom. Remote Control. 2004. V. 65. No. 10. P. 1675–1688.

9. *Дорофеюк А.А., Гольдовская М.Д., Киселева Н.Е. и др.* Процедуры коллективной многовариантной экспертизы в задачах анализа и совершенствования социально-экономических систем // Информационные технологии и вычислительные системы. 2016. № 4. С. 53–68.
10. *Трояновский В.М.* Программная инженерия информационно-управляющих систем в свете прикладной теории случайных процессов. М.: Издательский дом Форум. 2019.
11. *Айзерман М.А., Браверман Э.М., Розоноэр Л.И.* Метод потенциальных функций в теории обучения машин. М.: Наука, 1970.
12. *Шибзухов З.М.* Об одном робастном подходе к поиску центров кластеров // Интеллектуализация обработки информации: Тезисы докладов 13-й Междунар. конф. ИОИ-2020. М.: РАН, 2020. С. 121–122.
13. *Хайкин С.* Нейронные сети. Полный курс. М.-СПб.: Диалектика, 2020.

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

Поступила в редакцию 20.01.2021

После доработки 16.05.2021

Принята к публикации 30.06.2021

© 2021 г. Б.В. КУПРИЯНОВ, канд. техн. наук (kuprianovb@mail.ru),
А.А. ЛАЗАРЕВ, д-р физ.-мат. наук (jobmath@mail.ru)
(Институт проблем управления им. В.А. Трапезникова РАН, Москва)

ОПТИМИЗАЦИЯ РЕКУРСИВНОГО КОНВЕЙЕРА СВЕДЕНИЕМ К ЗАДАЧЕ УДОВЛЕТВОРЕНИЯ ОГРАНИЧЕНИЙ[1]

Рассматривается задача оптимизации расписания рекурсивного конвейера. Для этого вводится определение конвейера, описываемого связным ациклическим графом, каждая вершина которого представляет собой операцию или функцию управления, ассоциированную с соответствующей рекурсивной функцией из некоторого конечного набора. Каждая рекурсивная функция определяет отношение предшествования операции конвейера. Рассматривается решение задачи минимизации времени выполнения заказа конвейером на конечном множестве возобновляемых ресурсов. Решение осуществляется сведением к задаче удовлетворения ограничений.

Ключевые слова: теория расписаний, балансировка конвейера, flow-shop задачи, задача удовлетворения ограничений.

DOI: 10.31857/S0005231021110052

1. Введение

Задачи RCPSP (Resource-Constrained Project Scheduling Problem) являются одними из основных в теории расписаний. В задачах RCPSP задано множество заказов, которые необходимо обслужить на конечном множестве машин (далее по тексту ресурсов). Заказ тождественен множеству упорядоченных операций и характеризуется временем выполнения. Необходимо минимизировать время выполнения заказов на множестве распределений ресурсов по операциям. Данного вида задачи актуальны и для конвейерных систем [1]. Работы, которые рассматривают методы балансировки сборочного конвейера и планирования ресурсов на этапе проектирования конвейера, приведены в [2]. В [3] приводятся комплексный обзор и анализ различных методов проектирования и планирования конвейерных систем. Большинство работ посвящено балансировке сборочного конвейера (ALB). Для ALB модели предложены методы поиска оптимальных решений с использованием линейного программирования [4, 5] и целочисленного программирования [6]. Из современных работ по оптимизации планирования ресурсов для типовых задач теории расписаний представляют интерес работы [7–10].

Важно отметить две особенности постановки таких задач.

[1] Работа выполнена при финансовой поддержке Российского фонда фундаментальных исследований (проект № 20-58-S52006).

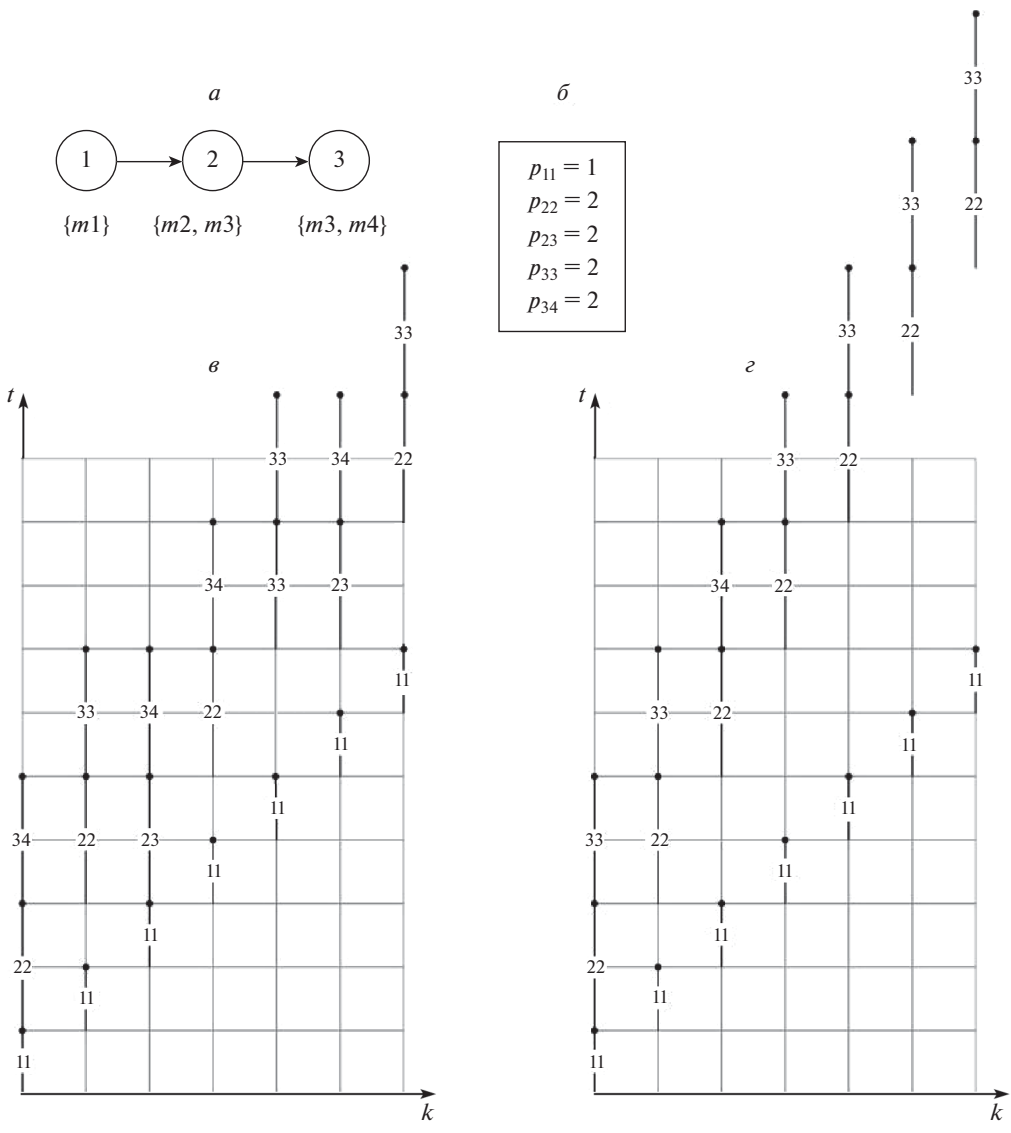

Рис. 1. Пример двух методов распределения ресурсов конвейера.

Первая состоит в том, что ресурсы распределяются до процесса выполнения заказа и на весь процесс выполнения. Это приводит к тому, что каждый ресурс закреплен за какой-либо одной операцией, даже если он может потенциально использоваться несколькими операциями. Однако можно показать на примере, что "перебрасывание ресурса" с одной операции на другую может привести к построению более эффективного расписания. На рис. 1,*а* показана модель конвейера из трех последовательно выполняемых операций. Под операциями в фигурных скобках указаны потенциально используемые ресурсы из множества $\{m1, m2, m3, m4\}$. На рис. 1,*б* указаны длительности операций

при использовании различных ресурсов. Первый индекс — номер операции, второй индекс — номер ресурса. На рис. 1,*в* и 1,*г* показаны временные диаграммы выполнения операций для семи заказов. По оси абсцис номер заказа, по оси ординат время. Отрезок с индексом i, j определяет временной интервал выполнения i-й операции с использованием j-го ресурса для k-го заказа. На рис. 1,*г* показана оптимальная диаграмма с закреплением ресурса за операцией на все время выполнения заказов, а на рис. 1,*в* диаграмма с распределением ресурсов без закрепления за операциями. Из сравнения диаграмм видно преимущество по времени второго метода.

Вторая особенность состоит в использовании ограниченного набора отношений предшествования. Это отношение предшествования между операциями и определяемое с помощью предиката and.

В данной статье предлагается решение, преодолевающее оба эти ограничения. Рассматривается распределение ресурсов с возможностью использования одного ресурса несколькими операциями и с расширением отношений предшествования с помощью рекурсивных функций. Задачи Удовлетворения Ограничений (ЗУО) и методы Constraint Programming [11–13] наряду с прочим используются для решения задач теории расписаний. В данной работе рассматривается решение задачи RCPSP сведением ее к ЗУО применительно к конвейерам, описываемым рекурсивными функциями [14]. Время выполнения операции i зависит от используемого ресурса j, т.е. равно p_{ij}. Примеры описания возможностей и применения таких конвейеров приведены в [15].

Статья имеет следующую структуру. В разделе 2 приведено определение рекурсивного конвейера. Подробно описаны рекурсивные функции. В разделе 3 приведены примеры рекурсивных конвейеров. В разделе 4 описывается определение ЗУО и задача распределения ресурсов конвейера формулируется как ЗУО. Раздел 5 является заключением, в котором рассматривается проблема сложности алгоритма решения поставленной в статье ЗУО.

2. Определение конвейера

Модель и свойства рекурсивного конвейера подробно рассмотрены в [14, 16]. Однако они описаны в предположении, что за каждой операцией закреплен свой ресурс на время выполнения всех заказов. В данном разделе определение рекурсивных функций конвейера дано с учетом распределения ресурса для каждой пары (операция, заказ).

Модель 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 из множества $I = \{1, \ldots, n\}$ таким образом, что первые n_0 $(1 \leqslant n_0 < n)$ вершин являются начальными, а вершина n конечной. Графическое изображение вершин и дуг графа представлено на рис. 2. Здесь i, j, j_1, j_2 – номера операций (вершин), k – номер заказа, $p_{i,j}$ – время выполнения i-й операции при использовании j-го ресурса и q_i – коэффициент мультиплицирования или ре-

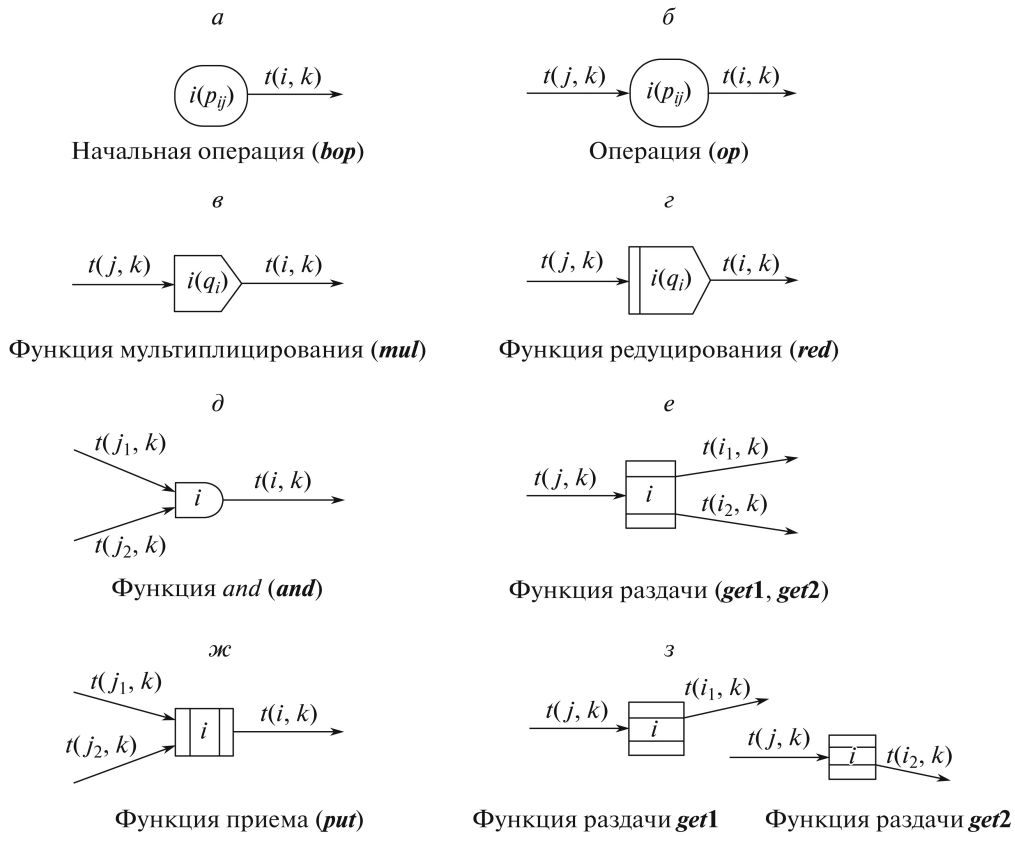

Рис. 2. Графическая нотация операций и спусковых функций конвейера.

дуцирования. Каждая вершина графа может иметь одну входную дугу (см. рис. 2 варианты *б*,*в*,*г*,*е*) или две (см. рис. 2 варианты *д*,*ж*) в зависимости от типа вершины. На множестве номеров вершин графа определены функции предшествования $pred, pred1, pred2 : I \to I$.

Функция $pred(i)$ определена для вершины, имеющей одну предшествующую вершину и вычисляет номер j вершины графа такой, что $(j,i) \in A$.

Следующие две функции определены для вершин, имеющих две предшествующие вершины, условно *вершина* 1 и *вершина* 2:

$pred1(i)$ – вычисляет номер j_1 *вершины* 1 графа такой, что существует дуга $(j_1,i) \in A$;

$pred2(i)$ – вычисляет номер j_2 *вершины* 2 графа такой, что существует дуга $(j_2,i) \in A$.

Каждая вершина графа может соответствовать операции или спусковой функции. Операции ассоциированы с производственными операциями. Спусковые функции определяют отношения временного предшествования операций. Вершины графа помечены с помощью отображения

$$type : I \to E, \quad \text{где} \quad E = \{\mathbf{bop}, \mathbf{op}, \mathbf{and}, \mathbf{mul}, \mathbf{red}, \mathbf{get1}, \mathbf{get2}, \mathbf{put}\}$$

на множество из восьми элементов. Каждая константа множества E обозначает тип вершины графа, а вершина имеет соответствующую графическую нотацию (рис. 2). Конвейер использует множество ресурсов $M = \{m_1, \ldots, m_r\}$. Каждая операция i потенциально может использовать ресурсы из некоторого множества $D_i \subseteq M$. Для каждого $1 \leqslant i \leqslant n$ определим переменную x_{ik} на множестве D_i. Если $x_{ik} = m_j$, то операция i при выполнении k-го заказа использует ресурс $m_j \in D_i$. Время выполнения операции равно p_{ij} и является константой. В каждый момент времени каждый ресурс может использоваться только одной операцией. Прерывания операций запрещены.

Спусковые функции ресурсы не потребляют, и время их выполнения равно 0.

Расписание выполнения операций строится с помощью рекурсивных функций $R : I \times K \times M \to T$, где K-конечное множество номеров заказов, а T-время.

Например, рекурсивная функция

$$t(i,k) = r_1(t(i,k-1), t(j,k), p_{il})$$

или

$$t(i,k) = r_2(t(i,k-1), t(j_1,k), t(j_2,k), p_{il}),$$

где $j = pred(i)$, $j_1 = pred1(i)$, $j_2 = pred2(i)$, вычисляет время завершения обработки i-й операцией k-го заказа при $k \in K$ исходя из времени завершения ею обработки $(k-1)$-го заказа и из времен завершения обработки k-го заказа предшествующими операциями. Рекурсивные функции имеют два аргумента: i – номер операции (номер вершины графа) и k – номер заказа. Для каждого типа вершины, т.е. элемента множества E, своя рекурсивная функция.

Рекурсивная функция $t(i,k)$, вычисляемая по формуле (1), обеспечивает суперпозицию рекурсивных функций, соответствующих вершинам графа, и является обращением к одной из них в соответствии с типом вершины.

$$t(i,k) = \begin{cases} bop(i,k), & \textbf{если} \quad type(i) = \textbf{bop}; \\ op(i,k), & \textbf{если} \quad type(i) = \textbf{op}; \\ and(i,k), & \textbf{если} \quad type(i) = \textbf{and}; \\ mul(i,k), & \textbf{если} \quad type(i) = \textbf{mul}; \\ red(i,k), & \textbf{если} \quad type(i) = \textbf{red}; \\ get1(i,k), & \textbf{если} \quad type(i) = \textbf{get1}; \\ get2(i,k), & \textbf{если} \quad type(i) = \textbf{get2}; \\ put(i,k), & \textbf{если} \quad type(i) = \textbf{put}. \end{cases} \tag{1}$$

Вычисление времени завершения выполнения k-го заказа конвейером осуществляется обращением к рекурсивной функции $t(n,k)$.

Далее описываются все виды вершин: их назначение, тип и соответствующая типу рекурсивная функция.

1. Начальная производственная операция (см. *bop* рис. 2,*а*) с номером $1 \leqslant i \leqslant n_0$ характеризуется временем выполнения k-го заказа p_{ij}, если операция использует ресурс m_j ($x_{ik} = m_j$). Время завершения выполнения данной операцией k-го заказа определяется как время завершения выполнения $(k-1)$-го заказа плюс время выполнения операции p_{ij}. Если операция при выполнении k-го и $(k-1)$-го заказа использует разные ресурсы, то времена их выполнений совмещаются. Время выполнения нулевого заказа равно p_{ij}.

(2) $bop(i,k)$:

$$bop = \begin{cases} p_{ij}, \\ \quad \textbf{если } (x_{ik} = m_j)\&(k = 0); & \text{а)} \\ bop(i,k-1) + p_{ij}, \\ \quad \textbf{если } (x_{ik} = x_{i,k-1})\&(x_{ik} = m_j)\&(k > 0); & \text{б)} \\ bop(i,k-1) + p_{ij} - p_{il}, \\ \quad \textbf{если } (x_{ik} \neq x_{i,k-1})\&(x_{i,k-1} = m_l)\&(x_{ik} = m_j)\&(k > 0). & \text{в)} \end{cases}$$

2. Не начальная производственная операция (см. *op* рис. 2,*б*) с номером $n_0 < i$ характеризуется временем выполнения p_{ij}, если используется ресурс m_j. Время завершения выполнения данной операцией k-го заказа определяется как максимум времен: завершения выполнения данной операцией $(k-1)$-го заказа и завершения выполнения предшествующей ей операцией k-го заказа и плюс время выполнения операции p_{ij}. При этом начала выполнений i-й операцией k-го и $(k-1)$-го заказов совмещаются, если для выполнения используются разные ресурсы. Время выполнения нулевого заказа равно времени выполнения нулевого заказа предшествующей операцией плюс p_{ij}.

(3) $op(i,k)$:

$$op = \begin{cases} t(pred(i),k) + p_{ij}, \quad \textbf{если } (x_{ik} = m_j)\&(k = 0); & \text{а)} \\ t(pred(i),k) + p_{ij}, \\ \quad \textbf{если } (t(pred(i),k) \geqslant t(i,k-1))\&(x_{ik} = m_j)\&(k > 0); & \text{б)} \\ t(pred(i),k) + p_{ij}, \\ \quad \textbf{если } (t(pred(i),k) < t(i,k-1))\& \\ \qquad \&(x_{ik} \neq x_{i,k-1})\&(x_{ik} = m_j)\&(k > 0); & \text{в)} \\ t(i,k-1) + p_{ij}, \\ \quad \textbf{если } (t(pred(i),k) < t(i,k-1))\& \\ \qquad \&(x_{ik} = x_{i,k-1})\&(x_{ik} = m_j)\&(k > 0). & \text{г)} \end{cases}$$

3. Спусковая функция *and* (см. *and* рис. 2,*д*) вычисляет время завершения выполнения k-го заказа для двух предшествующих операций.

(4) $$and(i,k) : \ and = \max(t(pred1(i),k), t(pred2(i),k)).$$

4. Спусковая функция мультиплицирования (см. *mul* рис. 2,*в*) для каждого времени завершения предшествующей ей операции вычисляет q_i времен

завершения операции i. Это означает, что на одно выполнение предшествующей i операции осуществляется q_i ($q_i \geqslant 1$) выполнений операции i.

$$mul(i,k):\ mul = t(pred(i), \lfloor k/q_i \rfloor), \tag{5}$$

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

5. Спусковая функция редуцирования (см. *red* рис. 2,*г*) является обратной по отношению к функции мультиплицирования и вычисляет время завершения выполнения очередной партии из q_i ($q_i \geqslant 1$) заказов.

$$red(i,k):\ red = t(pred(i), (k+1)q_i - 1). \tag{6}$$

6. Спусковая функция раздачи (см. *get* рис. 2,*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) см. рис. 2,*з*. Для верхнего варианта спусковая функция $get1$ вычисляет времена завершений четных заказов, т.е. $t(i,0) = t(p,0)$, $t(i,1) = t(p,2)$, $t(i,2) = t(p,4), \ldots$

$$get1(i,k):\ get1 = t(pred(i), 2k),\ k \geqslant 0. \tag{7}$$

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

$$get2(i,k):\ get2 = t(pred(i), 2k+1),\ k \geqslant 0. \tag{8}$$

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

$$put(i,k) : put = \begin{cases} t(pred1(i), k), \\ \qquad \textbf{если } k = 0; \\ \max(put(i,k-1), t(pred2(i), k/2)), \\ \qquad \textbf{если } (k \bmod 2 = 0)\&(k > 0); \\ \max(put(i,k-1), t(pred1(i), (k-1)/2)), \\ \qquad \textbf{если } k \bmod 2 = 1. \end{cases} \tag{9}$$

Здесь $x \bmod y$ – вычисление остатка от деления x на y.

На рис. 3 представлен пример графа конвейера и соответствующей суперпозиции рекурсивных функций для случая, когда каждой операции выделен один ресурс на все время процесса. В реальности такой граф не строится, он является результатом процесса вычисления рекурсивной функции $t(i,k)$.

Время начала функционирования конвейера полагается равным 0.

В предлагаемой модели интерпретация операций и спусковой функции *and* не отличается от таковых в теории расписаний. Остальной набор спусковых функций выработан практикой, на основе построения моделей и является двумя парами прямых и обратных функций ($mul-red$ и $get-put$). Возможно построение новых функций, расширяющих отношение предшествования.

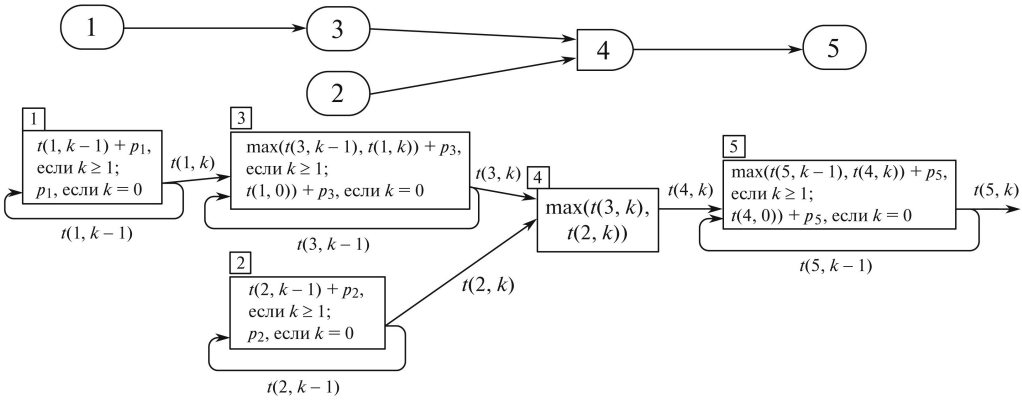

Рис. 3. Пример графа конвейера (сверху) и соответствующего графа суперпозиции рекурсивных функций (снизу).

3. Примеры моделей

Примеры некоторых моделей конвейеров для различных приложений описаны в [15]. В данной статье приведем дополнительные примеры рекурсивных конвейеров. Перед тем как привести примеры, отметим, что временная диаграмма должна составляться для каждой операции отдельно, это ненаглядно

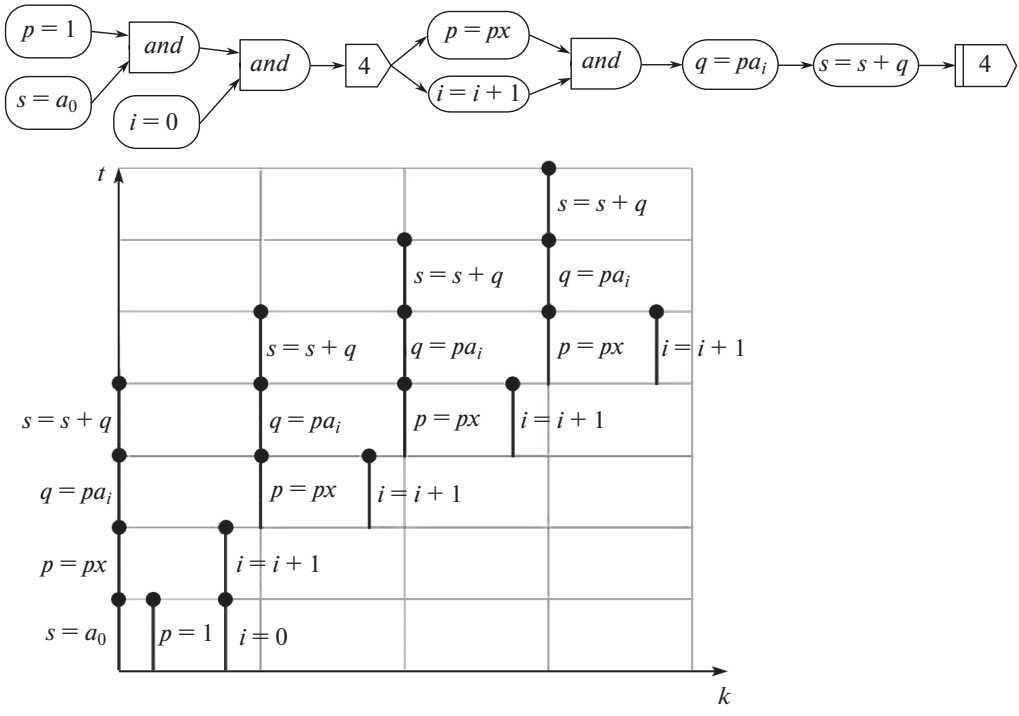

Рис. 4. Пример модели вычислительного конвейера и временной диаграммы.

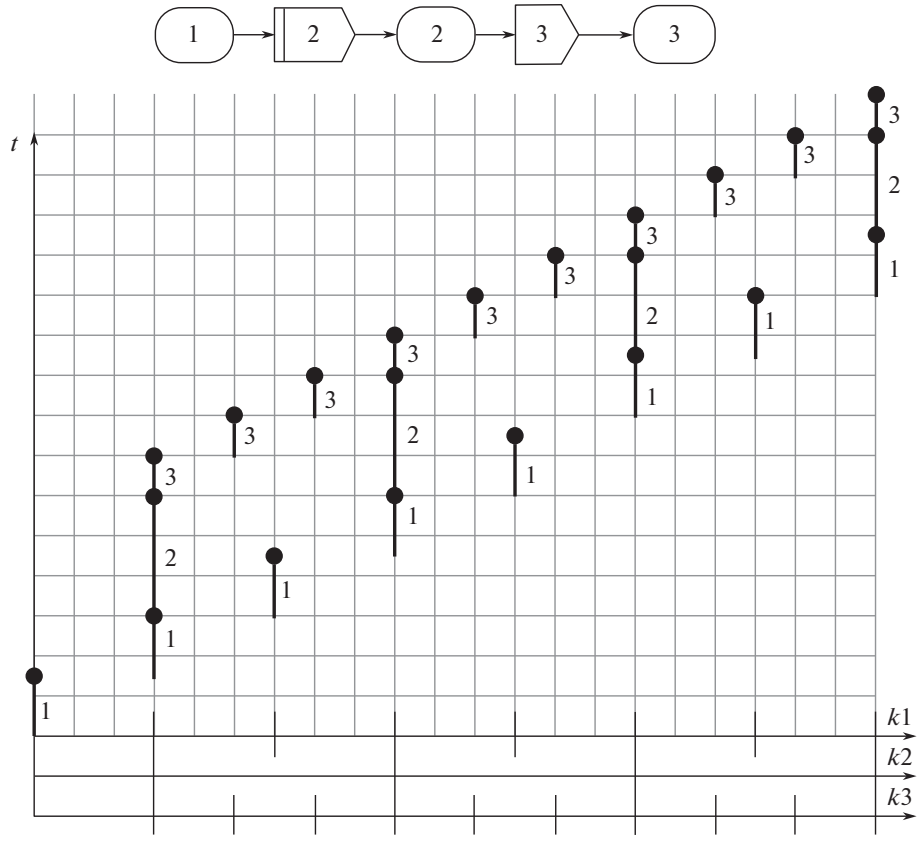

Рис. 5. Пример простого конвейера с функциями мультиплицирования и редуцирования.

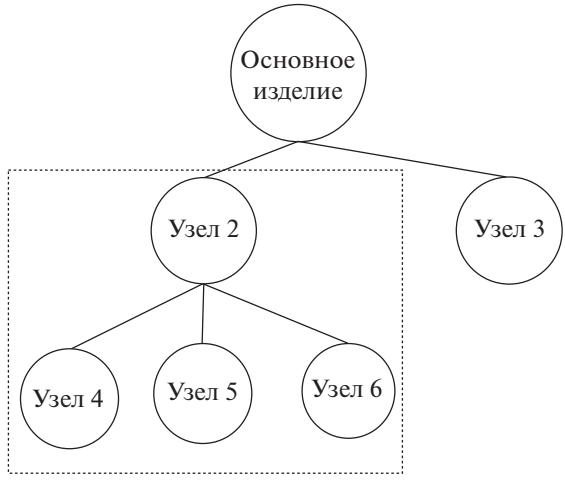

Рис. 6. Структура сборочного изделия.

и громоздко, диаграммы будут совмещены в одной. По оси абсцисс будут отложены номера заказов, а по оси ординат время. Некоторые отрезки времени при таком совмещении в одной диаграмме будут сливаться. Чтобы этого не происходило, отрезки, имеющие по оси абсцис значение k, будут размещаться в интервале между делениями k и $(k+1)$.

Пример 1. На рис. 4 представлена модель конвейера, вычисляющего полином

$$p = a_0 + a_1 x^1 + a_2 x^2 + a_3 x^3 + a_4 x^4,$$

и соответствующая временная диаграмма в предположении, что каждая операция выполняется своим процессором. Времена выполнения всех операций равны 1.

Пример 2. Прежде чем рассмотреть второй пример, разберем вспомогательный. На рис. 5 представлен пример конвейера и его временной диаграммы. Конвейер состоит из трех операций, функции редуцирования с коэффициентом 2 и функции мультиплицирования с коэффициентом 3. На диаграмме видно, что все операции выполняются с разной кратностью, но процесс является конвейерным. Чтобы понять, что такой конвейер может иметь место на практике, рассмотрим пример конвейерной сборки некоторого изделия, которое имеет структуру, представленную на рис. 6. Особенность сборки состоит в том, что узел 2 и узел 3 собираются в отдаленном от основной сборки месте. Поэтому на месте собирается партия из $q2$ узлов, которая потом перевозится на склад основного производства. Аналогично узел 3 собирается в другом отдаленном месте и перевозится партиями по $q3$ узлов. Требуется составить расписание: производства узлов, хранения на складе, перевозки и сборки основного изделия как единого конвейерного процесса. Пример модели такого процесса приведен на рис. 7. Суть ее в том, что узел 2 по мере конвейерной сборки накапливается на складе в количестве $q2$ и потом эта партия доставляется на место основной сборки. Конвейерность этого процесса позволяет описывать функции редуцирования и мультиплицирования. Аналогично для узла 3.

4. Постановка задачи

В данном разделе рассматривается рекурсивный конвейер, представленный в виде ациклического связного графа, для которого необходимо распределением ресурсов минимизировать время выполнения $\hat{k}$ заказов, где $\hat{k}$ – некоторая константа. Данная RCPSP задача сводится к минимизации функции $t(n, \hat{k})$ для заданного $\hat{k}$ на конечном множестве ресурсов M. Опишем формулировку постановки ЗУО для дискретных переменных с конечными множествами значений.

В теории ЗУО [11] представляет собой четверку (V, D, R, C),

где $V = \{x_1, \ldots, x_n\}$ – множество переменных,

$D = \{D_1, \ldots, D_n\}$ – множество доменов переменных,

R – множество отношений различной арности над областями D,

$C = \{C_1, \ldots, C_m\}$ – множество ограничений, связывающих множество значений переменных из V посредством отношений из R.

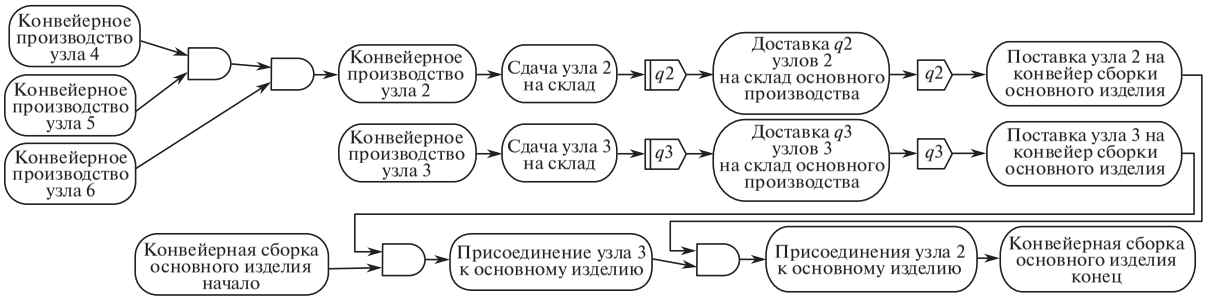

Рис. 7. Модель распределенной конвейерной сборки изделия.

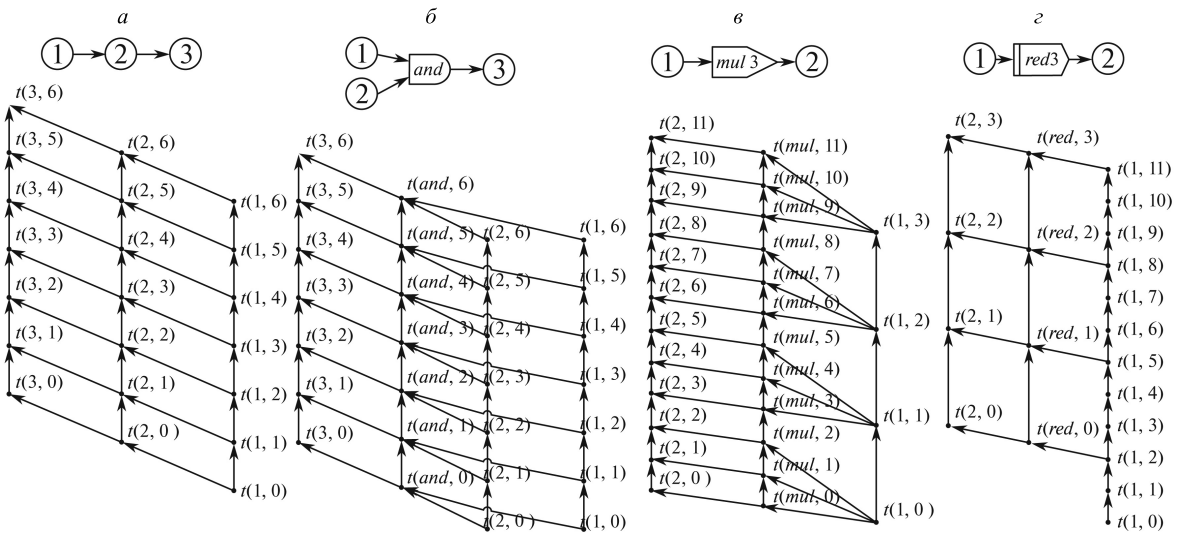

Рис. 8. Примеры графов конвейеров для конкретных значений $\hat{k}$.

Решением ЗУО является нахождение таких значений всех переменных множества V, которые удовлетворяют всем ограничениям. Решений ЗУО может быть одно или несколько.

ЗУО может быть представлена в виде сети ограничений. Было бы естественно в качестве такой сети взять граф модели конвейера, в котором вершине соответствует некоторая операция. Однако в связи с тем, что операция i выполняется k раз при наличии k заказов и в каждом случае может использоваться некоторый ресурс m_i, с операцией связано некоторое множество ресурсов. Чтобы избежать этой ситуации, сгенерируем из исходного графа модели для заданного $\hat{k}$ развернутый граф, в котором каждой вершине соответствует пара (номер операции, номер заказа) и сохраняются отношения предшествования. Построение такого графа возможно для любой модели конвейера и не представляет сложности. Простые примеры таких графов представлены на рис. 8. Сверху представлен исходный граф модели, а ниже граф для конкретного значения $\hat{k}$. Вершины графов помечены соответствующими рекурсивными функциями и ассоциируются с конкретными операциями и заказами. Если исходный граф имеет n вершин (операций), то новый граф имеет $n\hat{k}$ вершин. Легко показать, что развернутый граф так же является ациклическим. Очевидно, что в этом случае каждой вершине соответствует один ресурс, если вершина соответствует операции (тип вершины **bop** или **op**). В остальных случаях, когда вершина соответствует спусковой функции, ресурс не используется. Для отражения этого факта в задаче дополним множество ресурсов нулевым элементом $M = \{m_0, m_1, \ldots, m_r\}$.

Перенумеруем вершины развернутого графа некоторым способом. Пусть $I' = \{1, \ldots, n'\}$, $n' = n\hat{k}$, – множество новых номеров. Будем считать по умолчанию, что $i' \in I'$ обозначает номер вершины нового графа и существуют отображения $\psi : I' \to I$ и $\varphi : I' \to K$. Если в исходном графе существует предшествование по i, то в развернутом графе существует как предшествование по i, так и по k. Определим эти две функции. Функция $pk : I' \to I'$ вычисляет предшествующую по k вершину, т.е. если $i'' = pk(i')$, то $\psi(i') = \psi(i'')$, а $\varphi(i'') = \varphi(i') - 1$. Функция $pi : I' \to I'$ вычисляет предшествующую по i вершину, т.е. если $i'' = pi(i')$, то $\psi(i'') = pred(\psi(i'))$, а $\varphi(i'') = \varphi(i')$. Далее для упрощения записи в выражениях будут использоваться обе нумерации: сквозная со штрихом и двухкоординатная (i, k) без штриха в предположении, что $i' \to (i, k)$, $i = \psi(i')$ и $k = \varphi(i')$.

Перенесем приведенную постановку ЗУО в рассматриваемую область теории расписаний.

Четверка объектов ЗУО будет определена следующим образом:

1) $V = \{x_1, \ldots x_{n'}\}$ – множество дискретных переменных и для каждой из них определено множество значений.

2) $D_{i'} = D_i$ – домен переменной $x_{i'}$, связанный с соответствующей i-й вершиной исходного графа, т.е. $i' \to (i, k)$. При этом следует иметь в виду, что для каждой вершины i' – спусковой функции соответствующие переменные $x_{i'}$ будут определены на доменах $D_i = \{m_0\}$, содержащих один нулевой ресурс. Данный факт существенно сокращает пространство поиска.

3) $R \subseteq (D_1 \times D_2 \times \ldots \times D_{n'})$.

4) C – множество ограничений, которое делится на несколько групп.

Множество ограничений представляет собой в основном множество рекурсивных функций производных от рекурсивных функций операций. Модификации вызваны переходом к развернутому графу и связанной с этим заменой переменных. Данные функции порождают ограничения, обусловленные допустимыми расписаниями для каждой операции конвейера при некотором заданном распределении ресурсов для пар (операция, заказ). Последнее условие проверяет допустимость расписания с точки зрения того, чтобы ресурс в некоторый момент времени не использовался двумя операциями.

1-я группа — это множество унарных ограничений, определенных для начальных операций ($1 \leqslant i \leqslant n_0$) и построенных на основе функции (2).

$$bop(i,k) = \begin{cases} p_{ij}, \\ \qquad \textbf{если } (x_{i'} = m_j)\&(k = 0); \\ bop(i,k-1) + p_{ij}, \\ \qquad \textbf{если } (x_{i'} = x_{pk(i')})\&(x_{i'} = m_j)\&(0 < k \leqslant \hat{k}); \\ bop(i,k-1) - p_{ij} + p_{il}, \\ \qquad \textbf{если } (x_{i'} \neq x_{pk(i')})\&(x_{i'} = m_l)\&(x_{pk(i')} = m_j)\&(0 < k \leqslant \hat{k}). \end{cases}$$

2-я группа — это множество бинарных ограничений, определенных для пары вершин графа, соединенных дугой и построенных на основе функций (3), (5)–(8).

$$op(i,k) = \begin{cases} t(pred(i),k) + p_{ij}, \\ \qquad \textbf{если } (type(i) = \textbf{op})\& \\ \qquad\qquad \&(x_{i'} = m_j)\&(k = 0); \\ t(pred(i),k) + p_{ij}, \\ \qquad \textbf{если } (type(i) = \textbf{op})\&(t(pred(i),k) \geqslant t(i,k-1)\& \\ \qquad\qquad \&(x_{i'} = m_j)\&(0 < k \leqslant \hat{k}); \\ t(pred(i),k) + p_{ij}, \\ \qquad \textbf{если } (type(i) = \textbf{op})\&(t(pred(i),k) < t(i,k-1)\& \\ \qquad\qquad \&(x_{i'} \neq x_{pk(i')})\&(x_{i'} = m_j)\&(0 < k \leqslant \hat{k}); \\ t(i,k-1) + p_{ij}, \\ \qquad \textbf{если } (type(i) = \textbf{op})\&(t(pred(i),k) < t(i,k-1)\& \\ \qquad\qquad \&(x_{i'} = x_{pk(i')})\&(x_{i'} = m_j)\&(0 < k \leqslant \hat{k}). \end{cases}$$

$$mul(i,k) = t(pred(i), \lfloor k/q_i \rfloor), \ \textbf{если } (type(i) = \textbf{mul})\&(0 \leqslant k \leqslant \hat{k});$$

$$red(i,k) = t(pred(i), (k+1)q_i - 1), \ \textbf{если } (type(i) = \textbf{red})\&(0 \leqslant k \leqslant \hat{k});$$

$$get1(i,k) = t(pred(i), 2k), \\ \textbf{если } (type(i) = \textbf{get1})\&(k \bmod 2 = 0)\&(0 \leqslant k \leqslant \hat{k});$$

$$get2(i,k) = t(pred(i), 2k+1), \\ \textbf{если } (type(i) = \textbf{get2})\&(k \bmod 2 = 1)\&(0 \leqslant k \leqslant \hat{k}).$$

3-я группа — это множество ограничений, определенных для тройки вершин графа и построенных в соответствии с функциями (4) и (9).

$$and(i,k) = \max(t(pred1(i),k), t(pred2(i),k)), \quad \textbf{если } (type(i) = \textbf{and}).$$

$$put(i,k) = \begin{cases} t(pred1(i),k), \\ \qquad \textbf{если } (type(i) = \textbf{put})\&(k=0); \\ \max(put(i,k-1), t(pred2(i),k/2)), \\ \qquad \textbf{если } (type(i) = \textbf{put})\& \\ \qquad\quad \&(k \bmod 2 = 0)\&(0 < k \leqslant \hat{k}); \\ \max(put(i,k-1), t(pred1(i),(k-1)/2)), \\ \qquad \textbf{если } (type(i) = \textbf{put}) \\ \qquad\quad \&(k \bmod 2 = 1)\&(0 < k \leqslant \hat{k}). \end{cases}$$

4-я группа ограничений относится к ограничениям типа **all-different**. Она отображает тот факт, что использование ресурса в каждый момент времени возможно не более чем одной операцией. Введем следующие обозначения:
$\tau_{i'} = (t^b_{i'}, t^e_{i'})$ обозначает интервал времени от $t^b_{i'}$ до $t^e_{i'}$ $(t^b_{i'} \leqslant t^e_{i'})$, в течение которого выполняется i-й операцией k-й заказ $(i = \psi(i'),\ k = \varphi(i'))$ и используется некоторый ресурс m_j. Операция $\cap$ принимает значение истины, если временные отрезки пересекаются.

$$\tau_{i'} \cap \tau_{i''} = \begin{cases} true, \text{ если } (t^b_{i'} < t^e_{i''} \leqslant t^e_{i'}) \ or \ (t^b_{i'} \leqslant t^b_{i''} < t^e_{i'}); \\ false, \ otherwise. \end{cases}$$

В этих определениях глобальное ограничение будет выглядеть следующим образом:

$$\begin{aligned} \forall i' \forall i'' \ (1 \leqslant i', i'' \leqslant n')\&(i' \neq i'')\&(\tau_{i'} \cap \tau_{i''}) \vdash \\ \vdash ((type(i') \neq \textbf{bop})\&(type(i') \neq \textbf{op})) \vee \\ \vee (type(i'') \neq \textbf{bop})\&(type(i'') \neq \textbf{op})) \vee \\ \vee (x_{i'} \neq x_{i''}). \end{aligned}$$

Смысл условия в том, что если интервалы двух разных вершин i' и i'' пересекаются, то либо хотя бы одна из них соответствует спусковой функции, либо они используют разные ресурсы.

Решением ЗУО будет нахождение таких значений переменных $x_{i'}$, при которых все перечисленные условия будут удовлетворены, и очевидно, что в данной постановке ЗУО всегда будет иметь некоторое решение. Для любого допустимого распределения ресурсов можно построить расписание. В статье требуется оптимальное расписание. Пусть T_{opt} – время выполнения оптимального расписания. Будем полагать, что решение квазиоптимальное с точностью до некоторой, наперед заданной величины Δ, если найденное решение T такое, что $T - T_{opt} \leqslant \Delta$. Вычисление квазиоптимального значения времени выполнения расписания можно осуществить с помощью следующего известного алгоритма:

1) Выбрать некоторое множество значений V, равное V_0.
Вычислить значение $T_{\max} = t(n', \hat{k})$ для V_0.
Положить $T_{\min} = 0$.

2) Если $T_{\max} - T_{\min} \leqslant \Delta$, то задача решена и $T_{\max}$ является решением, а соответствующее данному решению множество V является искомым распределением ресурсов.

3) Вычислить значение $T = T_{\min} + (T_{\max} - T_{\min})/2$.

4) Ввести в ЗУО дополнительное ограничение: $t(n', \hat{k}) \leqslant T$.

5) Если ЗУО с дополнительным условием не будет иметь решение, то положить $T_{\min} = T$ и перейти к п. 3.

6) Если ЗУО с дополнительным условием будет иметь решение, то положить $T_{\max} = T$ и перейти к п. 2.

Конец алгоритма.

Найденное значение $T_{\max}$ является квазиминимальным, но квазиоптимальность этого значения следует доказать. Суть проблемы состоит в том, что рекурсивные функции на локальном уровне строят оптимальное расписание (реализуют жадный алгоритм), и следует доказать, что при фиксированном распределении ресурсов локальная оптимизация является и глобальной.

Утверждение 1. Для любого допустимого значения $V = \{x_1, \ldots, x_{n'}\}$ *рекурсивная функция* $t(i, k)$ *вычисляет одно из оптимальных значений для* $1 \leqslant i \leqslant n$ *и* $0 \leqslant k \leqslant \hat{k}$.

Доказательство осуществим методом математической индукции.

Для системности изложения напомним формулировку принципа математической индукции. Некоторое утверждение $P(i)$, зависящее от натурального параметра i, считается доказанным, если:

1) установлено, что $P(1)$ верно;

2) для любого $1 \leqslant i < n'$ из предположения, что $P(i)$ верно, следует, что $P(i+1)$ верно.

Применим данный метод к развернутому графу. Из теории графов известно [17], что ациклический граф является строго частично упорядоченным. Существуют алгоритмы построения для строго частично упорядоченного графа некоторого линейного порядка. Неформально линейный порядок — это когда все вершины графа расположены в строку и все дуги направлены слева направо. Пример построения линейного порядка из строго частичного показан на рис. 9, где *а*) — это исходный ациклический граф, а *б*) — один из возможных линейных порядков. В данной статье устроит любой такой алгоритм с условием, что первые n_0 вершин исходного графа так и останутся первыми, хотя можно и в другом порядке, а вершина n останется последней. При вычислении рекурсивной функции в соответствии с линейным порядком ее аргументы уже будут вычислены.

Метод математической индукции применим следующим образом.

Шаг 1. Исходя из определения рекурсивного конвейера и способа упорядочивания вершин исходного графа очевидно, что все начальные вершины

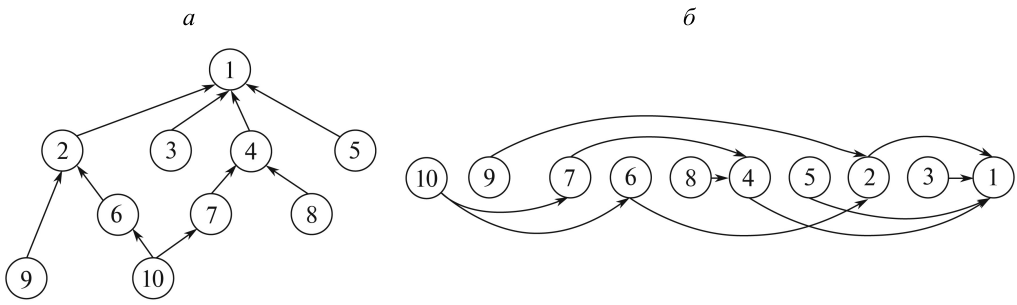

Рис. 9. Пример линейного порядка графа.

исходного графа являются операциями (не функциями) и являются производными для начальных вершин развернутого графа. Если в исходном графе n_0 начальных вершин, то и в развернутом графе тоже n_0 начальных вершин и для каждой начальной вершины i в исходном графе существует вершина $(0, i)$ в развернутом графе, которой не предшествует никакая другая вершина. Если $1 \leqslant i \leqslant n_0$ – некоторая начальная вершина исходного графа, то i' такая, что $\psi(i') = i$ и $\varphi(i') = 0$ является начальной вершиной развернутого графа.

Таким образом, для каждой начальной вершины $i' \to (i, k)$ развернутого графа справедливы следующие утверждения:

$$type(i) = \mathbf{bop},$$
$$1 \leqslant \psi(i') \leqslant n_0 \quad \text{и} \quad \varphi(i') = 0, \quad \text{т.е.} \quad (1 \leqslant i \leqslant n_0)\&(k = 0).$$

В этом случае в соответствии с функцией (1)

$$t(i, 0) = bop(i, 0) = p_{ij}, \quad \text{если} \quad x_{i'} = m_j.$$

Таким образом, для конкретного значения $x_{i'}$ значение $t(i, 0)$ единственно и, следовательно, оно оптимально. Утверждение верно. Поскольку данное утверждение верно для любой из n_0 начальных операций, далее эти варианты рассматривать не будем.

Шаг 2. Допустим, что для вершины любого номера $n_0 < i' < n'$ утверждение верно. Докажем, что в этом случае оно верно и для вершины с номером $i'' = i' + 1$. В соответствии с (1) возможны 8 вариантов.

Рассмотрим варианты 1 и 2, когда

$$type(i'') = \mathbf{bop} \quad \text{и}$$
$$type(i'') = \mathbf{op}.$$

Вариант 1. Выполняется начальная операция над ненулевым заказом, т.е. $i'' \to (i, k)$ и $(1 \leqslant i \leqslant n_0)\&(k > 0)$. Если $x_{i''} = x_{pk(i'')}$, то операция i при выполнении k-го и $(k-1)$-го заказов использует один ресурс.

В этом случае время завершения вычисляется по формуле (2,а) (диаграмма на рис. 10,*а*). Если же используются разные ресурсы, то выполнение операции i можно осуществить одновременно над заказами $(k-1)$ и k. Завершение выполнения вычисляется по формуле (2,б) (диаграмма на рис. 10,*б*). Длительности операций могут не совпадать. Учитывая, что время выполне-

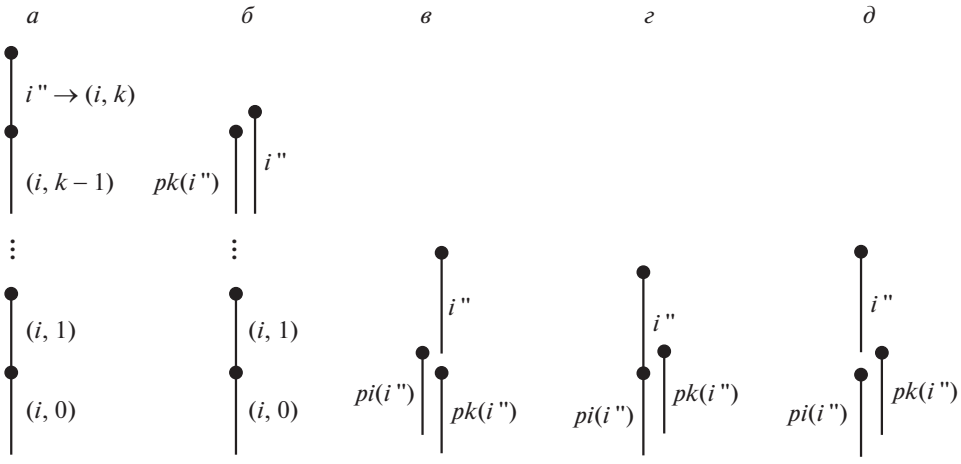

Рис. 10. Варианты фрагментов временных диаграмм.

ния операцией i $(k-1)$-го заказа оптимально, время выполнения i k-го заказа также оптимально. Вариант рассмотрен.

Вариант 2. В этом случае выполняется неначальная операция. Если $k = 0$, то время завершения вычисляется по формуле (3,а). Если $k > 0$, то время завершения вычисляется по вариантам (3,б)–(3,г). Соответствующие примеры временных диаграмм приведены на рис. 10,*c–e*. Во всех вариантах новое расписание с добавлением вершины i'' будет оптимальным, если предшествующее расписание было оптимальным.

Варианты 3–8. Во всех данных вариантах вершины соответствуют спусковым функциям. В этом случае не производится распределение ресурсов и итоговое расписание строится единственным возможным способом исходя из отношения временного предшествования, определяемого соответствующей рекурсивной функцией.

Утверждение доказано.

Временная сложность алгоритма равна

$$O\left(\log_2\left(\frac{T_0}{\Delta}\right)\prod_{i=1}^{nk} \mid D_i \mid\right),$$

где T_0 – значение времени выполнения расписания для некоторого произвольного множества значений V_0, Δ – некоторая наперед заданная константа точности вычисления результата, а $\mid D_i \mid = 1$ для спусковых функций.

5. Заключение

В данной статье осуществлена теоретическая постановка задачи. Впервые для исследования конвейерной модели используется такой инструмент, как Constraint Programming. Применение данного метода на практике является дальнейшим направлением работы авторов статьи.

В настоящее время существует большое количество коммерческих и свободного использования систем *программирования в ограничениях*. Обзор таких систем можно посмотреть в [2].

Изучение вычислительной сложности решения ЗУО представляется фундаментальной проблемой. В [18] было показано, что класс всех ЗУО является NP-трудным, так что вряд ли существуют эффективные (полиномиальные) алгоритмы общего назначения для решения всех видов ЗУО. Классы легко разрешимых ЗУО, которые могут быть решены за полиномиальное время, обычно описываются или деревьями и древовидными графовыми структурами [19, 20], или определенной комбинацией алгебраических операторов [21]. Рассматриваемая в статье задача базируется на дереве ограничений (исходный ациклический граф и расширенный граф), что внушает оптимизм по поводу существования эффективного алгоритма решения поставленной задачи. Построение расписания для конвейера вычислением суперпозиции рекурсивных функций предполагает для решения ЗУО вариант *поиска с возвратами* (backtracking) в различных модификациях [2]. Важно отметить, что введение в модель конвейера спусковых функций нисколько не увеличивает алгоритмическую сложность решения задачи.

Использование данного метода дает два важных преимущества.

Первое. Метод оптимизации рекурсивного конвейера, рассмотренный в статье [22], сводится к задаче целочисленного линейного программирования и рассматривает строго определенный набор рекурсивных функций. Введение новых функций требует дополнительного рассмотрения корректности применения метода. Подход, используемый в данной статье, хотя и рассматривает конкретный набор функций, однако никаких ограничений, кроме вычислимости, к ним не предъявляет. Это позволяет вводить в рассмотрение новые рекурсивные функции без обсуждения вопроса корректности применения метода.

Второе. Предложенный метод позволяет распределять ресурсы без их закрепления за отдельными операциями. Данный подход увеличивает размерность задачи, но открывает возможности для построения более эффективного расписания.

СПИСОК ЛИТЕРАТУРЫ

1. *Лазарев А.А., Гафаров Е.Р.* Теория расписаний. Задачи и алгоритмы. М.: Изд-во МГУ, 2011.
2. *Rekiek B., Dolgui A., Delchambre A., Bratcu A.* State of art of optimization methods for assembly line design // Annual Reviews in Control. 2002. V. 26. P. 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. 26. No. 4. P. 637-0670.
4. *Helgeson W.B., Salveson M.E., Smith W.W.* How to balance an assembly line / Technical Report, Carr Press. 1954. New Caraan, Conn.
5. *Bowman E.H.* Assembly line balancing by linear programming // Oper. Res. 1960. No. 8(3). P. 3850–389.

6. *Salveson M.E.* The assembly line balancing problem // J. Indust. Engineer. 1955. No. 6. P. 18–25.
7. *Neumann K., Schwindt C., Zimmermann J.* Recent results on resource-constrained project scheduling with time windows: Models, solution methods, and applications // Central Eur. J. Oper. Res. 2002. V. 10. 2. P. 113–148.
8. *Drexl A., Kimms A.* Optimization guided lower and upper bounds for the resource investment problem // J. Oper. Res. Society. 2001. V. 52. No. 3. P. 340–351.
9. *Ranjbar M., Kianfar F., Shadrokh S.* Solving the resource availability cost problem in project scheduling by path relinking and genetic algorithm // Applied Mathematics and Computation. 2008. V. 196. No. 2. P. 879–888.
10. *Yamashita D.S., Armentano V.A., Laguna M.* Robust optimization models for project scheduling with resource availability cost // Journal of Scheduling. 2007. V. 10. No. 1. P. 67–76.
11. *Щербина О.А.* Удовлетворение ограничений и программирование в ограничениях// Интеллектуальные системы. 2011. Т. 15. № 1–4. С. 53–170.
12. *Apt K.R.* Principles of Constraint Programming. New York: Cambridge University Press, 2003.
13. *Нариньяни А.С., Седреева Г.О., Седреев С.В., Фролов С.А.* TimeEX/Windows – новое поколение недоопределенной технологии календарного планирования / Проблемы представления и обработки не полностью определенных знаний. Под ред. И.Е. Швецова. Москва; Новосибирск, 1996. С. 101–116.
14. *Куприянов Б.В.* Рекурсивные конвейерные процессы – основные свойства и характеристики // Экономика, статистика и информатика. Вестн. УМО. 2015. № 1. С. 163–170.
15. *Куприянов Б.В.* Применение модели конвейерных процессов рекурсивного типа для решения прикладных задач // Экономика, статистика и информатика. Вестник УМО. 2014. № 5. С. 170–179.
16. *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. P. 435–449.
17. *Оре О.* Теория графов. М.: Наука, 1980.
18. *Mackworth A.K.* Consistency in networks of relations // Artificial Intelligence. 1977. V. 8. No. 1. P. 99–118.
19. *Dechter R., Pearl J.* Tree clustering for constraint networks // Artificial Intelligence. 1989. V. 38. No. 3. P. 353–366.
20. *Freuder E.C.* Backtrack-free and backtrack-bounded search / Search in Artificial Intelligence / L. Kanal, V. Kumar (eds.). Springer-Verlag, 1988. P. 343–369.
21. *Jeavons P.G., Cohen D.A., Gyssens M.* Closure properties of constraints // Journal of ACM. 1997. V. 44. P. 527–548.
22. *Куприянов Б.В.* Оценка и оптимизация производительности рекурсивного конвейера // АиТ. 2020. № 5. С. 6–25.
    Kupriyanov B.V. Estimation and Optimization of the Efficiency of a Recursive Conveyor // Autom. Remote Control. 2020. V. 81. P. 775–790.

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

Поступила в редакцию 25.01.2021
После доработки 21.06.2021

Принята к публикации 30.06.2021

© 2021 г. А.А. САРАТОВ, канд. тех. наук (sapfor58@.gmail.com)
(ООО "Интерактивные системы автоматизации проектирования", Тула)

ЖАДНЫЙ АЛГОРИТМ РЕШЕНИЯ КЛАССИЧЕСКОЙ NP-ТРУДНОЙ ЗАДАЧИ ТЕОРИИ РАСПИСАНИЙ МИНИМИЗАЦИИ СУММАРНОГО ЗАПАЗДЫВАНИЯ

Представлен эффективный метод решения классической NP-трудной задачи теории расписаний для одного прибора минимизации суммарного запаздывания $1||\sum T_j$. Предлагается алгоритм решения задачи, основанный на декомпозиции исходной задачи на подзадачи своевременного обслуживания каждого из требований и размещении в конец расписаний тех из них, прирост запаздывания которых в наибольшей степени компенсируется сокращением запаздываний предшествующих требований. Трудоемкость алгоритма не превышает $O\left(n^2\right)$ операций, где n – количество требований.

Ключевые слова: теория расписаний, один прибор, минимизация суммарного запаздывания, жадные алгоритмы.

DOI: 10.31857/S0005231021110064

Рассматривается классическая NP-трудная в обычном смысле задача теории расписания, в которой необходимо обслужить множество требований $N = \{1, \ldots, , n\}$ на одном приборе [1]. Прерывания при обслуживании и обслуживание более одного требования в любой момент времени запрещены. Для требования $j \in N$ заданы продолжительность обслуживания $p_j > 0$ и директивный срок окончания обслуживания d_j. Все требования поступают на обслуживание одновременно в момент времени t_0, с которого прибор готов начать обслуживание этих требований. Расписание обслуживания требований строится с момента времени t_0 и однозначно задается перестановкой элементов множества N.

Требуется построить расписание π^* обслуживания требований множества N, при котором достигается минимум функции

$$F(\pi) = \sum_{j=1}^{n} \max\{0, c_j(\pi) - d_j\},$$

где $c_j(\pi)$ – момент завершения обслуживания требования j при расписании π. Пусть $\pi = (j_1, \ldots, j_n)$, тогда $c_{j_1}(\pi) = t_0 + p_{j_1}$ и $c_{j_k}(\pi) = c_{j_{k-1}} + p_{j_k}$ для $k = 2, 3, \ldots, n$.

Величина $T_j(\pi) = \max\{0, c_j(\pi) - d_j\}$ называется запаздыванием требования j при расписании π, а $F(\pi)$ – суммарным запаздыванием требований при расписании π. Если обслуживание требования i предшествует обслуживанию требования j, то для этого будем использовать запись $(i \to j)_\pi$. Если

обслуживание требования i переносится на более ранний или поздний срок, то будем называть это смещением влево или вправо.

К настоящему времени опубликовано большое число работ, посвященных подходам к решению данной задачи [1, 2]. Все предложенные решения не позволяют получить точное решение задачи за приемлемое для практического использования время. Здесь предлагается метод синтеза оптимального расписания, основанный на декомпозиции исходной задачи на подзадачи своевременного обслуживания каждого из требований и размещении в конец расписаний тех из них, прирост запаздывания которых в наибольшей степени компенсируется сокращением запаздываний предшествующих требований.

Поскольку в оптимальном расписании π^* любое перемещение требования j и любая парная перестановка требований i, j не могут привести к уменьшению $F(\pi)$, то для всех $i, j \in \pi^*$ должно выполняться условие:

$$(1)\quad \begin{cases} \nabla T'_j(\pi) \geq \sum\limits_{i=j+1}^{j+k} \nabla T'_i(\pi), & j > i \\ \nabla T''_j(\pi) \leq \sum\limits_{i=j-k}^{j-1} \nabla T''_i(\pi), & j < i \end{cases} \wedge \begin{cases} \nabla T_{ij}(\pi) \geq \sum\limits_{v=i+1}^{j-1} \nabla T'_v(\pi), & p_i > p_j \\ \nabla T_{ji}(\pi) \leq \sum\limits_{v=i+1}^{j-1} \nabla T''_v(\pi), & p_i < p_j, \end{cases}$$

где

$\nabla T'_j(\pi) \nabla T''_j(\pi)$ — изменение запаздывания требования j при перемещении j в очереди на k позиций соответственно вправо и влево;

$\sum_{j+1}^{j+k} \nabla T'_i(\pi)$ — суммарное уменьшение запаздывания требований $i = \{j+1, \ldots, j+k\} \in N$ при их перемещении в очереди на k позиций влево;

$$(2)\quad \sum_{j+1}^{j+k} \nabla T'_i(\pi) = \sum_{j+1}^{j+k} \left\{ \max\left[0, \left(t_0 + \sum_{m=0}^{i} p_m - d_i\right)\right] - \max\left[0, \left(t_0 + \sum_{m=0}^{i} p_m - p_j - d_i\right)\right] \right\};$$

$\sum_{j-k}^{j-1} \nabla T''_i(\pi)$ — суммарное увеличение запаздывания требований $i = \{j-k, \ldots, j-1\} \in N$ при их перемещении в очереди на k позиций вправо;

$$(3)\quad \sum_{j-k}^{j-1} \nabla T''_i(\pi) = \sum_{j-k}^{j-1} \left\{ \max\left[0, \left(t_0 + \sum_{m=0}^{i} p_m + p_j - d_i\right)\right] - \max\left[0, \left(t_0 + \sum_{m=0}^{i} p_m - d_i\right)\right] \right\};$$

$\nabla T_{ij}(\pi)$ — увеличение запаздывания требования i при замещении j;

$\nabla T_{ji}(\pi)$ — уменьшение запаздывания требования j при замещении i;

$\sum_{i+1}^{j-1} {}_{\nabla}T'_v(\pi)$ — суммарное уменьшение запаздывания требований $v = \{i+1, \ldots, j-1\}$ при перестановке требований i, j и $p_i > p_j$;

$\sum_{i+1}^{j-1} {}_{\nabla}T''_v(\pi)$ — суммарное увеличение запаздывания требований $v = \{i+1, \ldots, j-1\}$ при перестановке требований i, j и $p_i < p_j$.

Множества перемещаемых требований формул (2), (3) будем называть частичными расписаниями $\pi^{\wedge}$ и $\pi^{\wedge\wedge}$;

$$\pi^{\wedge} = \{j+1, \ldots, j+k\} \in N, \quad \pi^{\wedge\wedge} = \{j-k, \ldots, j-1\} \in N.$$

Выражение (1) при $k = 1, \ldots, n-1$, является эффективным инструментом оценки оптимальности построенного расписания, ибо вычисляется за $0\left(n^2\right)$ операций.

Для обеспечения эффективного процесса синтеза π^* на основе формулы (1) необходим быстрый алгоритм формирования частичных расписаний $\pi^{\wedge}$ и $\pi^{\wedge\wedge}$, эквивалентных по соотношению суммарных смещений $\sum_{i+1}^{i+k} {}_{\nabla}T'_i(\pi)$, $\sum_{i-k}^{i-1} {}_{\nabla}T''_i(\pi)$ с суммарными смещениями в расписании π^*. Такая эквивалентность может быть достигнута в частичных расписаниях $\pi^{\wedge}$, $\pi^{\wedge\wedge}$, отвечающих (1) при $k = 1$. Истинность выражения (1) при $k = 1$ отражает известный факт, что в оптимальном расписании перестановка двух смежных требований не может уменьшить их суммарного запаздывания и является необходимым условием для проверки оптимальности π^*. Очередность обслуживания требований $j \in \pi^{\wedge}$ определяется соотношениями их параметров d_j, p_j, поэтому расписания, отвечающие условию (1) при $k = 1$, будем называть локально оптимальными.

Рассмотрим метод построения расписания π^* начиная от последнего требования к первому, используя процедуры формирования $\pi^{\wedge}$.

Алгоритм A построения $\pi^{\wedge}$ имеет вид.

1. Принимаем время старта $t = t_0$.
2. Моделируем попарно перестановки требований $i \to j$, $j \to i$ и вычисляем сумму ${}_{\nabla}T_{ij} = T_i(\pi) + T_j(\pi)$. Если при $i \to j$ суммарное смещение меньше, чем при $j \to i$, то i помечается, как приоритетное требование i^* и продолжает участвовать в перестановках, а j исключается. Если суммарные смещения для $i \to j$ и $j \to i$ равны, то помечается требование с меньшим d_j.
3. Исключаем i^* из N и помещаем в N^*, добавляя список справа.
4. Принимаем время старта $t = t_0 + p_i$.
5. Повторяем шаги 2–4, пока $N \neq \varnothing$.

Для канонических DL примеров [1] алгоритм A достаточен для синтеза оптимального расписания. Рассмотрим работу алгоритма A на одном из таких примеров (рис. 1).

В канонических LG расписаниях [1] первые $n/2-1$ требований не опаздывают, и при построении таких расписаний можно использовать соотношения прироста запаздываний от перемещения требований вправо с уменьшением суммарных запаздываний требований, перемещаемых влево на освоившиеся места (2).

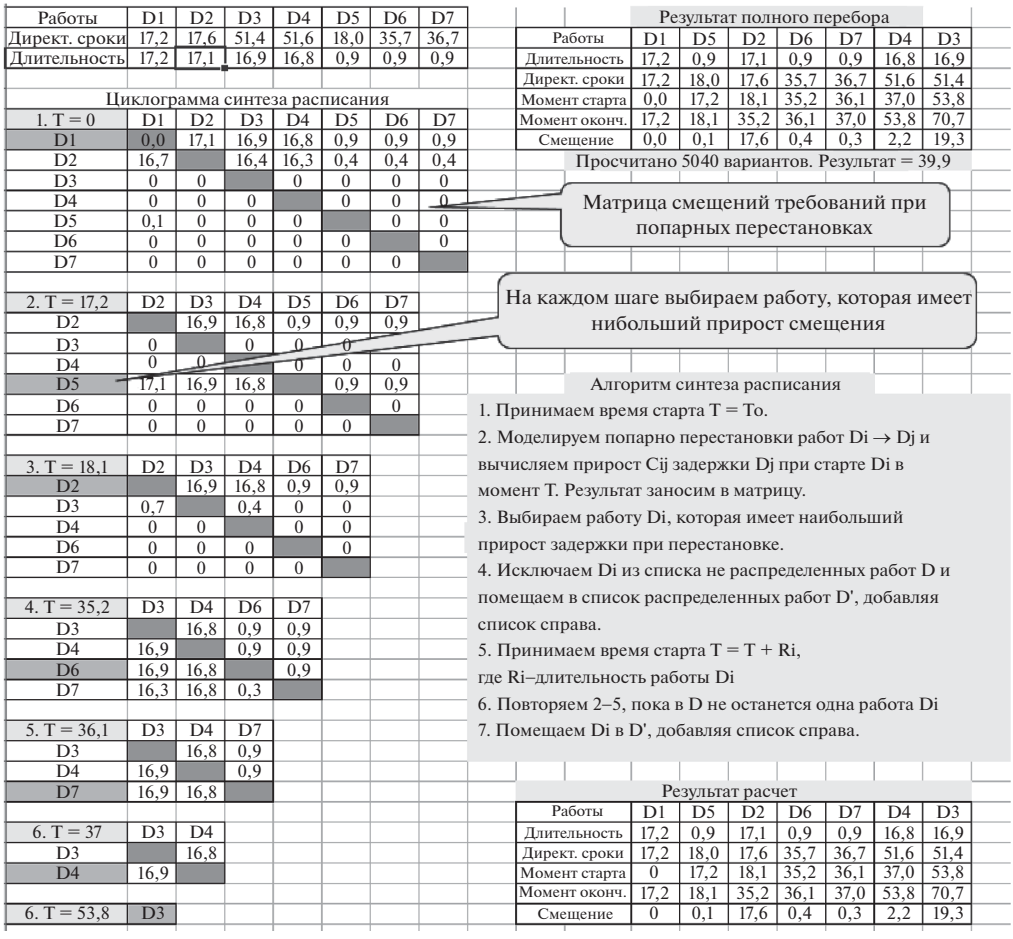

Рис. 1. Решение канонического DL-примера.

Обозначим как π_j расписание, состоящее из $\pi^\wedge$ и следующего за ним требования j, а суммарное запаздывание требований π_j — как $F(\pi_j)$.

Если принять условие (1) достаточным для проверки оптимальности $\pi^\wedge$, то в расписании $\pi_j = (\pi^\wedge, j)$ с минимальным суммарным смещением $F(\pi_j)$ требование $j \in \pi_j$ будет последним в оптимальном расписании π^*. Исходя из этого, алгоритм построения π^* имеет вид:

1. Для каждого требования j формируем расписание $\pi_j = (\pi^\wedge, j)$, в котором j размещается в конце очереди, а остальные работы упорядочены согласно алгоритму A. Вычисляем суммарные смещения $F(\pi)$.
2. Выбираем расписание $\pi_j = (\pi^\wedge, j)$ с минимальным $F(\pi)$.
3. Исключаем j из N и помещаем в список последних требований $N^* \in \pi^*$, добавляя список слева.
4. Повторяем шаги 1, 2, 3, пока в N останутся только успевающие работы.
5. Объединяем N и N^*.
6. Конец.

1	02	03	05	04	01	
p	20,00	18,10	16,00	18,00	20,10	
d	52,25	53,70	54,25	53,75	52,10	
t	0	20,00	38,10	54,10	72,10	
t'	20,00	38,10	54,10	72,10	92,20	
T	0	0	0	18,35	40,10	58,45

7	01	04	05	03	
p	20,10	18,00	16,00	18,10	
d	52,10	53,75	54,25	53,70	
t	0	20,10	38,10	54,10	
t'	20,10	38,10	54,10	72,20	
T	0	0	0	18,50	18,50

2	01	03	05	04	02	
p	20,10	18,10	16,00	18,00	20,00	
d	52,10	53,70	54,25	53,75	52,25	
t	0	20,10	38,20	54,20	72,20	
t'	20,10	38,20	54,20	72,20	92,20	
T	0	0	0	18,45	39.95	58,40

8	01	03	05	04	
p	20,10	18,10	16,00	18,00	
d	52,10	53,70	54,25	53,75	
t	0	20,10	38,20	54,20	
t'	20,10	38,20	54,20	72,20	
T	0	0	0	18,45	18,45

3	D1	D2	D5	D4	D3	
p	20,10	20,00	16,00	18,00	18,10	
d	52,10	52,25	54,25	53,75	53,70	
t	0,00	20,10	40,10	56,10	74,10	
t'	20,10	40,10	56,10	74,10	92,20	
T	0,00	0,00	1,85	20,35	38,50	60,70

9	D1	D3	D4	D5	
p	20,10	18,10	18,00	16,00	
d	52,10	53,70	53,75	54,25	
t	0,00	20,10	38,20	56,20	
t'	20,10	38,20	56,20	72,20	
T	0,00	0,00	2,45	17,95	20,40

4	D1	D2	D5	D3	D4	
p	20,10	20,00	16,00	18,10	18,00	
d	52,10	52,25	54,25	53,70	53,75	
t	0,00	20,10	40,10	56,10	74,20	
t'	20,10	40,10	56,10	74,20	92,20	
T	0,00	0,00	1,85	20,50	38,45	60,80

10	D3	D5	D1	
p	18,10	16,00	20,10	
d	53,70	54,25	52,10	
t	0,00	18,10	34,10	
t'	18,10	34,10	54,20	
T	0,00	0,00	2,10	2,10

5	D1	D2	D4	D3	D4	
p	20,10	20,00	18,00	18,10	16,00	
d	52,10	52,25	53,75	53,70	54,25	
t	0,00	20,10	40,10	58,10	76,20	
t'	20,10	40,10	58,10	76,20	92,20	
T	0,00	0,00	4,35	22,50	37,95	64,80

11	D1	D5	D3	
p	20,10	16,00	18,10	
d	52,10	54,25	53,70	
t	0,00	20,10	36,10	
t'	20,10	36,10	54,20	
T	0,00	0,00	0,50	0,50

6	D3	D4	D5	D1	
p	18,10	18,00	16,00	20,10	
d	53,70	53,75	54,25	52,10	
t	0,00	18,10	36,10	52,10	
t'	18,10	36,10	52,10	72,20	
T	0,00	0,00	0,00	20,10	20,10

12	D1	D3	D5	
p	20,10	18,10	16,00	
d	52,10	53,70	54,25	
t	0,00	20,10	38,20	
t'	20,10	38,20	54,20	
T	0,00	0,00	0,00	0,00

Итог	D1	D3	D5	D4	D2
p	20,10	18,10	16,00	18,00	20,00
d	52,10	53,70	54,25	53,75	52,25
t	0,00	20,10	38,20	54,20	72,20
t'	20,10	38,20	54,20	72,20	92,20
T	0,00	0,00	0,00	18,45	39,95

Рис. 2. Синтез оптимального расписания для LG примера.

Рассмотрим пример. Пусть имеется 5 требований $N = \{D_1, D_2, D_3, D_4, D_5\}$ с параметрами: $p_1 = 20{,}1$, $p_2 = 20$, $p_3 = 18{,}1$, $p_4 = 18$, $p_5 = 16$, $d_1 = 52{,}1$, $d_2 = 52{,}25$, $d_3 = 53{,}7$, $d_4 = 53{,}75$, $d_5 = 54{,}25$. Необходимо составить расписание с минимальным суммарным запаздыванием (T_i) требований. Процедура синтеза расписания (рис. 2) будет включать 12 шагов:

1. Переносим D_1 в конец расписания, а остальные требования выстраиваем по результатам перестановок $i \to j$, $j \to i$ (алгоритм A). Вычисляем суммарное запаздывание $T = T_4 + T_1 = 18{,}35 + 40{,}1 = 58{,}45$.
2. Выполняем п. 1 для $D_2, \ldots, D_5$ (шаги $2, \ldots, 5$).
3. Выбираем требование (D_2), с наименьшим $T = 58{,}4$, и переносим D_2 из N в N^*.
4. Выполняем пп. 1–3 для $N = \{D_1, D_3, D_4, D_5\}$ (шаги $6, \ldots, 9$). Переносим D_4 из N в N^*.
5. Выполняем пп. 1–3 для $N = \{D_1, D_3, D_5\}$ (шаги $10, \ldots, 12$). Переносим D_5 из N в N^*.
6. Оставшиеся требования $N = \{D_1, D_3\}$ имеют $T = 0$. Переносим D_1, D_3 из N в N^*.
7. Конец.

Полученное расписание $N^* = \{D_1, D_3, D_5, D_4, D_2\}$ имеет минимальное суммарное запаздывание $T = T_1 + T_3 + T_5 + T_4 + T_2 = 18{,}45 + 39{,}95 = 58{,}4$.

Испытания предложенного метода проводились на канонических DL и LG примерах с количеством требований от 5 до 26. Во всех случаях были получены оптимальные решения.

СПИСОК ЛИТЕРАТУРЫ

1. *Лазарев А.А., Гафаров Е.Р.* Теория расписаний. Задачи и алгоритмы. М.: Изд-во МГУ, 2011, 222 с.
2. *Лазарев А.А., Архипов Д.И.* Оценка абсолютной погрешности и полиномиальной разрешимости для классической NP-трудной задачи теории расписаний // Доклады АН. 2018. Т. 480. № 5. С. 523—527.
3. *Саратов А.А.* Согласование производственных циклов методом взаимных штрафов // Автоматизация процессов управления. 2019. № 1. С. 66–73.

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

Поступила в редакцию 27.01.2021

После доработки 28.05.2021

Принята к публикации 30.06.2021

© 2021 г. Ю.В. СИДЕЛЬНИКОВ, д-р техн. наук (sidelnikovy@mail.ru)
(Институт проблем управления им. В.А. Трапезникова РАН, Москва;
Московский авиационный институт
(национальный исследовательский университет))

РАСШИРЕНИЕ ВОЗМОЖНОСТЕЙ МЕТРИЧЕСКОГО ПОДХОДА НА ОСНОВЕ ТЕОРИИ СРЕДНИХ И ТЕОРИИ ОШИБОК

Предлагается расширить возможности метрического подхода для решения специальных задач, например в теории расписаний ослаблением требований к аксиомам метрики или введением вероятностных мер близости. Рассматриваются результаты автора на стыке теории средних и такой области исследования, как показатели экспертных ошибок, которые аксиоматически заданы. Усилен результат, полученный акад. А.Н. Колмогоровым при рассмотрении им системы аксиом для вывода аналитической формулы ассоциативной средней.

Ключевые слова: требования к аксиомам метрики, показатели экспертных оценок, теорема Колмогорова и ее усиление.

DOI: 10.31857/S0005231021110076

1. Введение

Метрики достаточно широко используются в фундаментальных и прикладных исследованиях. Например, в теории расписаний, которая является фундаментальной областью дискретной оптимизации [1].

Обзоры, в рамках которых рассматриваются всевозможные меры близости, публиковались достаточно давно. Например, обзор мер близости в пространстве множеств был опубликован Г.В. Раушенбахом в 1982 г. [2]. В настоящее время появляются исследования, где рассматриваются принципиально новые меры близости (сходства), например вероятностные меры близости, которые рассмотрены Д.З. Уздиным в монографии [3]. Полагаем, что метрический подход для решения разнообразных задач может получить новый импульс при использовании не только мер близости, но и функций, которые играют роль метрики, однако метриками в классическом понимании не являются, так как их аксиомы “ослаблены”, а также не дискретных, а вероятностных мер близости. Такие публикации имеются, но их существенно меньше.

В статье автор рассматривает меру близости как однозначную, неотрицательную действительную функцию $\rho(x, y)$, определенную для любых x и y на X и удовлетворяющую трем стандартным условиям (аксиомам) [4, с. 48].

Цель статьи — рассмотреть варианты перспективных направлений в фундаментальных и прикладных исследованиях, связанные с использованием ве-

роятностного подхода для мер близости и ослаблением требований к аксиоматике метрик, и получить в их рамках новые результаты.

2. Возможные варианты перспективных направлений

Среди первых трех, по числу аксиом меры близости, вариантов перспективных направлений рассмотрим лишь те, которые связаны с ослаблением требований к аксиоматике метрики. Причем если первое и второе лишь обозначим, то в третьем направлении попытаемся не только его предложить, но и получить в его рамках новые результаты.

В качестве первого направления предложим использовать для анализа информации не только метрики, но и псевдометрики[1]. В этом случае ослабляется первая аксиома метрики (рефлексивность) таким образом, что не требуется выполнение условия: чтобы из $E(x,y)=0$ следовало бы $x=y$ [7, с. 740–741].

Исходим из определения псевдометрического пространства как множества X, наделенного псевдометрикой $E(x,y)$. При этом любым двум элементам $x,y\in X$ ставится в соответствие действительное число E такое, что:

первая аксиома: если $x=y$, то $E(x,y)=0$,

вторая аксиома: неравенство треугольника. Когда для любой тройки $x,y,z\in X$

$$E(x,y)\le E(x,z)+E(z,y).$$

Легко показать, что псевдометрика симметрична и положительно полуопределена, т.е. $E(x,y)\ge 0$ (см., например, [5, с. 31]).

В качестве второго направления предложим рассмотреть вариант ослабления аксиомы неравенства треугольника, как, например, в диссертации Г.В. Иофиной [6]. Эта диссертация направлена на поиск способов измерения расстояний на объектах, задаваемых наборами порядковых признаков. Г.В. Иофина полагает, что “иногда полезно рассматривать функции на значениях признаков, которые играют роль метрики, однако метриками в классическом понимании не являются”. Цель этой диссертации — поиск и использование оптимальных функций расстояний, удовлетворяющих всем аксиомам метрик или полуметрик кроме неравенства треугольника. Г.В. Иофина рассматривает задачу классификации с двумя классами K_1 и K_2. При этом дано множество объектов $\{S^1,\dots,S^{m1}\}$ из класса K_1 и $S^{m1+1},\dots,S^{m1+m2}$ из класса K_2. Каждый объект принадлежит признаковому пространству размерности n. Значения признаков принадлежат конечному множеству $E^N=\{0,1,\dots,N-1\}$, на котором задано отношение порядка: $0\le 1\le 2\le\dots\le N-1$. На каждом признаке задана своя функция расстояния, которая удовлетворяет всем аксиомам метрики кроме неравенства тре-

[1] В более ранних публикациях на русском языке данное понятие именовалось как “квазирасстояние” [5, с. 31]. Встречаются публикации, где данное понятие называется полуметрикой [6].

угольника, т.е. функция $p(x, y) : E^N \times E^N \to E^M$ удовлетворяет следующим условиям:

1. $p(x, y) = 0 \Leftrightarrow x = y$.

2. $p(x, y) = p(y, x)$.

3. Если $x \geq y$, то $p(x, z) \geq p(y, z)$ и $p(x, z) \geq p(x, y)$, для любых $z \in E^N$, $z \leq y$.

Таким образом, Г.В. Иофина использует условие, замещающее неравенство треугольника на более слабое — расстояние между дальними значениями не меньше, чем между ближними (условие порядка).

При этом требуется, чтобы выбранная мера близости была оптимальна для различных задач, главным образом для задач распознавания. Кроме того, Г.В. Иофина обоснованно полагает, что результат работы алгоритмов классификации и кластеризации сильно зависит от метрик, заданных на объектах. И, значит, особо важно выбрать правильную функцию близости в задачах кластеризации [8, 9].

Третье перспективное направление. В данном случае рассматривается вариант, когда допускается асимметричность метрики и при этом ослабляются тем или иным способом первая и (или) третья аксиомы метрики. В таком случае можно использовать асимметричные меры, например для числового случая показатель ошибки $E1(x, y) = |x - y|/|y|$. Обычно все такие меры называют показателями ошибки. Использование таких асимметричных мер возможно, когда на множестве можно задать тернарное отношение “между”, и полезно, когда нужно рассматривать качественно другие ситуации. Например, вы пришли за одну минуту до отхода поезда или опоздали на ту же минуту. Надо ли учитывать асимметричность меры в таких ситуациях? Несомненно, и далее это будет показано.

Четвертое перспективное направление — введение и использование вероятностных и вероятностно-метрических мер близости и их эффективный подбор.

3. Обзор существующих современных публикаций по данной тематике

Рассмотрим сначала публикации, в рамках которых используются асимметричные меры близости. Таких работ по сравнению с исследованиями с использованием обычных метрик гораздо меньше.

Приведем ряд примеров с использованием асимметричных мер. Необходимость учета асимметричности меры возникает, например, при формализации понятия степени приближения к истине отдельных фрагментов научного знания — теорий, гипотез, научных утверждений и т.п. в рамках решения проблемы правдоподобности научного знания (логической теории правдоподобности). Пример, иллюстрирующий сложности этой задачи, рассмотрен в монографии основателя этой теории Карла Поппера, которая базируются на основе аппарата формальной логики. В качестве пояснения К. Поппер рассмотрел ситуацию с опозданием на поезд, а для учета степени приближения

(меру правдоподобности) к истине использовал показатель ошибки $E(x, y) = |x - y|$, задавая затем риторический вопрос: “Отклонения от истинного времени в обоих случаях равны, но равная ли мера правдоподобности должна быть приписана этим утверждениям? На основе только одной интуиции решить этот вопрос нельзя, и, во всяком случае, во многих практических ситуациях (например, отправление поезда) ошибка одного существенно менее значима, чем ошибка другого” [10].

Следующий пример употребления асимметричных мер можно обнаружить в исследованиях по кластерному анализу, который используется в различных научных и прикладных областях и также является актуальной темой исследований. Так, в публикации А.Р. Айдиняна и О.Л. Цветковой предложены алгоритмы, предназначенные для кластеризации объектов, которые описываются векторами признаков в пространстве с несоблюдением аксиомы симметрии [11]. “Суть первого из предложенных алгоритмов кластеризации заключается в последовательном формировании кластеров с одновременным перенесением кластеризованных объектов из ранее созданных кластеров в текущий кластер в случае, если это уменьшит критерий качества. По сравнению с существующими алгоритмами неиерархической кластеризации такой подход к формированию кластеров позволяет уменьшить вычислительные затраты” [11].

По мнению исследователей из ИПУ РАН, “недостатком большинства мер, основанных на онтологических структурах, является симметричность (экспертные оценки показывают, что мера близости не всегда симметрична). Кроме того, эти меры независимы от контекста и чувствительны к структуре иерархии” [12].

Среди англоязычных публикаций можно отметить следующие.

В [13] “предлагается асимметричная мера семантической близости. В зависимости от направления прохождения ребрам придается разный вес, так как потомок более подобен родителю, чем родитель — потомку”.

В [14] представлена концепция асимметричной модели сходства и взвешенного вектора частот слов или термов (HFV) и сконструированы две новые асимметричные меры: HFM и HIPM. В статье [14] показано, что симметричные меры сходства не всегда могут эффективно работать в рамках общей проблемы измерения сходства текстов. Будь то информационный поиск, интеллектуальный анализ текстов, извлечение веб-контента, классификация и кластеризация текстов в коллекции и обнаружении копии документов в смысле плагиата и т.д. Такого рода направления исследования требуют методов, способных обрабатывать неструктурированную информацию, поэтому самым популярным подходом является схема, основанная на частоте повторений того или иного элемента, в случае текстов, слов или сочетаний символов. Для документов эта схема использует вектор частот слов из словаря, построенного на всех изучаемых документах, тем самым обеспечивается единая размерность. А стандартными векторными мерами сходства являются функция косинуса, скалярное произведение и функция пропорций. Но все они симметричны.

Представляют интерес и другие англоязычные исследования, см. например, [15–17].

Обзор исследований с использованием асимметричной меры близости показал, что хотя такие работы и существуют, но и в них не учитывается ряд трудностей их использования. Рассмотрим их в разделе 4.

4. Трудности использования асимметричных мер близости

Первая трудность. Автор статьи не может использовать вышеуказанное определение псевдометрического пространства, так как из него вытекает именно симметричность меры. Таким образом, необходимо либо:

I. Ослаблять первую аксиому и (или) аксиому "неравенство треугольника";

II. Использовать нижеследующие аксиоматики для показателей ошибки.

Пусть X^N — множество упорядоченных наборов вещественных чисел. Введем отношение частичного порядка $\geq$ на X^N:

$$x \geq y \Leftrightarrow x_i \geq y_i,$$

где

$$x, y \in X^N, \quad x = (x_1, x_2, \ldots, x_N), \quad y = (y_1, y_2, \ldots, y_N), \quad i = 1, \ldots, N.$$

"*Определение 3.1* [18, с. 170]. *Показателем ошибки E называется действительная функция $E = E(x, y)$, определенная на декартовом квадрате $X^N \times X^N$ и удовлетворяющая следующим условиям*:

1. $E(x, y) = 0 \Leftrightarrow x = y$;

2. $E(x, y) \geq 0$;

3. *Для каждого фиксированного $x \in X^N$ функция $E_x(y) = E(x, y)$ строго монотонно убывает на луче $\{y \in X^N : y < x\}$ и монотонно возрастает на луче $\{y \in X^N : y > x\}$*;

4. *Для каждого фиксированного $y \in X^N$ функция $E_y(x) = E(x, y)$ строго монотонно убывает на луче $\{x \in X^N : x < y\}$ и строго монотонно возрастает на луче $\{x \in X^N : x > y\}$*" [18, с. 170].

Либо "пусть Y — линейно упорядоченное отношением $\leq$ множество, наделенное порядковой топологией, а X — связное подмножество Y.

Определение 3.4 [18, с. 173]. *Показателем ошибки E называется любая действительная функция $E(x, y)$, определенная на декартовом произведении $X \times Y$ и удовлетворяющая следующим условиям*:

1. $E(x, y) = 0 \Leftrightarrow x = y$;

2. $E(x, y) \geq 0$;

3. *Для любого фиксированного $x \in X$ функция $E_x(y)$ строго монотонно убывает на луче $\{y \in Y : y < x\}$ и монотонно возрастает на луче $\{y \in Y : : y > x\}$*;

4. *Для каждого фиксированного* $y \in Y$ *функция* $E_y(x)$ *строго монотонно убывает на луче* $\{x \in X : x < y\}$ *и строго монотонно возрастает на луче* $\{x \in X : x > y\}$;

5. *Функция* $E(x, y)$ *непрерывна по* y." [18, с. 173].

Заметим, что второе определение показателя ошибки отличается от первого не только областью определения, но и наличием пятого условия, которое позволяет рассмотреть новый необходимый и достаточный критерий.

Вторая трудность. Необходимо учитывать, что вывод при сопоставительном анализе, полученный на основе используемой асимметричной меры (показатель ошибки), может зависеть от вида, точнее от класса эквивалентности использованной меры. Возможно, первый, кто обратил на это внимание, был Г. Галилей. В его письме к Ноццолино, приведенном в монографии К. Джини, рассмотрен следующий условный пример [19]: "Пусть перед покупкой лошака один знаток оценил скотину в 10 скудо (денежная единица того времени в Италии), второй — в 1 тыс. скудо, а хозяин продал лошака за 100 скудо. Спрашивается, какая из оценок точнее? Ноццолино утверждал, что второй знаток допустил большую ошибку, потому что его оценка отличается от фактической на 900 скудо, тогда как оценка первого — лишь на 90. На что Галилей возразил, что превышение в оценке второго эксперта равняется приуменьшению в оценке первого, и, значит, ошибки одинаковы."

Таким образом, перед отбором меры необходимо обосновать, какую меру выбрать, а точнее, какой класс эквивалентности таких мер исследователю подходит. Приведем поясняющий пример. Пусть необходимо выяснить, подойдет ли для верификации прогноза прироста ВВП РФ на 2021 г. следующая мера (показатель ошибки) $E(x, y) = |\ln \frac{x}{y}|$, где $x, y \in R$, x – экспертная оценка, y – истинное значение. Первый эксперт дал прогнозную оценку $X1 = 1\%$ ожидаемого прироста, второй эксперт дал оценку $X2 = 25\%$ ожидаемого прироста ВВП по сравнению с 2020 г. Подводя итог 2021 г., экономисты получили оценку прироста ВВП по сравнению с 2020 г. пять процентов ($Y = 5\%$). Если воспользуемся логарифмическим показателем ошибки, то получим, что ошибки двух прогнозов одинаковы. ($|\ln 1/5| = |\ln 25/5|$). По мнению автора, такой показатель для верификации прогноза прироста ВВП РФ на 2021 г. не подходит. И в следующий раз не нужно приглашать для оценивания второго эксперта. Значит, для каждой ситуации надо находить подходящий показатель ошибки. Такой пример является подсказкой для нахождения процедуры подбора вида показателя ошибки, точнее класса эквивалентности показателей ошибки.

Именно с этим и связана третья трудность: построение тех или иных вариантов процедуры подбора вида показателя ошибки. В монографии автора предложен один из вариантов процедуры подбора класса эквивалентности видов показателей ошибки на основе экспертных заключений [18, с. 189]. Поясним последнее утверждение. Классы эквивалентности показателей ошибки вполне естественно вводятся в рамках:

• любых двух различных аксиоматических заданий показателей ошибки: определение 3.1. [18, с. 170] и определение 3.4. [18, с. 173];

• отношения знаковой эквивалентности ($\sim$). См. определение 3.3. [18, с. 171];

• утверждения 3.1. "Отношение знаковой эквивалентности ($\sim$) есть отношение эквивалентности" [18, с. 171].

В рамках рассмотренной и вновь создаваемых аналогичных экспертных процедур возникает необходимость усреднения экспертных оценок. Возникают следующие задачи:

1. Вид используемой средней величины необходимо обосновывать;

2. Кроме того, необходимо уметь рассчитывать найденный вид средних. Но для этого необходимо, чтобы средние были заданы не аксиоматически, а в аналитическом виде.

Поясним последнюю задачу. Если основой для отбора вида средней является следующая аксиома: "можно заменить некоторую группу значений их собственным средним, не меняя общего среднего", то нужно обратить внимание на результат акад. А.Н. Колмогорова [20, 21], которой показал, что каждый тип среднего M^*, если он удовлетворяет четырем условиям:

I. $M^*(x_1, x_2, \ldots, x_n)$ – непрерывна и монотонна по каждому переменному (для определенности будем считать, что M^* – возрастает по каждому переменному);

II. $M^*(x_1, x_2, \ldots, x_n)$ – симметрическая функция;

III. Среднее от одинаковых чисел равно их общему значению:

$$M^*(x, x, \ldots, x) = x;$$

IV. Можно заменить некоторую группу значений их собственным средним, не меняя общего среднего:

$$M^*(x_1, x_2, \ldots, x_m, y_1, y_2, \ldots, y_n) = M^*_{n+m}(x, x, \ldots, x, y_1, y_2, \ldots\ldots, y_n),$$

где $x = M^*(x_1, x_2, \ldots, x_m)$ необходимо имеет вид:

$$M^*(x_1, x_2, \ldots, x_n) = F^{-1}(F(x_1) + F(x_2) + \cdots + F(x_n))/n),$$

где F – непрерывная строго монотонная функция, $(x_i \in [a, b] \subset R)$.

Таким образом, имеем возможность рассчитывать значения такой средней на основе аналитической формулы. Но в рассматриваемом случае, а автор включает и асимметричные меры, вторая аксиома это запрещает. Таким образом, необходимо получить более общую формулу для любой ассоциативной средней, но без второй аксиомы.

В исследовании де Финетти показано, что любая ассоциативная средняя является монотонной [22], таким образом, ему удалось усилить результат акад. А.Н. Колмогорова.

Задача автора настоящей статьи состояла в ослаблении требования симметричности.

В случае если находим среднюю величину для множества, без учета его формирования, условие симметричности является вполне естественным, но

уже для множества, которое формируется из последовательности, — по-видимому, нет. Для подтверждения этого заключения рассмотрим такой широко распространенный способ формирования экспертных групп, как метод снежного кома [18, с. 159]. В этом случае вполне уместно полагать, что эксперт, называя другого эксперта, вряд ли назовет того, у которого уровень знаний по задаваемому вопросу будет ниже, чем у него самого. Таким образом, вполне уместно назначать бóльшую весомость числовой оценки последующего эксперта.

Общая постановка задачи: предложить аналитическое выражение для любой ассоциативной средней, но без условия симметричности, частным случаем которой была бы формула Колмогорова.

Теорема[2]. *Для любой строго монотонной и непрерывной действительнозначной функции* $F : R^0 \to R$ *следующее выражение является формулой для непрерывной ассоциативной средней*:

$$M^*(x_1, x_2, \ldots, x_n) = F^{-1}\left\{\sum_1^n q_i F(x_i)\right\},$$

где $x_i \in R^0$ *– произвольному ограниченному интервалу действительных чисел,* $\sum_1^n q_i = 1$, $q_i \geq 0$, $q_i \in R$.

5. Направления будущих исследований в сфере расширения возможностей метрического подхода

Предлагается рассмотреть следующий вариант расширения метрического подхода. Необходимо обратить внимание, что во всех вышеуказанных случаях областью значения мер близости или функций, которые играют роль метрики, является положительно определенное подмножество действительных чисел.

По нашему мнению, желательно расширять трактовку и этого элемента в аксиоматике меры близости. Для этого рассмотрим объекты, на множестве которых полезно ввести аналоги мер близости и соответствующее понятие “близость”. Среди них, например, такие объекты, как меры близости.

Поясним полезность введения мер близости на мерах близости.

Именно для них могут быть полезны различные варианты изменения области значения мер близости или функций, которые играют роль метрики.

Предварительно необходимо отметить, что в рамках различных исследований их авторы иногда ставят вопрос относительно наиболее подходящих мер близости на множествах различной природы. Например, Г.В. Иофина в диссертации требует, чтобы выбранная мера близости была оптимальна для различных задач, главным образом для задач распознавания [6].

В монографиях Н.Г. Загоруйко и В.Д. Мазурова говорится о выборе правильной функции близости в задачах кластеризации [8, 9].

[2] Данная теорема была впервые доказана в диссертации Ю.В. Сидельникова и анонсирована в его автореферате [23].

Поиск наиболее подходящих мер близости обусловлен различиями реальных и виртуальных объектов, в математических моделях которых используются эти меры близости. Но можно и нужно пойти дальше.

Итак, в дальнейшем исследовании будем исходить из следующего Постулата № 1. В рамках одного и того же реального или виртуального объекта можно, а в ряде случаев необходимо, использовать различные меры близости или их расширения как инструмент их анализа.

При этом будем исходить из того, что эти объекты могут иметь неоднородную структуру и требовать для своего изучения различные меры в различных частях этих же объектов.

Приведем поясняющий пример. Для этого рассмотрим множества, которые обладают одинаковыми свойствами во всех направлениях — изотропные и анизотропные, т.е. обладающие различными свойствами в разных направлениях.

Рассмотрим анизотропные множества, которые содержат N не пустых изотропных подмножеств. При этом эти множества могут быть отображением как реальных, так и виртуальных объектов.

Постулат № 2. Полагаем, что для исследования не только изотропных, но и анизотропных непустых множеств необходимо на этих множествах ввести понятия направления.

Это возможно, например, введя линейный или частичный порядок на множестве элементов этих множеств, а для дальнейших исследований ввести и меру близости или ее различные расширения на элементах этих множеств.

Но, рассматривая различные меры близости на одном и том же объекте, необходимо исследовать, как преобразуются эти меры при переходе из одного изотропного множества в другое в рамках одного и того же объекта. Таким образом, желательно ввести аналог меры близости на мерах близости — "супермеру близости".

Введем понятие супермеры близости (P) на множестве мер близости $E(x, y)$, и исследуем возможности метрической конструкции — $P\{E1(x, y), E2(x, y)\}$, $x, y \in X$, $i = 1, 2, \ldots, n$.

Полагаем, что на множестве мер близости $E(x, y)$ можно ввести как линейный, так и частичный порядок.

Кроме того, для этого отображения необходимо рассмотреть области определения и значения однозначного отображения $P(E(x, y))$, а также ее закон соответствия. А также нужно ввести понятия "различия" и "совпадения" значения супермеры на множестве мер близости.

Полагаем, что область определения отображения $P(E(x, y))$ есть все возможные меры близости или их расширения, например псевдометрика.

На области определения (множество мер близости) можно ввести отношение линейного или частичного порядка, а также ввести тернарное отношение "между".

Это возможно, например, на основе следующего утверждения 3.16: "Пусть Y — линейно упорядоченное отношением $\leq$ множество, наделенное порядко-

вой топологией[3], а X — связное подмножество Y, Φ — непрерывная строго монотонно возрастающая действительнозначная функция. Тогда функция

$$E(x,y) = \begin{cases} \text{K1}[\Phi(x) - \Phi(y)], & \text{если } x > y, \\ \text{K2}[\Phi(y) - \Phi(x)], & \text{если } y \geq x, \end{cases}$$

является показателем ошибки, где $x \in X$; $y \in Y$; $\text{K1}, \text{K2} \in R+$; $\text{K1} \neq \text{K2}$" [18, с. 188].

Так как K1 и K2 являются различными действительными числами, то при фиксированной функции Φ они определяют различные меры близости. А отношение линейного порядка на действительной прямой индуцирует линейный порядок на множестве мер близости, заданных в утверждении 3.16.

Полагаем, что понятие "совпадения" между мерами близости на множестве мер близости можно ввести, например, через отношение "тождества" между функциями $E(x,y)$.

Необходимо обратить внимание, что область значений супермеры является множеством преобразований одной меры в другую. В случае дискретного и конечного множества преобразований характеристику этого множества можно задать минимальным количеством преобразований, необходимых для перехода одной меры в другую, и таким образом получить положительное число.

Пояснить понятие "преобразование" одной из мер в другую можно следующим образом. Пусть, например, имеем два показателя ошибок

$$E(x,y) = |x - y| \quad \text{и} \quad E`(x,y) = |\ln_n x - \ln_n y|.$$

Рассмотрим, как преобразуется одна из мер близости, выраженной функцией $E(x,y)$, в другую функцию $E`(x,y)$.

Полагаем, что "переход" одной функции в другую в этом конкретном случае возможен через преобразование $\ln_n x$ в x. График функции $z = \ln_n x$, при $x \geq 1$ приближается к графику функции $z = x$ при стремлении основания логарифма к бесконечности.

Мера различия между мерами близости на множестве мер близости может выражаться и в порядковой шкале, например в баллах. Приведем поясняющий пример такого различия. Рассмотрим двух разных токарей, но их различие будет только в их разрядах. Один из них будет токарем третьего разряда, а второй токарь — второго разряда.

Покажем полезность и необходимость такой конструкции. При этом будем исходить из Постулат № 3.

Само множество, где определены меры близости $E(x,y)$, может быть не только изотропным, т.е. обладать одинаковыми свойствами во всех направлениях, но и анизотропным, т.е. обладать различными свойствами в разных направлениях.

[3] Порядковой топологией называется наименьшая система подмножеств множества Y, замкнутая относительно объединения и конечного пересечения и содержащая все открытые лучи [24].

Так, например, в реальном космическом пространстве вблизи массивных тел для расчетов используется метрика Минковского, а вдали от массивных тел можно использовать метрику Эвклида.

По нашему мнению, необходимо исследовать свойства такой метрической конструкции, где введена супермера близости P на множестве мер близости $E(x, y)$.

6. Заключение

В статье рассмотрены три варианта расширения метрического подхода для решения специальных задач, например в теории расписаний, связанные с ослаблением требований к аксиоматике метрики. Кроме того, предложено поддержать вариант расширения метрического подхода на основе вероятностных мер близости и вероятностно-метрических мер близости, рассмотренных Д.З. Уздиным в монографии [3].

Указано на перспективность выбора функций близости для разнообразных типов задач. Так, Г.В. Иофина в исследовании требует, чтобы выбранная мера близости была оптимальна для различных задач, главным образом для задач распознавания [6]. Н.Г. Загоруйко и В.Д. Мазуров пишут о выборе правильной функции близости в задачах кластеризации [8, 9]. Но как выбирать оптимальную или правильную меру близости? Этот вопрос пока не решен в полной мере. Хотя для асимметричных числовых мер близости удалось построить такую процедуру подбора вида показателя ошибки [18, с. 189].

Кратко описаны результаты Ю.В. Сидельникова, связывающие наиболее известные аналитические средние с характеристиками классов эквивалентности числовых показателей ошибки в рамках двух различных аксиоматических заданий показателей ошибки.

Усилен результат, полученный А.Н. Колмогоровым, при рассмотрении им системы аксиом для вывода аналитической формулы ассоциативной средней. В статье получена формула для ассоциативной средней, но уже без условия ее симметричности.

Предложено новое направление будущих исследований в сфере расширения возможностей метрического подхода, связанное с расширением метрического подхода. Введено понятие супермеры близости и рассмотрен случай, когда областью определения супермеры близости являются меры близости или функции, которые играют роль метрики. Показана полезность данной конструкции.

ПРИЛОЖЕНИЕ

Доказательство теоремы.

Во-первых, докажем, что $M^*(x_1, x_2, \ldots, x_n)$ является средним по Коши, т.е. $\min x_i \le M^*(x_1, x_2, \ldots, x_n) \le \max x_i$.

Действительно, пусть F — строго монотонно возрастающая функция, тогда

$$\sum_{i=1}^{n} q_i F(\min x_i) \le \sum_{i=1}^{n} q_i F(x_i) \le \sum_{i=1}^{n} q_i F(\max x_i),$$

$$F(\min x_i) \le \sum_{i=1}^{n} q_i F(x_i) \le F(\max x_i).$$

Аналогично можно показать и для F – строго монотонно убывающей функции.

Теперь докажем, что средняя, выраженная в виде $M^*(x_1, x_2, \ldots, x_n) = F^{-1}\{\sum_1^n q_i F(x_i)\}$, является ассоциативной, т.е.

$$M^*(x_1, x_2, \ldots, x_m, x_{m+1}, \ldots, x_n) = M_n^*(x, x, \ldots, x, x_{m+1}, \ldots, x_n),$$

где $x = M_m^*(x_1, x_2, \ldots, x_m)$ для любого $m < n$.

Формула для среднего при $m < n$, образуется следующим образом:

$$x = M_m^*(x_1, x_2, \ldots, x_m) = F^{-1}\left\{\sum_1^m \left(q_i / \sum_1^m q_j\right) F(x_i)\right\},$$

т.е. сохраняя прежние коэффициенты, заново нормируем их, чтобы сохранить условие

$$\sum_1^m q_i^* = \sum_1^m \left(q_i / \sum_1^m q_j\right) = 1$$

$$M_m^*(x, x, x_{m+1}, \ldots, x_n) = F^{-1}\left\{\sum_1^m q_i F(x) + \sum_{m+1}^n q_i F(x_i)\right\} =$$

$$= F^{-1}\left\{\sum_1^m q_i F\left[F^{-1}\left(\sum_1^m \left(q_i / \sum_1^m q_j\right) F(x_i)\right)\right] + \sum_{m+1}^n q_i F(x_i)\right\} =$$

$$= F^{-1}\left\{\sum_1^m q_i \left[\sum_1^m \left(q_i / \sum_1^m q_j\right) F(x_i)\right] + \sum_{m+1}^n q_i F(x_i)\right\} =$$

$$= F^{-1}\left\{\sum_1^m q_i F(x_i) + \sum_{m+1}^n q_i F(x_i)\right\} = M^*(x_1, x_2, \ldots, x_n).$$

Теорема доказана.

СПИСОК ЛИТЕРАТУРЫ

1. *Lazarev A.A., Lemtyuzhnikova D.V., Werner F.* A Metric Approach for Scheduling Problems with Minimizing the Maximum Penalty // Appl. Math. Modell. Oxford, Great Britain, Elsevier. 2021. No. 89. P. 1163–1176.
2. *Раушенбах Г.В.* Меры близости в пространстве множеств / Алгоритмы анализа данных социально-экономических исследований ЭИ и ОПП СО АН СССР. Новосибирск: 1982. С. 29–44.
3. *Уздин Д.З.* Новые меры близости, функции состояний и решающие правила в теории распознавания состояний (статистической классификации). 2-е изд., доп. и испр. М.: МАКС Пресс, 2016.
4. *Колмогоров А.Н., Фомин С.В.* Элементы теории функций и функционального анализа. М.: Наука, 1968.
5. *Коллатц Л.* Функциональный анализ и вычислительная математика / Пер. с немецкого И.Г. Нидеккера, под ред. А.Д. Горбунова. М.: Изд-во Мир, 1964.
6. *Иофина Г.В.* Выбор оптимальных метрик в задачах распознавания с порядковыми признаками. Дисс. на соискание уч. степ. канд. физ.-мат. наук: М.: 2010.
7. Математическая энциклопедия. М.: Сов. энциклопедия. Т. 4. Ок–Сло. 1984.
8. *Загоруйко Н.Г.* Прикладные методы анализа данных и знаний. Новосибирск: Изд. Ин-та математики СО РАН, 1999.
9. *Мазуров В.Д.* Метод комитетов в задачах оптимизации и классификации. М.: Наука, 1990.
10. *Popper K.R.* Conjectures and Refutations. The Growth of Scientific Knowledge. Ch. 10, section 3. London: Routledge and Kegan Paul; N.Y.: Basic Books Inc., 1963.
11. *Айдинян А.Р., Цветкова О.Л.* Алгоритмы кластерного анализа задач с асимметричной мерой близости // Сиб. журн. вычисл. матем. 2018. № 2. С. 127–138.
12. *Крюков К.В., Панкова Л.А., Пронина В.А., Шипилина Л.Б.* Меры семантической близости в онтологиях // Проблемы управления. 2010. № 5. С. 2–14.
13. *Bulskov H., Knappe R., Andreasen T.* On Measuring Similarity for Conceptual Querying / T. Andreasen, A. Motro, H. Christiansen, H.L. Larsen (Eds.). Flexible Query Answering Systems. Lecture Notes in Artificial Intelligence. Berlin: Springer, 2002. V. 2522. P. 100–111.
14. *Jun-Peng Bao, Jun-Yi Shen, Xiao-Dong Liu, Hai-Yan Liu.* Quick Asymmetric Text Similarity Measures // Proc. Second Int. Conf. on Machine Learning and Cybernetics. Wan, 2–5 November 2003. V. 1. P. 374–379.
15. *Hubert L.J.* Min and Max Hierarchical Clustering Using Asymmetric Similarity Measures // Psychometrika. 1973. No. 38. P. 63–72.
16. *Faith D.P.* Asymmetric Binary Similarity Measures // Oecologia. 1983. V. 57. No. 3. P. 287–290.
17. *Chen H.H., Giles C.L.* ASCOS++ An Asymmetric Similarity Measure for Weighted Networks to Address the Problem of SimRank // ACM Trans. Knowledge Discovery from Data (TKDD). 2015. V. 10. No. 2. P. 1–26.
18. *Сидельников Ю.В.* Системный анализ технологии экспертного прогнозирования. М.: Изд-во МАИ-ПРИНТ, 2007.
19. *Джини К.* Средние величины. М.: Статистика, 1970.
20. *Kolmogorov A.N.* Sur la Notion de Moyenne. Atti Accad. Naz. Lincei Rend. Cl. Sci. Fis. Mat. Natur. (6) 12 (1930). P. 388–391.

21. *Колмогоров А.Н.* Об определении среднего / Математика и механика. Избранн. тр. Отв. ред. С.М. Никольский, сост. В.М. Тихомиров. М.: Наука, 1985. Т. 1. С. 136–138.
22. *Finetti B. de.* Sul Concetto di Media // Giornale dell Instituto Ital. degli Attuari, 1931. Anno 11. No. 3. P. 365–396.
23. *Сидельников Ю.В.* Автореферат на соиск. уч. степ. д-ра техн. наук "Технология экспертного прогнозирования". 2002, ОКОРОИ ИМЭМО РАН.
24. *Пфанцагель И.* Теория измерений. М.: Мир, 1976.

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

Поступила в редакцию 10.02.2021

После доработки 23.06.2021

Принята к публикации 30.06.2021

© 2021 г. В.М. СТАРОЖИЛЕЦ (starvsevol@gmail.com),
Ю.В. ЧЕХОВИЧ, канд. физ.-мат. наук (chehovich@forecsys.ru)
(Федеральный исследовательский центр
«Информатика и управление» РАН, Москва)

ОБ ОДНОМ ПОДХОДЕ К СТАТИСТИЧЕСКОМУ МОДЕЛИРОВАНИЮ ТРАНСПОРТНЫХ ПОТОКОВ НА МКАД И УПРАВЛЕНИЮ ВЪЕЗДАМИ[1]

Работа посвящена математическому моделированию транспортных потоков в большой автомобильной сети. Использована предложенная авторами статистическая модель транспортных потоков, предназначенная для полномасштабного моделирования функционирования транспортных систем значительного размера в течение длительных интервалов времени. Построена модель одной из сторон МКАД и проведены эксперименты для двух путей возникновения заторного движения на магистрали. В обоих типах экспериментов проведена проверка на работоспособность наивной модели управления въездами. Показана эффективность с точки зрения временных потерь на проезд по автомагистрали даже простейшего метода ограничения входного потока.

Ключевые слова: моделирование транспортных потоков, фундаментальная диаграмма потоков, группы автомобильно-транспортных средств (АТС), МКАД, моделирование автомагистрали.

DOI: 10.31857/S0005231021110088

1. Введение

В работе рассматривается проблема математического моделирования автомобильных транспортных потоков в рамках транспортных систем значительных масштабов. Непосредственно рассматривается задача применения математической модели, описанной в [1], к моделированию конкретной транспортной магистрали — Московской кольцевой автомобильной дороги (МКАД). Целью моделирования является проверка гипотезы о возможности посредством управления потоком въезжающих на магистраль автомобильно-транспортных средств (АТС) уменьшить суммарные потери времени на МКАД и увеличить пропускную способность автомагистрали.

На сегодняшний день моделирование крупных транспортных сетей представлено в [2, 3] в виде примеров применения существующих программных

[1] Работа выполнена при финансовой поддержке Российского фонда фундаментальных исследований (проект № 20-07-01057 А).

пакетов, таких как SUMO (Simulation of Urban Mobility), iTETRIS (An Integrated Wireless and Traffic Platform for Real-Time Road Traffic Management Solutions) и др. Детальное описание подхода к моделированию автомагистрали в данных пакетах зачастую отсутствует.

В [1] предложена математическая модель, свойства которой были исследованы на модельных элементарных фрагментах транспортной сети. Эта модель создавалась для расчетов пропускной способности в различных режимах работы транспортных графов значительного масштаба, включающих тысячи сегментов и имеющих протяженность десятки километров. Модель основана на оригинальном мезоскопическом подходе, оперирующем в качестве объектов моделирования группами автомобильных транспортных средств (АТС), объединяющими автомобили со сходными параметрами, находящимися на одном сегменте транспортного графа. Скорость групп автомобилей рассчитывается с помощью фундаментальной диаграммы поток-плотность на магистрали [4]. Такой подход позволяет быстро обсчитывать достаточно большие транспортные сети, в том числе такую магистраль, как МКАД, что необходимо для решения оптимизационных задач, для которых проводится моделирование.

Данный подход отличается от двух классических направлений к моделированию транспортных потоков, представленных микроскопическим подходом, основанным на моделировании движения каждого отдельного автомобиля [5–8], и макроскопическим подходом, опирающимся на сходство движения АТС с жидкостью или газом [9–12].

Моделирование транспортных потоков на автомагистрали тесно связано с задачей оптимизации светофорного управления в транспортной сети [13]. Так, в [14, 15] используется обучение с подкреплением для получения оптимальной схемы управления перекрестком. В [16] конструируется адаптивное управление светофором на перекрестке на основе загруженности сегментов дороги за ним, а в [17] строится модель выхлопов от автомобилей и целью светофорного управления задается минимизация выхлопов.

Из мезоскопических моделей можно выделить [18], где рассматривается комбинация микро-, мезо- и макроскопических моделей для расчета выделения углекислого газа в атмосферу в транспортной сети, а также [19], где мезоскопическая модель используется для моделирования пешеходного движения, однако в ней проводятся расчеты вычислительной сложности полученной модели и проводится сравнение зависимости вычислительных затрат относительно плотности потока пешеходов для рассматриваемой мезоскопической модели и выбранных микро- и макромоделей.

В большинстве работ, посвященных светофорному управлению, дороги на перекрестке считаются равнозначными и не ставится задача обеспечения максимальной пропускной способности выделенной автомагистрали, как это происходит в данной статье.

В данной работе строится модель одной из сторон МКАД с крупными въездами и съездами с нее. На основе имеющихся данных с дорожных датчиков на некоторых из въездов и статистических данных Центра органи-

зации дорожного движения генерируются модельные данные на въездах на автомагистраль двух типов: с утренней пиковой загрузкой и с вечерней, что соответствует большому потоку автомобилей в Москву и из Москвы. Проводится моделирование поведения автомагистрали с различным модельными данными на въездах и сравниваются результаты с контролем на въездах и без него. Для проверки гипотезы об эффективности управления въездами рассчитываются временные потери на проезд по МКАД за день, а также число автомобилей, проехавших по автомагистрали.

2. Описание модели

Опишем кратко ранее разработанную процедуру моделирования, представленную детально в [1].

В модели транспортная сеть представляет из себя связный ориентированный граф $\mathbf{G} = (\mathbf{V}, \mathbf{E})$, где $\mathbf{V}$ – множество вершин, $\mathbf{E} = \{(i,j)\}$ – множество ветвей графа с ограничением на степень вершин $d(i)$: $\min(d(i)) = 1$ и $\max(d(i)) = 3$. Причем не существует вершины с $d(i) > 1$, в которой только заканчивалось бы несколько ветвей или только начиналось бы несколько ветвей. Таким образом, рассматриваются только автомагистрали, на которых не может быть перекрестков, только съезды либо въезды на магистраль.

Каждая ветвь модели характеризуется следующими свойствами:

1. Длина ветви — $l_{i,j}$.
2. Число полос, по которым разрешено движение автомобилей по данной ветви — $n_{i,j}$.
3. $I_{i,j} = \{0, 1\}$ — идентификатор того, является ветвь съездом или нет. Если является, то $I_{i,j} = 1$.
4. Функция скорости для данной ветви — $V = f_{i,j}(\rho)$, $f_{i,j} : \mathbb{Q}_+ \to \mathbb{Q}_+$, где $\rho \in \mathbb{R}_+$ — плотность АТС. В данной работе рассматриваются только ограниченные непрерывные монотонно убывающие функции скорости. Процедура получения данной функции из экспериментальных данных детально описана в [4].
5. Матрица перемешивания в узле j в момент времени t задается функцией $M_j(t)$. Причем для узлов j, из которых в модели нет съезда, $M_j(t) = 0$, т.е. автомобили либо продолжают движение по автомагистрали, либо данный узел инцидентен ветви-стоку, на которой автомобили в модели заканчивают свое движение.
6. Интенсивность источника в узле i в момент времени t задается функцией $F_i(t)$. Для всех узлов i, не являющихся источниками в модели, $F_i(t) = 0$.

В модели по ветвям графа движутся не отдельные автомобили, а группы автомобильно-транспортных стредств (АТС): $\mathbf{A}_k^t = \{\mathrm{Pos}_k, V_k, N_k\}$, обладающие следующими характеристиками:

1. Pos_k — позиция начала группы относительно начала ветви, на которой она расположена.

2. V_k — скорость группы АТС.
3. N_k — размер группы АТС из $\mathbb{R}_{\geqslant 0} = \mathbb{R}_+$.

Пусть теперь $\mathbf{A}^t_{i,j} = \{\mathbf{A}^t_k\}$ — упорядоченное множество автомобильных групп на ветви (i, j). Причем $\forall l,\ m : l < m \to \mathrm{Pos}_l > \mathrm{Pos}_m$ — группы не могут обгонять друг друга.

Таким образом, вводится состояние системы в момент времени t как $\mathbf{A}^t = \{\mathbf{A}^t_{i,j}\} \cup \{A^t_{\mathrm{out},i,j}\}$, т.е. положение, скорость, размер и тип всех автомобильных групп на всех ветвях дорожно-транспортной сети. Группы АТС $\{A^t_{\mathrm{out},i,j}\}$ представляют собой специальные группы-буферы для обработки очередей на съездах с автомагистралей.

Переход от времени t ко времени $t + \tau$ осуществляется с помощью последовательного выполнения следующей процедуры:

1. Удаляем все группы АТС, находящиеся на ветвях-стоках.
2. Формируем новые группы АТС во всех узлах-источниках.
3. Пусть $\mathbf{C}$ — некоторое подмножество ветвей графа. Будем выполнять следующие действия, пока оно не пусто:
   (a) Исходно $\mathbf{C}$ — множество всех ветвей-стоков.
   (b) $\forall (i, j) \in \mathbf{C} \to \forall \mathbf{A}^t_k \in \mathbf{A}^t_{i,j}$ — для каждой группы АТС рассчитываем ее новое положение. Расчет производится упорядоченно по убыванию величины Pos_k, причем группы-буферы обсчитываются первыми.
   (c) $\mathbf{C}' = \{(k, i)\} : \exists j : (i, j) \in \mathbf{C}$ — формируем новое подмножество для расчетов.
   (d) $\mathbf{C} = \mathbf{C}'$.
4. $\forall (i, j) \in \mathbf{E} \to \forall \mathbf{A}^t_k \in \mathbf{A}^t_{i,j}$ — объединяем группы АТС, если это возможно.

3. Построение модели МКАД

Модель транспортной сети в данной работе представляет из себя связанный ориентированный граф $\mathbf{G}$. Данный граф строится на основе топологии МКАД и прилегающих к нему дорог. При построении графа вручную размечаются основные ребра-въезды на автомагистраль и ребра-съезды с автомагистрали. Причем разметка въездов разделяет их на два типа — въезды с магистралей, направленных в Москву, и с магистралей, направленных из Москвы. Это нужно ввиду того, что пиковый поток на этих двух типах въездов приходится на разное время суток.

Поскольку в топологии не выделены сегменты, отвечающие за сам МКАД, но указаны координаты каждого ребра, то выделение автомагистрали проводится следующим образом:

1. Выбирается один сегмент i на автомагистрали; так как координаты начала и конца сегмента известны, то можем представить его как вектор $\mathbf{i}$.

2. Ищутся все сегменты топологии, идущие после него, и считается их векторное представление. Обозначим множество этих векторов через $\mathbf{S}$.
3. $\forall \mathbf{j} \in \mathbf{S}$ — рассчитываем угол между векторами $(\mathbf{i}; \mathbf{j})$ и выбираем сегмент с наименьшим углом как продолжение магистрали $\mathbf{i}'$.
4. Возвращаемся к пункту 1 с $\mathbf{i} = \mathbf{i}'$, пока не вернулись в исходный сегмент (для МКАД) или не достигнем ее конца (в этом случае требуется задать сегмент — конец автомагистрали).

Данная процедура значительно уменьшает объемы ручной работы для формирования графа $\mathbf{G}$. Однако она все же допускает ошибки и требуется формирование небольшого списка сегментов топологии, точно не являющихся искомой автомагистралью. В данной работе этот список состоит из 14 сегментов. Все еще требуется вручную разметить основные въезды и съезды с автомагистрали, однако можно проигнорировать незначительные, т.е. съезды на небольшие прилегающие дороги и въезды на них, поток на которых пренебрежимо мал для целей этой работы.

В результате работы по данному алгоритму получен связный ориентированный граф $\mathbf{G}$ одной из сторон МКАД со всеми необходимыми въездами и съездами.

4. Генерация синтетических данных на въездах

Ввиду отсутствия открытых источников данных с дорожных датчиков на въездах требуется сгенерировать реалистичный поток автомобилей синтетически. По информации от ЦОДД [20] по всему МКАД за сутки проезжает 1,36 млн автомобилей (т.е. приблизительно 680 тысяч АТС по одной стороне),

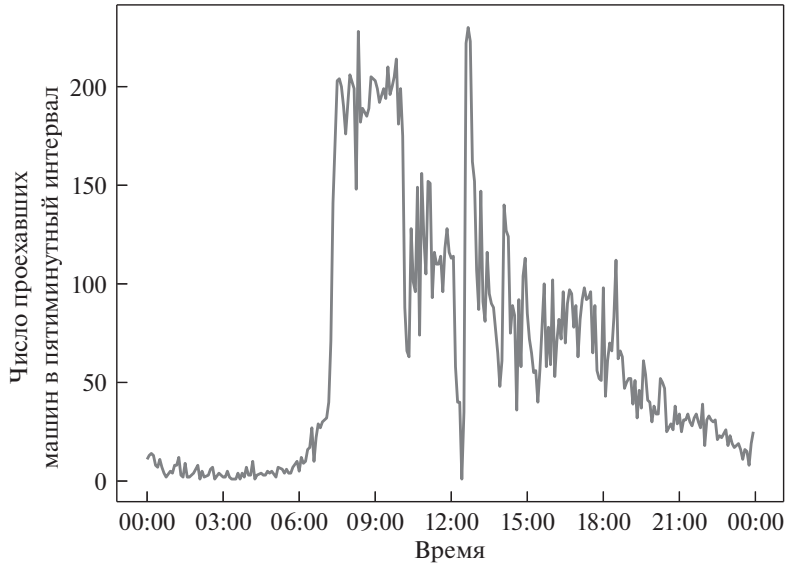

Рис. 1. Данные с дорожного датчика за один день. Пиковая нагрузка 45 АТС/мин в 8:20.

Рис. 2. Типичные пробки по понедельникам в 18:15 на основе статистики сервиса «Яндекс-пробки» транспортной сети Москвы и МКАД, в частности по состоянию на 16.05.21.

а по имеющимся данным с дорожных датчиков пиковый поток АТС на въезде составляет 60 АТС/мин. Пример данных с дорожных датчиков показан на рис. 1.

Таким образом, в экспериментах функции входного потока на въездах строились так, чтобы походить на данные с реального дорожного детектора и соответствовали информации о пиковой (или средней, если это требовалось) нагрузке на въезде и числу проезжающих по автомагистрали за день АТС.

Как говорилось выше, все въезды также вручную были разделены на два класса:

1. Въезды с магистралей по направлению из Москвы, на которых поток АТС нарастает ближе к вечеру.
2. Въезды с магистралей по направлению в Москву, на которых поток АТС нарастает утром и спадает к вечеру.

5. Описание данных

В работе проводится моделирование внешней стороны МКАД (рис. 2). Для построения графа использовалась топология, взятая у компании Яндекс в 2014 г. Полученная топология, а также увеличенный участок МКАД со съездами и въездами изображены на рис. 3.

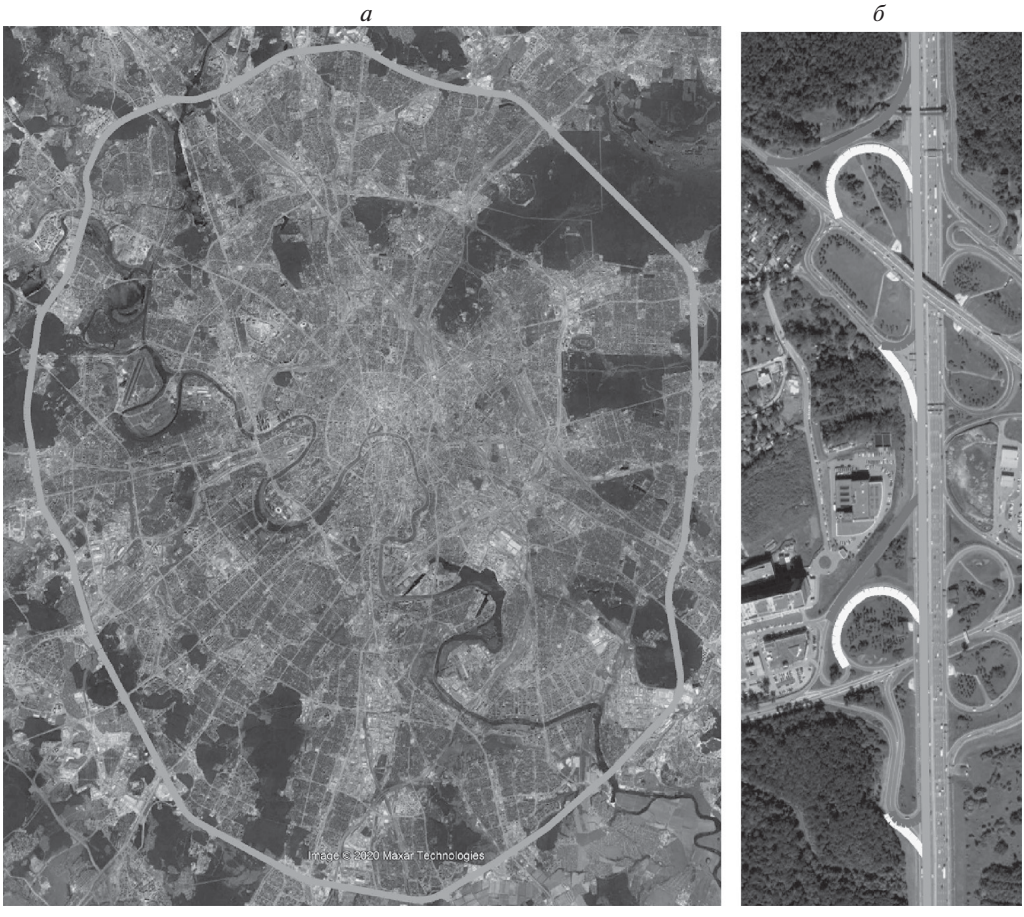

Рис. 3. *а*) Вид расчетного графа МКАД, полученного с помощью топологии компании Яндекс, *б*) конфигурация въездов и съездов с МКАД в полученном на основе топологии компании Яндекс графе. Темная вертикальная линия — МКАД, светло-серая — въезды на магистраль, темная — съезды с нее.

В данной работе во всех экспериментах использовались фундаментальные диаграммы поток-плотность, полученные анализом реальных данных с дорожных датчиков за 2012 г. Построение фундаментальной диаграммы сводится к следующим шагам:

1. Для каждого надежного датчика на выбранном участке дороги извлекаем данные измерений плотности и потока за наблюдаемый период времени. Каждая точка на диаграмме определяется парой значений «плотность-поток» на плоскости $Q(\rho)$.
2. Проводим фильтрацию выбросов путем построения альфа-оболочек облака точек диаграммы до тех пор, пока разница площадей оболочек для двух смежных итераций не будет мала.
3. Находим опорные точки на границе облака точек диаграммы и строим на их основе функцию-огибающую, которую и принимаем за фундаментальную диаграмму поток-плотность.

Детально процедура построения фундаментальной диаграммы описана в [4].

6. Вычислительные эксперименты

Проведены следующие группы вычислительных экспериментов:

1. Эксперименты со средней, но продолжительной, пиковой загрузкой на въезды с проверкой эффекта от динамического ограничения входного потока в зависимости от состояния автомагистрали.
2. Эксперименты с высокой, но непродолжительной, пиковой загрузкой въездов (что более соответствует данным от ЦОДД) с проверкой эффекта от динамического ограничения входного потока в зависимости от состояния автомагистрали.
3. Эксперименты с длинными въездами с высокой, но непродолжительной, пиковой загрузкой въездов с проверкой эффекта от динамического ограничения входного потока в зависимости от состояния автомагистрали. В данной группе экспериментов максимальная длина очереди на въездах на МКАД увеличена для расчетов времени ожидания на въезде на магистраль без управления въездами и с ним.

В связи с отсутствием реальных данных о числе покидающих автомагистраль транспортных средств в каждый момент времени считаем эту долю фиксированной и выбранной из следующих соображений:

1) проезжать более половины МКАД в одну сторону неосмысленно, так как в данном случае можно поехать в другую сторону;
2) на половине МКАД в рассматриваемой модели 32 съезда.

Таким образом, если x — доля съезжающих на каждом съезде АТС, то величина $(1-x)^{30}$ должна быть мала. В экспериментах используется величина $x = 12\%$.

В каждой группе экспериментов также проводится моделирование ситуации установки светофора на въездах на автомагистраль. В таком случае алгоритм ограничения входного потока на МКАД выглядит следующим образом:

- Для каждого сегмента автомагистрали по направлению движения АТС после рассматриваемого въезда посчитаем плотность автомобилей ρ на ней.
- В зависимости от величины $\rho_{\text{opt}} - \rho$, где ρ_{opt} — плотность, при которой достигается максимальный поток на рассматриваемом сегменте автомагистрали, входной поток ограничивается на $l < l_{\max}$ процентов.
- Ограничения для каждого из сегментов складываются и получается результирующее понижение входного потока АТС.

Важной характеристикой моделируемой системы считаем временные потери на проезд по транспортной сети относительно пустой автомагистрали, т.е. автомагистрали, по которой возможно движение АТС с максимально допустимой скоростью. Каждую минуту рассчитывается среднее продвижение каждого автомобиля в модели и сравнивается с расстоянием, которое он мог бы преодолеть по пустой магистрали. Это преобразуется в график временных потерь, который трактуется как время простоя автомобиля в транспортной сети за минуту.

7. Эксперименты со средней загрузкой

В данной группе экспериментов въезды считаются однополосными и функции входного потока изображены на рис. 4. В данном случае есть два типа въездов на автомагистраль — с утренней и вечерней пиковыми загрузками в течение трех часов.

7.1. Эксперимент без управления въездами

Результаты моделирования автомагистрали при такой конфигурации въездов представлены на рис. 5. Число реально въехавших автомобилей и коли-

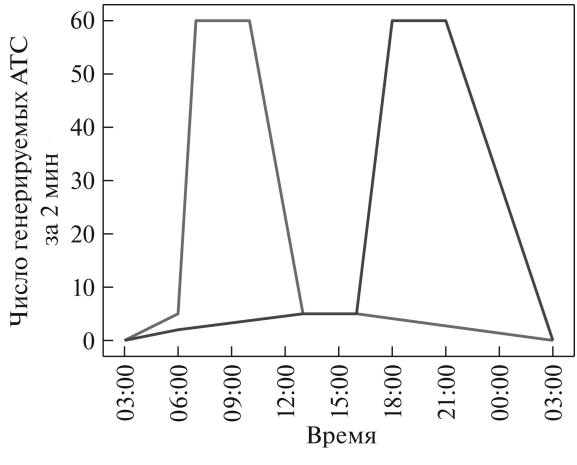

Рис. 4. Графики загрузки двух типов въездов — с утренней и вечерней пиковыми загрузками в эксперименте со средней загрузкой.

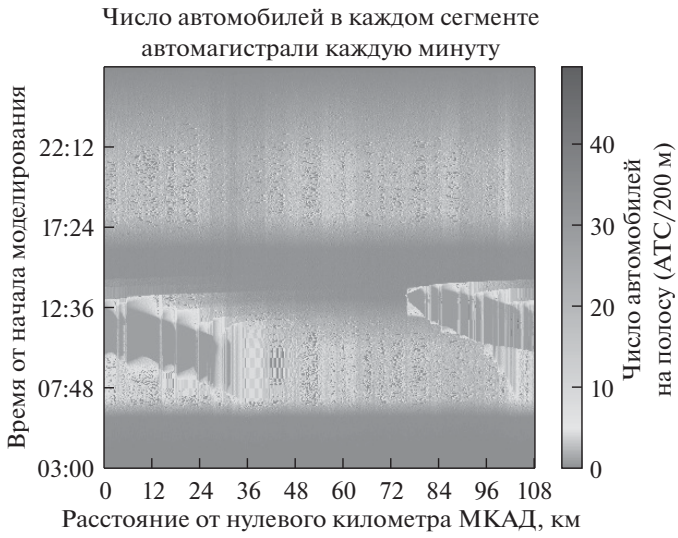

Рис. 5. Количество автомобилей на полосе в модели транспортной сети за день в эксперименте со средней загрузкой.

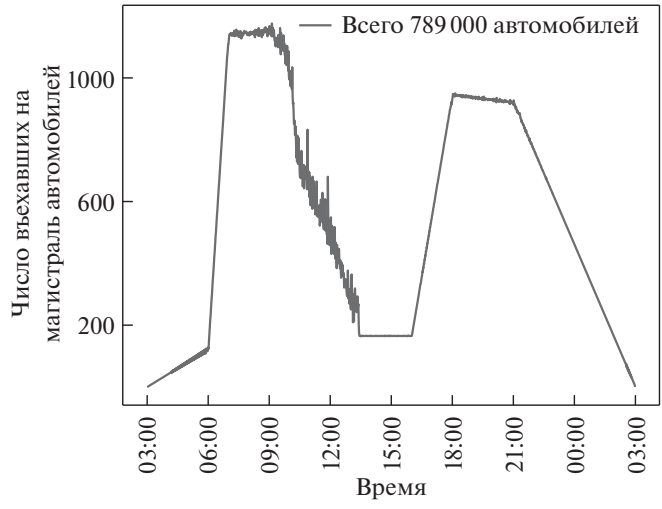

Рис. 6. График суммарно въехавшего на автомагистраль со всех въездов числа автомобилей в эксперименте со средней загрузкой.

чество проехавших за день по транспортной сети АТС изображены на рис. 6. График временных потерь — на рис. 7.

Видно, что при такой конфигурации входных потоков заторы возникают всего в нескольких местах и потом со временем распространяются по автомагистрали. Так как пробки успевают исчезнуть к вечеру, то МКАД не останавливается полностью, хотя при меньшей доли съезжающих автомобилей это произойдет.

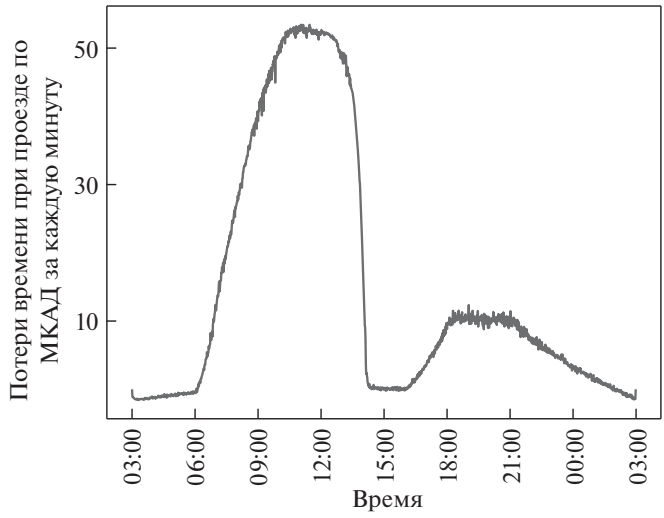

Рис. 7. Временные потери на проезд по автомагистрали в эксперименте со средней загрузкой.

7.2. Эксперимент с управлением въездами

Промоделируем ситуацию, в которой при увеличении потока на автомагистрали будем ограничивать поток с ближайших въездов на автомагистраль искусственно, например с помощью светофора. В данном эксперименте можем перекрывать въезд вплоть до 80% в зависимости от плотности автомо-

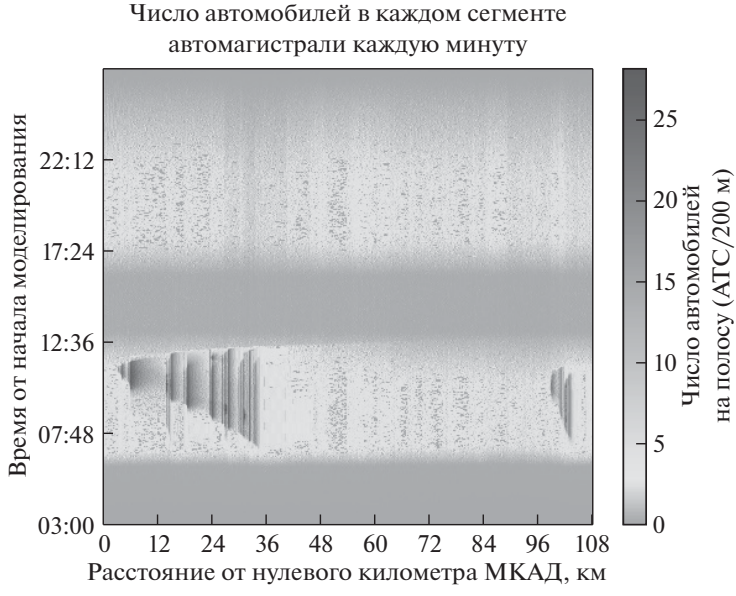

Рис. 8. Количество автомобилей на полосе в модели транспортной сети за день в эксперименте со средней загрузкой с управлением въездами.

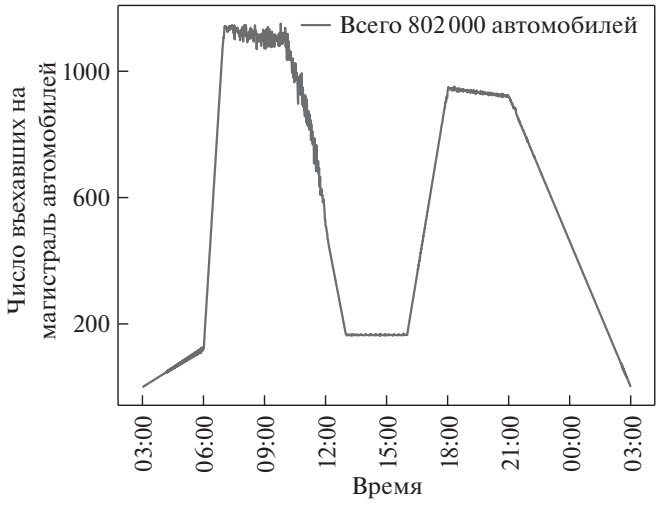

Рис. 9. График суммарно въехавшего на автомагистраль со всех въездов числа автомобилей в эксперименте со средней загрузкой с управлением въездами.

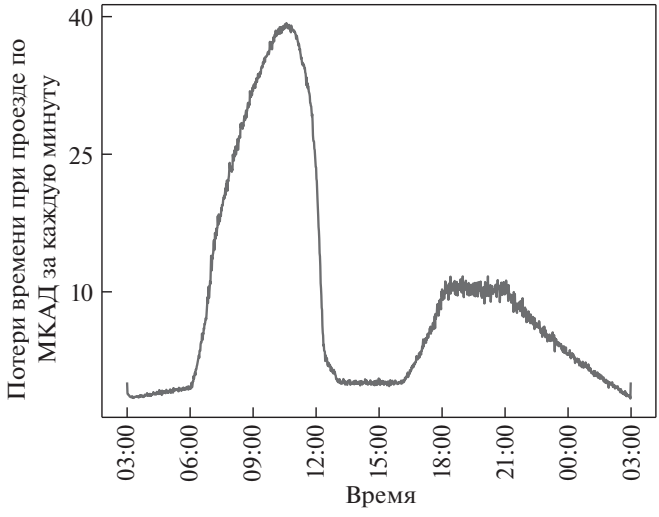

Рис. 10. Временные потери на проезд по автомагистрали в эксперименте со средней загрузкой с управлением въездами.

билей на магистрали. Результаты моделирования при такой конфигурации въездов представлены на рис. 8. Число реально въехавших автомобилей и количество проехавших за день по транспортной сети АТС изображены на рис. 9. График временных потерь проезда по всей автомагистрали представлен на рис. 10.

На графиках видно уменьшение времени затора на МКАД, а также небольшое увеличение числа проехавших автомобилей. Однако временные потери на проезд по автомагистрали значительно снизились. Интегральная

разность между графиками временных потерь на рис. 7 и 10 составляет около 4, 5 мин.

8. Эксперименты с высокой загрузкой

В данной группе экспериментов въезды считаются двухполосными и функции входного потока изображены на рис. 11. В данном случае есть два типа въездов на автомагистраль — с утренней и вечерней пиковыми загрузками в течение трех часов.

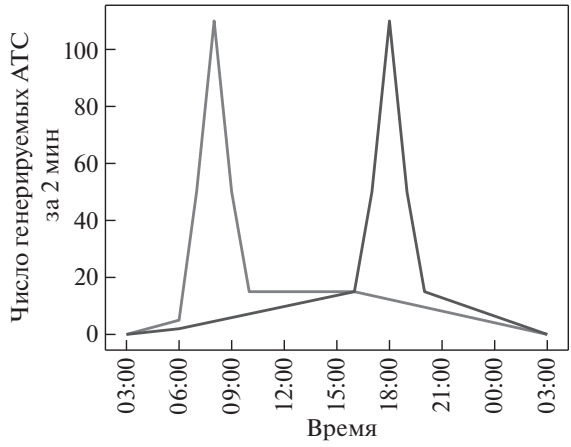

Рис. 11. Графики загрузки двух типов въездов — с утренней и вечерней пиковыми загрузками в эксперименте с высокой загрузкой.

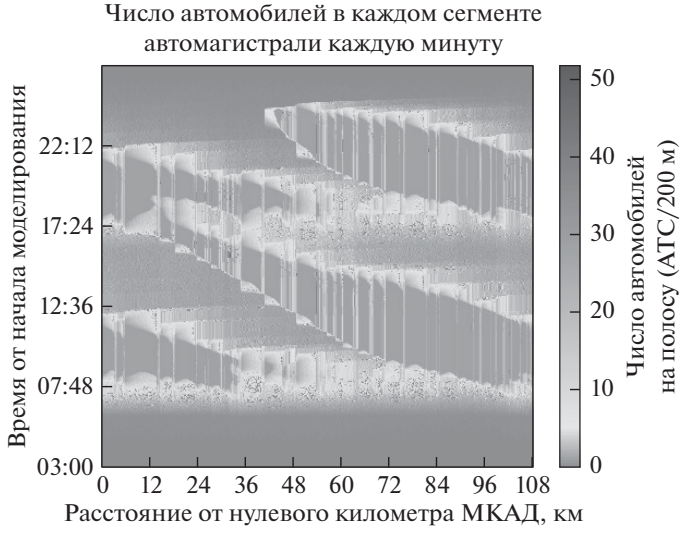

Рис. 12. Количество автомобилей на полосу в модели транспортной сети за день в эксперименте с высокой загрузкой.

8.1. Эксперимент без управления въездами

Результаты моделирования при такой конфигурации въездов представлены на рис. 12. Видно, что в данной конфигурации потоков на въездах заторные движения образуются по всей протяженности автомагистрали, объединяясь впоследствии в один большой. В данном эксперименте МКАД практически полностью занята пробкой с утра до вечера. На рис. 13 показано число реально въехавших автомобилей и количество проехавших за день по

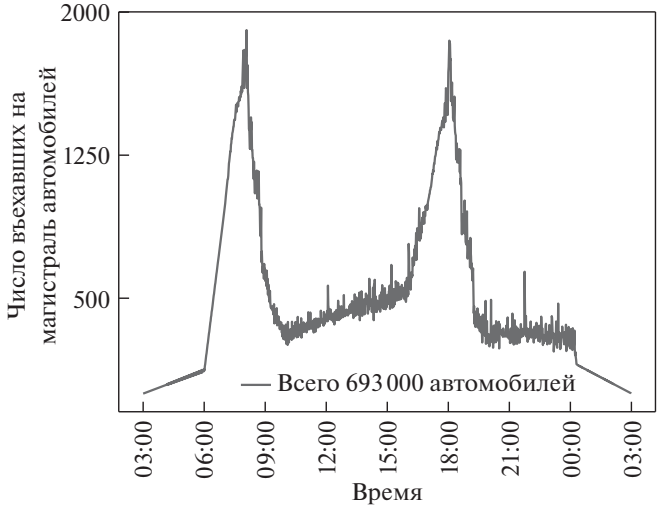

Рис. 13. График суммарно въехавшего на автомагистраль со всех въездов числа автомобилей в эксперименте с высокой загрузкой.

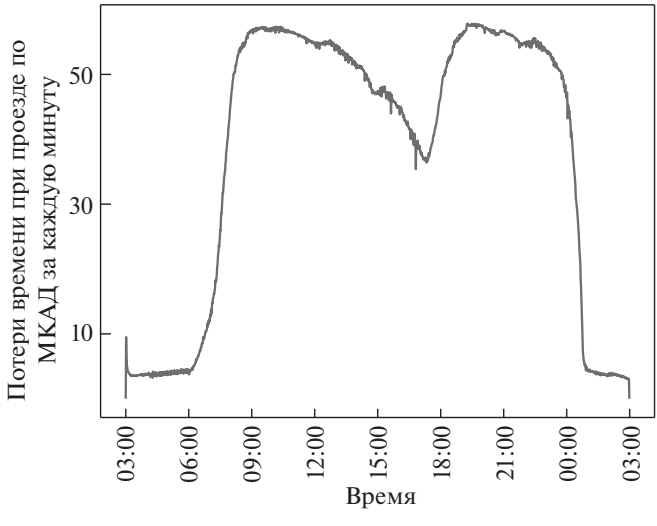

Рис. 14. Временные потери на проезд по автомагистрали в эксперименте с высокой загрузкой.

магистрали АТС. График временных потерь проезда по всей автомагистрали представлен на рис. 14.

8.2. Эксперимент с управлением въездами

В данном эксперименте с управлением въездами также перекрываем въезды вплоть до 80% в зависимости от плотности автомобилей на магистрали. Результаты моделирования, число въехавших автомобилей и график временных потерь при проезде по магистрали изображены на рис. 8, 9 и 10 соответственно.

На графиках видно уменьшение времени затора на МКАД, а также небольшое увеличение числа проехавших автомобилей. Хотя число проехавших по МКАД автомобилей увеличилось незначительно, временные потери на проезд по автомагистрали сильно снизились, а временной интервал затрудненного движения уменьшился. Интегральная разность между графиками на рис. 14 и 17 составляет чуть более 18 мин.

9. Эксперименты с высокой загрузкой с длинными въездами

В данной группе экспериментов функции входного потока соответствуют потоку в предыдущем эксперименте и изображены на рис. 11. В данном случае есть два типа въездов на автомагистраль — с утренней и вечерней пиковыми загрузками в течение трех часов. Въезды на автомагистраль — все протяженностью в 6 км в отличие от уже проведенных экспериментов, в которых их длина бралась 2 км.

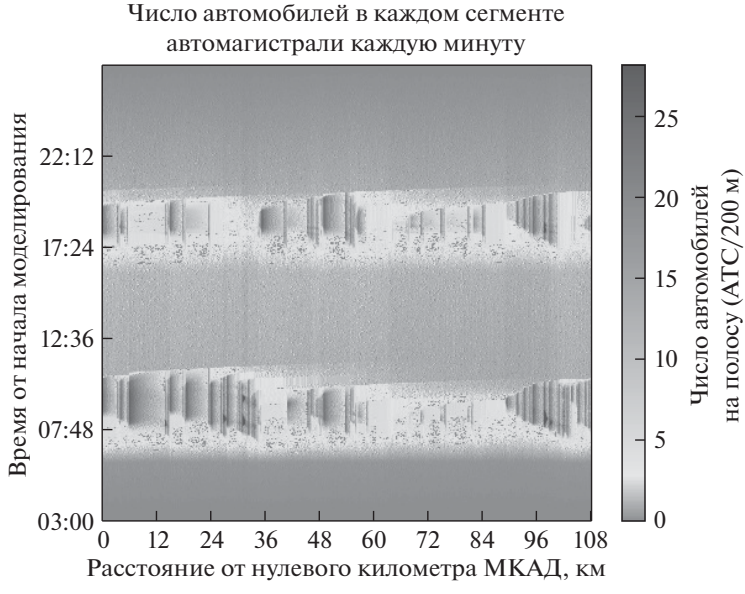

Рис. 15. Количество автомобилей на полосе в модели транспортной сети за день в эксперименте с высокой загрузкой с управлением въездами.

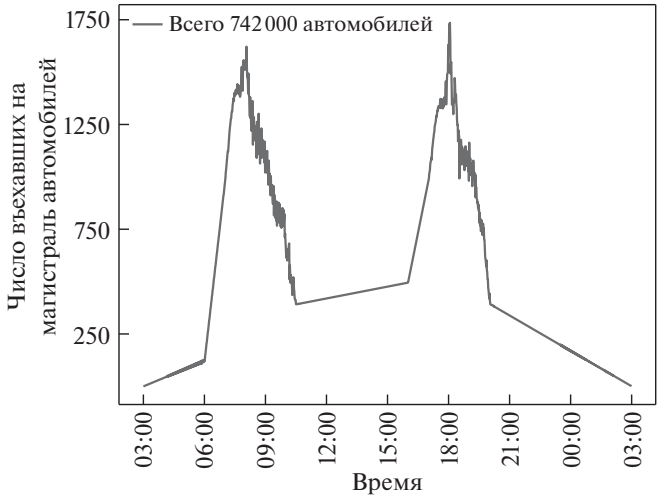

Рис. 16. График суммарно въехавшего на автомагистраль со всех въездов числа автомобилей в эксперименте с высокой загрузкой с управлением въездами.

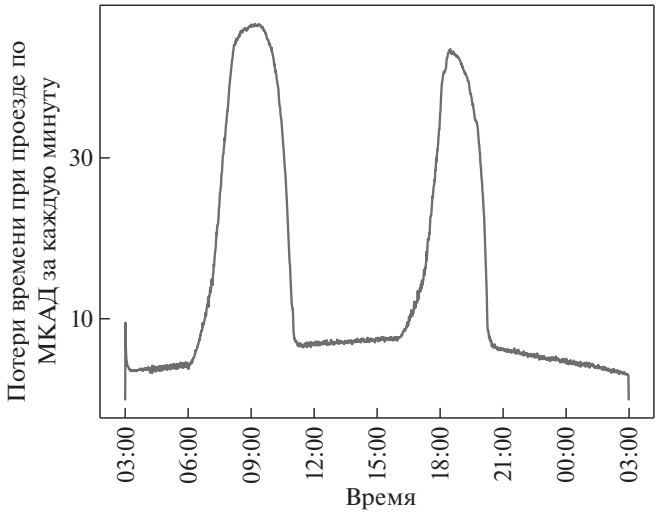

Рис. 17. Временные потери на проезд по автомагистрали в эксперименте с высокой загрузкой с управлением въездами.

В данном разделе приведем все результирующие графики парами. Результаты моделирования представлены на рис. 18. Число реально въехавших автомобилей и количество проехавших за день по магистрали АТС изображены на рис. 19. График временных потерь проезда по всей автомагистрали представлен на рис. 20. График временных потерь въезда на автомагистрали относительно пустой транспортной сети показан на рис. 21.

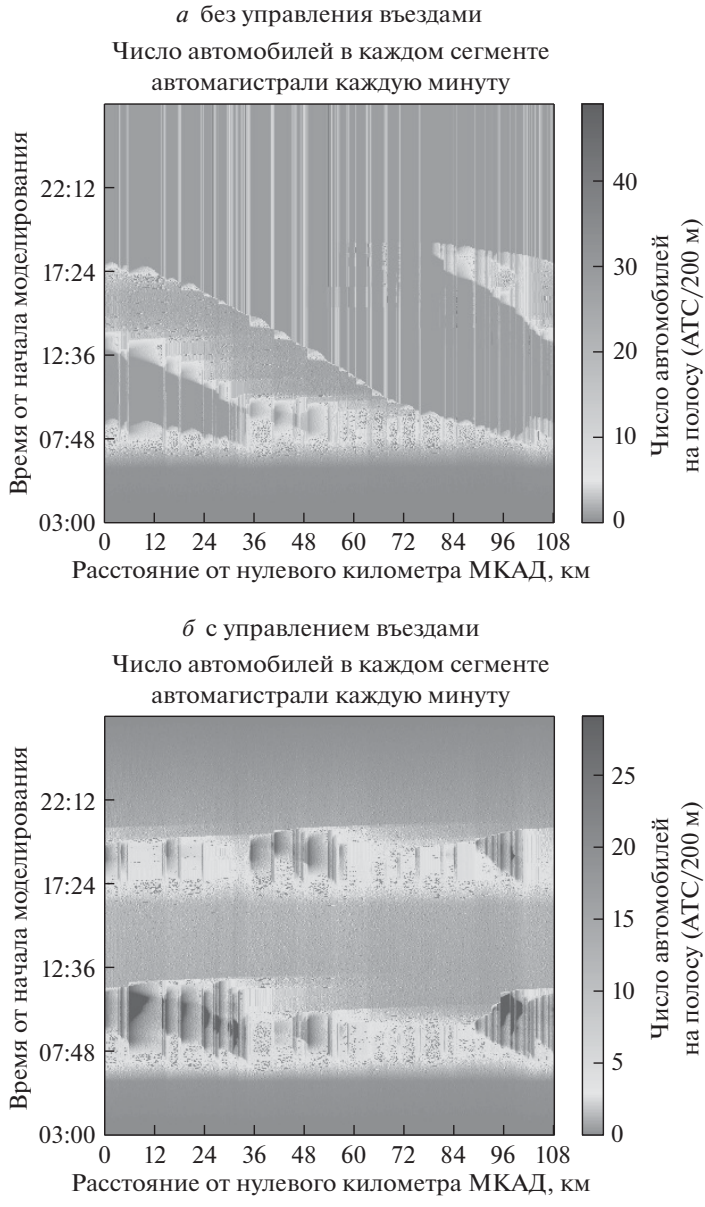

Рис. 18. Количество автомобилей на полосу в модели транспортной сети за день в эксперименте с высокой загрузкой.

В эксперименте видно, что из-за большого числа автомобилей, ожидающих въезда на МКАД без управления въездами, автомагистраль полностью забивается и не успевает освободиться до конца моделирования. Поскольку динамическое управление въездами не позволяет пробке поддерживаться за счет ограничения входного потока на магистраль, наблюдаются значительное улучшение состояния магистрали и сильная локализация затора во времени.

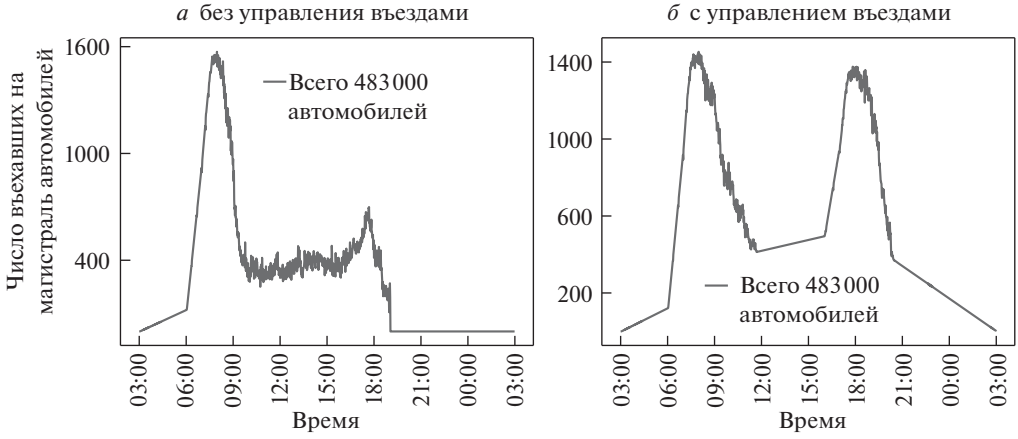

Рис. 19. Графики суммарно въехавшего на автомагистраль со всех въездов числа автомобилей в эксперименте с высокой загрузкой.

Интегральная разность между графиками на рис. 21 составляет чуть более 26 мин. Однако имеет смысл не учитывать сугубо экстремальную вечернюю ситуацию с практически полной остановкой автомагистрали. В этом случае при расчете интеграла до 15:00 задержка ожидания проезда по МКАД составит около 7 мин за половину суток.

10. Обсуждение результатов

В работе проведена проверка работоспособности статистической модели для задачи моделирования транспортных потоков в сети большой протяженности.

На графиках рис. 5, 12 видно, что модель адекватно ведет себя при образовании заторов и моделирует их образование, распространение по магистрали и рассасывание.

Графики на рис. 8, 15 показывают возможность локализации заторного движения на магистрали при наивном управлении ее въездами, что позволяет быстрее вывести ее в оптимальный режим. Преимущества управления явно показаны на графиках рис. 7, 10, 14, 17 ввиду существенного уменьшения времени проезда по МКАД. Заметим, что, например, на рис. 14 временные потери достигают 430 мин, или 7 ч, что соответствует скорости в 15 км/час. В реальности такое время проезда по всей автомагистрали, конечно же, не наблюдается ввиду того, что пробка через несколько часов уже пропадает и движение становится более свободным. На рис. 20, 21 видны временные потери на проезд по МКАД и на въезд на автомагистраль при высокой загрузке въездов шестикилометровой протяженности. Заметим, что в данном эксперименте магистраль полностью останавливается ввиду того, что в модели не учитываются изменения поведения водителей из-за существенной загруженности МКАД. В такой экстремальной ситуации автомобилисты начнут

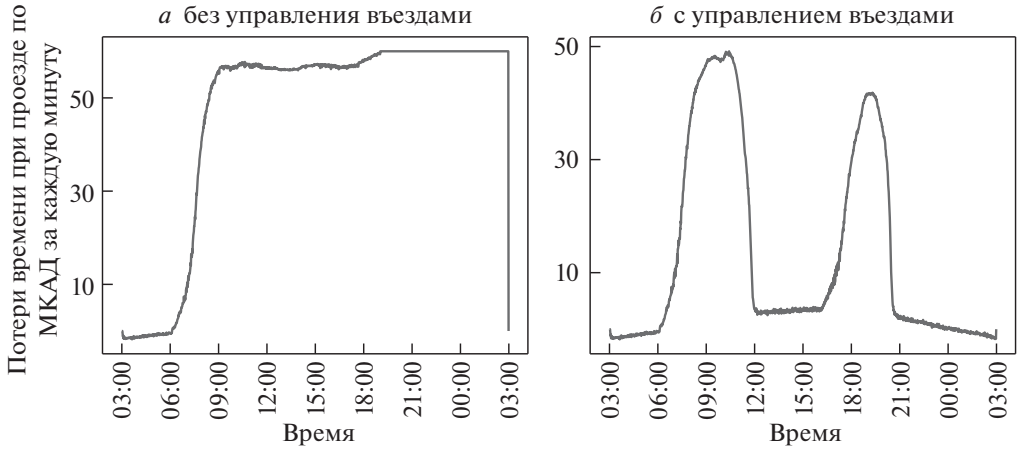

Рис. 20. Временные потери на проезд по автомагистрали в эксперименте с высокой загрузкой.

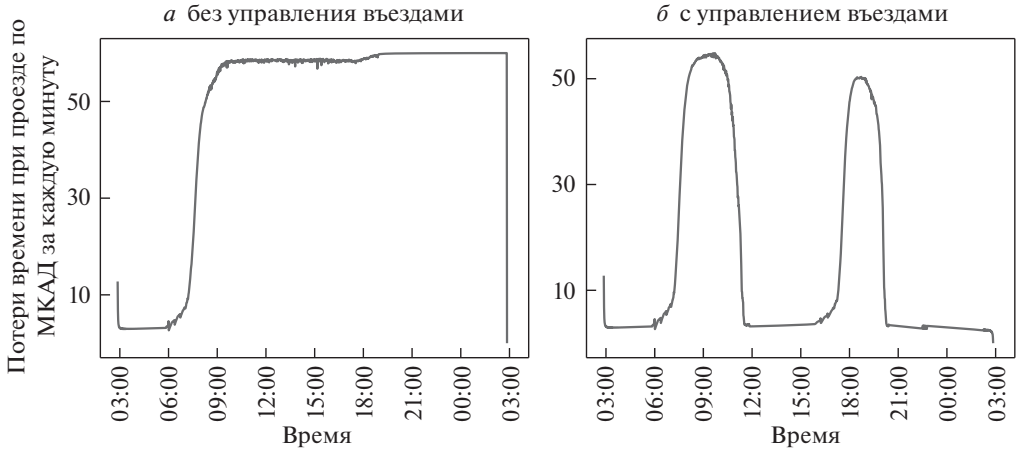

Рис. 21. Временные потери на въезд на автомагистраль в эксперименте с высокой загрузкой.

больше съезжать с МКАД, а системы построения маршрутов будут пытаться прокладывать пути без въезда на загруженную магистраль. Из-за этого эксперимент с управлением въездами, не дающий автомагистрали перейти в полностью заторное состояние, показывает существенный выигрыш как по времени проезда по магистрали, так и по времени въезда на нее.

На графиках рис. 6, 9, 13, 16 видно увеличение числа проехавших по магистрали АТС при использовании контроля въездов. Число проехавших по магистрали автомобилей во всех экспериментах соответствует данным от ЦОДД в 1,36 млн автомобилей за сутки по всему МКАД или приблизительно 680 тысяч автомобилей по одной стороне.

11. Заключение

Получен инструмент для моделирования транспортной сети большой протяженности за небольшое время. В однопоточном режиме моделирование одних суток на одной половине МКАД занимает 15 мин. Рассмотрены различные варианты входных потоков на автомагистраль и их влияние на образование и распространение заторного режима. Показано на примере МКАД, что управление въездами может быть использовано для локализации пробок на автомагистрали и увеличения ее пропускной способности.

Проведена проверка адекватности работы модели на небольших сегментах автомагистрали в различных конфигурациях дорог и входных потоков в [1], а также ее адекватности при моделировании большой дорожно-транспортной сети в данной работе. Ввиду вычислительной простоты модель может быть использована для онлайн-моделирования магистрали для различных целей, в том числе для управления въездами на нее.

СПИСОК ЛИТЕРАТУРЫ

1. *Старожилец В.М., Чехович Ю.В.* Об одном подходе к статистическому моделированию транспортных потоков // Журн. вычисл. матем. и мат. физики. 2021. № 7. С. 1220–1232.
2. *Yuta A., Nobuyasu I., Hajime I., Tetsuo I., Uchitane T.* Traffic simulation of Kobe-city // Proceedings of the international conference on social modeling and simulation, plus Econophysics Colloquium 2014. Berlin: Springer. 2015. V. 229. P. 255–264.
3. *Bieker L., Krajzewicz D., Morra A., Michelacci C., Cartolano F.* Traffic simulation for all: a real world traffic scenario from the city of Bologna // Modeling Mobility with Open Data. Berlin: Springer, 2015. V. 229. P. 47–60.
4. *Алексеенко А.Е., Холодов Я.А., Холодов А.С. и др.* Разработка, калибровка и верификация модели движения трафика в городских условиях. Ч. I // Компьютерные исследования и моделирование. 2015. Т. 7. № 6. С. 1185–1203.
5. *Гасников А.В., Кленов С.Л., Нурминский Е.А. и др.* Введение в математическое моделирование транспортных потоков. М.: Litres, 2015. С. 89.
6. *Guo Hong-Wei, Gao Zi-You, Zhao Xiao-Mei, Xie Dong-Fan.* Dynamics of motorized vehicle flow under mixed traffic circumstance // Communications in Theoretical Physics, 2011. V. 55. № 4. P. 719.
7. *Gundaliya P.J., Mathew T.V., and Dhingra S.L.* Heterogeneous traffic flow modelling for an arterial using grid based approach // J. Advanced Transport. 2008. V. 42. № 4. P. 467–491.
8. *Lan L.W., Chang C.-W., Gangopadhyay S.* Inhomogeneous cellular automata modeling for mixed traffic with cars and motorcycles // J. Advanced Transport. 2005. V. 39. № 3. P. 323–349.
9. *Payne H.J.* Models of freeway traffic and control // Math. Models Public Syst. 1998. № 4. P. 51–61.
10. *Zhang M.* Anisotropic property revisited — does it hold in multi-lane traffic? // Transport. Res. B Meth. 2003. V. 37. № 6. P. 561–577.
11. *Siebel F., Mauser W.* On the fundamental diagram of traffic flow // SIAM J. Appl. Math. 2006. V. 66. № 4. P. 1150–1162.

12. *Холодов Я.А., Алексеенко А.Е., Холодов А.С. и др.* Разработка, калибровка и верификация модели движения трафика в городских условиях. Ч. II // Компьютерные исследования и моделирование. 2015. Т. 7. № 6. С. 1205–1219.
13. *Alekseenko A.E., Kholodov Y.A., Kholodov A.S. et. al.* Adaptive traffic light control on highway entrances // 2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC). IEEE. 2017. P. 1–6.
14. *El-Tantawy S., Abdulhai B., Abdelgawad H.* Multiagent reinforcement learning for integrated network of adaptive traffic signal controllers (MARLIN-ATSC): methodology and large-scale application on downtown Toronto // IEEE Transactions on Intelligent Transportation Systems. 2013. Т. 14. № 3. P. 1140–1150.
15. *Mannion P., Duggan J., Howley E.* An experimental review of reinforcement learning algorithms for adaptive traffic signal control // Autonomic road transport support systems. 2016. P. 47–66.
16. *Gayah V.V., Gao X.S., Nagle A.S.* On the impacts of locally adaptive signal control on urban network stability and the macroscopic fundamental diagram // Transportation Research Part B: Methodological. Elsevier. 2014. Т. 70. P. 255–268.
17. *Han K., Liu H., Gayah V.V., Friesz T.L., Tao T.* A robust optimization approach for dynamic traffic signal control with emission considerations // Transportation Research Part C: Emerging Technologies. Elsevier. 2016. Т. 70. P. 3–26.
18. *Oskarbski J., Kaszubowski D.* Applying a Mesoscopic Transport Model to Analyse the Effects of Urban Freight Regulatory Measures on Transport Emissions–An Assessment // Sustainability. Multidisciplinary Digital Publishing Institute. 2018. V. 10. № 7. P. 2515.
19. *Tordeux A., Lämmel G., Hänseler F., Steffen B.* A mesoscopic model for large-scale simulation of pedestrian dynamics // Transportation research part C: emerging technologies. Elsevier. 2018. V. 93. P. 128–147.
20. Центр организации дорожного движения, число АТС на МКАД. https://twitter.com/gucodd/status/1135591064453222401.

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

Поступила в редакцию 21.01.2021

После доработки 04.06.2021

Принята к публикации 30.06.2021

© 2021 г. В.А. ТОКАРЕВА (victoria.tokareva@kit.edu)
(Технологический институт Карлсруэ, Германия)

РАСПИСАНИЯ С ПРИОРИТЕТАМИ ДЛЯ ЗАДАЧ ОНЛАЙН-УПРАВЛЕНИЯ РЕСУРСАМИ В СИСТЕМЕ АГРЕГИРОВАННОГО ДОСТУПА К ДАННЫМ[1]

Исследуется задача управления ресурсами в системе агрегированного доступа к данным на примере облачного хранилища, предназначенного для управления данными научных экспериментов астрофизики частиц средних и малых размеров. На основе подходов теории массового обслуживания и теории расписаний сформулирована и исследована математическая модель рассматриваемых процессов. Выполнено имитационное моделирование для нахождения эвристики, которая минимизирует сумму критериев расписания. Показано, что применение данной вычислительной дисциплины устойчиво показывает лучшие результаты по сравнению с "наивными" подходами по ряду рассматриваемых параметров.

Ключевые слова: онлайн-планирование, управление данными, управление ресурсами, распределенные вычисления, облака данных, открытые исследования, компьютерные расписания, моделирование.

DOI: 10.31857/S000523102111009X

1. Введение

Для современной науки характерен тренд на междисциплинарные исследования и свободный обмен научными данными. Создание инструментов и платформ, агрегирующих доступ исследователей к научным данным, является объектом внимания многочисленных международных проектов, в том числе с участием отечественных специалистов.

В настоящее время хранение и обработка больших объемов разнородных данных вызывает большой интерес со стороны бизнеса и исследователей. Актуальные методы анализа данных, в частности методы машинного и глубокого обучения, зачастую показывают большую точность при обучении на увеличенной выборке. Использование новейших методов анализа позволяет также получать новые результаты по архивным данным, которые не могли быть получены ранее в связи с техническими и методологическими ограничениями. Совместный анализ данных, полученных в различных экспериментах, помогает проверять точность методов, калибровать детекторы, увеличивать точность анализа вновь собранных данных, извлекать принципиально новое

[1] Работа выполнена при финансовой поддержке совместного гранта Российского научного фонда (№ 18-41-06003) и Общества Гельмгольца (№ HRSF-0027).

научное знание, недоступное ранее. Так, например, одним из наиболее молодых и многообещающих направлений на стыке современных астрономии, астрофизики и физики частиц является так называемая многоканальная астрономия (multi-messenger astronomy) — область науки, комплексно изучающая данные об электромагнитном излучении, гравитационных волнах и элементарных частицах, таких как нейтрино и космические лучи высокой энергии, с целью получения сведений о происходящих в космосе процессах [1]. Сходный запрос на междисциплинарные исследования и коллаборацию между научными экспериментами, в частности экспериментами малого и среднего размера, наблюдается и в других областях науки [2–5].

В то же время сводный анализ данных в таких коллаборациях зачастую оказывается технически перегруженным из-за отсутствия гибкой системы доступа к данным, которая бы предлагала пользователю единый интерфейс запросов и инкапсулировала решение технических задач, возникающих в процессе работы с удаленными хранилищами. Для мега-экспериментов (ATLAS [6], CMS [7] и т.д.), в контексте которых чаще всего обсуждается управление данными в науке, можно выделить популярные решения управления данными, такие как GRID [8] и Hadoop [9]. Данные решения стали де-факто стандартами в этой области, но они зачастую оказываются плохо применимыми для агрегации данных экспериментов меньшего размера. На настоящий момент поиском решений в данной области заняты такие проекты, как [2–4, 10–13].

В качестве недостатка перечисленных подходов можно выделить незначительное количество практико-ориентированных математических моделей, подкрепляющих принятые при разработке инженерные решения и предоставляющих возможность аргументированного исследования и оптимизации характеристик рассматриваемой системы.

В данной статье предлагается динамическая модель онлайн обработки пользовательских заявок сервером для случая агрегированных запросов к K хранилищам со стохастическими элементами, являющая развитием предыдущих исследований автора [14, 15]. Описаны вероятностные и функциональные зависимости в системе. Приводятся результаты имитационного моделирования, нацеленного на поиск обслуживающей дисциплины, оптимизирующей общее время пользователя в системе.

Статья имеет следующую структуру: в разделе 2 приводится обзор литературы и вводятся основные понятия; в разделе 3 описываются постановка задачи и используемые методология и методы; в разделе 4 представлен ход исследования; в разделе 5 описываются основные выводы и направления дальнейшей работы.

2. Обзор литературы

Разрабатываемая облачная среда является системой массового обслуживания (СМО). Исследованием характеристик таких систем занимается теория массового обслуживания, в основе которой лежат работы таких за-

служенных математиков, как А.А. Марков [16, 17], А.Н. Колмогоров [18], Е.С. Вентцель [18], Л. Клейнрок [19], Т.Л. Саати [20], А.Я. Хинчин [21] и др.

Классическими моделями, хорошо изученными в литературе, являются модели с одним или несколькими каналами обслуживания [22], имеющими одинаковую производительность и имеющими распределение потоков [22] входящих и исходящих заявок, соответствующее так называемым марковским процессам (обозначаемым символом M). Также относительно хорошо изученными являются сравнительно простые модели с другими (G, GI, E_r) видами распределений потоков событий. Более сложные модели со связанными очередями исследуются в так называемых сетях массового обслуживания [23–25]. Публикации данного направления опираются на классическую статью Дж.Р. Джексона [26], где доказано, что результаты, справедливые для СМО типа $M/M/m/\infty$, можно применить при расчете стационарного (установившегося) режима работы разомкнутой (открытой) сети СМО. Использование указанного математического аппарата в дальнейшем развитии работы, описанной в данной статье, является одной из перспективных задач, которую ставит перед собой автор.

Время пребывания пользователя в системе определяется суммой времени ожидания в очереди и времени непосредственного обслуживания заявки. Оптимизация первого параметра возможна с помощью выбора разумной дисциплины обслуживания заявки и при использовании параллелизации подзадач, выполняемых в рамках обработки заявки [14], что приводит также к сокращению ожидаемого времени обслуживания.

Будем понимать дисциплину обслуживания как некое расписание обработки пользовательских задач. Алгоритмы составления оптимальных расписаний изучаются в рамках теории расписаний и описаны как в классических работах [18, 19, 27], так и в современных работах [28, 29]. Задачи составления расписаний можно разделить по рассматриваемым характеристикам системы: задачи с одним или многими серверами, с распределением ресурсов, задачи online и real time обработки, а также по типу характера распределения операций по обслуживающим серверам (job shop, flow shop, open shop, mixed shop).

Задачи составления онлайн расписаний изучались в [30–32]. Рассматриваемая в данной статье задача построения расписаний для многопроцессорных систем с конфликтами доступа к ресурсу рассматривалась в публикациях Błażewicz и соавт. [33–35], а также в [34, 36]. В [15] автором давались комментарии об условиях применимости результата [33] в достижении цели исследования.

В практических задачах составления онлайн расписаний зачастую используются [29] аппроксимированные алгоритмы, такие как алгоритмы локального поиска, генетические алгоритмы, эвристики декомпозиции, правила приоритетного обслуживания (priority dispatching rules, PDR). Данные подходы позволяют получить решение в заданных временных рамках. Важным свойством для моделирования горизонтально масштабируемых систем является масштабируемость вычислительных алгоритмов. Исследование [37] ука-

зывает на то, что эвристические алгоритмы, такие как эволюционные, некоторые алгоритмы локального поиска (Taboo search, Simulated annealing) могут быть уязвимы с точки зрения масштабируемости, и делает вывод, что наиболее успешно масштабируются эвристики, обладающие линейной вычислительной сложностью, такие как PDR алгоритмы. Такие достоинства этих эвристик, как вычислительная простота и хорошая масштабируемость, приводят к частому применению данных эвристик в практических задачах онлайн обработки событий. Публикации [38, 39] показывают, что в то время как ни одно из правил этой группы не превосходит другую в целом, однако для конкретных моделей исследуемых процессов и критериев оптимальности оказывается возможным выбор искомой дисциплины обслуживания.

3. Общая формулировка проблемы

На рис. 1 показана схема распределенной системы агрегирования и обработки данных GRADLCI. На схеме $S_1, \ldots, S_3$ обозначают удаленные хранилища, которые используются для извлечения агрегированных данных с быстрым поиском событий с использованием базы данных метаданных (MDDB) [12], после чего пользователь может начать обработку выбранных данных в виртуальной среде на сервере приложений.

Далее смоделируем доступ пользователя к серверу приложений и обработку запросов этим сервером. Запросы, отправляемые пользователем на сервер, могут адресоваться одному хранилищу или иметь агрегированный характер и использовать несколько из них. Рассмотрим основные этапы обработки агрегированного пользовательского запроса.

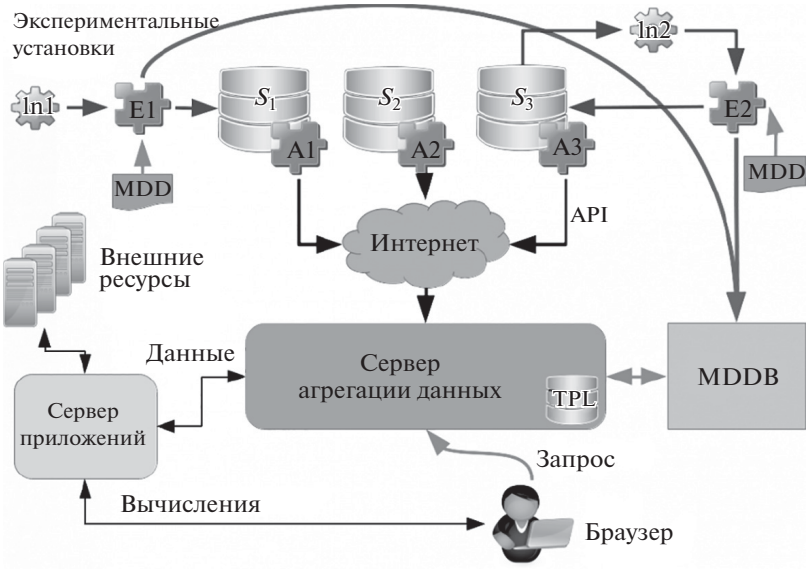

Рис. 1. Архитектура платформы GRADLCI [12].

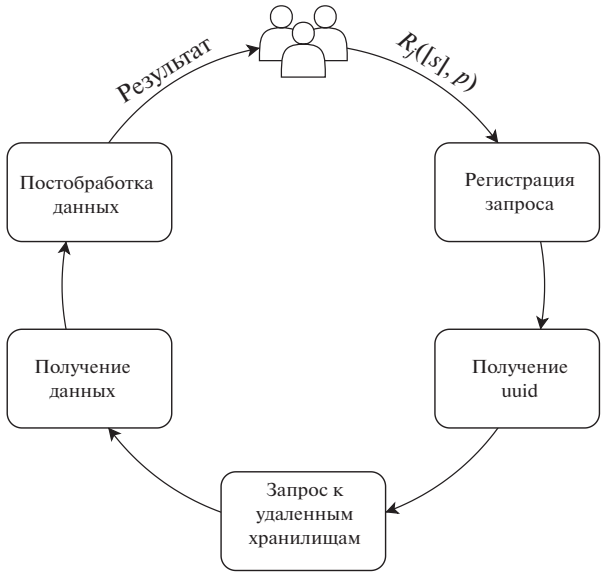

Рис. 2. Цикл обработки агрегированного запроса.

На рис. 2 пользователь отправляет запрос агрегатору и этот запрос ставится в очередь на обработку. Выполнение запроса включает в себя отправку запроса в MDDB для получения уникальных идентификаторов записей (uuids), затем доступ к хранилищам, получение необходимых записей из хранилища, индивидуальную постобработку запроса на стороннем агрегаторе и последующую совместную постобработку. Поскольку сбор данных из разных хранилищ может занимать разное время и влиять на скорость работы системы и эффективность обслуживания клиентов, актуальной проблемой является балансировка запросов в очереди и составление превентивного расписания обработки задач с использованием распределенных ресурсов. Далее построим математическую модель описанных процессов и сформулируем задачу поиска оптимального расписания обработки запросов пользователей.

Пусть $\mathbb{S} = \{S_1, S_2, \ldots, S_K\}$, $K \in \mathbb{N}$ — набор удаленных гетерогенных хранилищ (возобновляемых ресурсов), подключенных к системе агрегации данных. Каждое хранилище представлено одним экземпляром в системе и может быть доступно не более чем одному процессу одновременно. Пусть $[s]$ — подмножество $\mathbb{S}$, а $[k]$ — подмножество соответствующих индексов хранилищ. Пусть p — вектор параметров запроса, принадлежащих множеству $S_1 \bigcup \cdots \bigcup S_K$.

Пусть $R_j([s]_j, p_j)$ — это запрос пользователя к системе, где $j \in [1, \ldots, J]$, $J \in \mathbb{N}$ — порядковый номер запроса к системе, $[s]_j$ — подмножество хранилищ, для которых нужно выполнить агрегированный запрос, а p_j — параметры запроса. Обозначим время поступления запросов через r_j. Для статической модели полагалось $r_j = 0 \ \forall j$.

Предположим, что в системе имеются m идентичных процессоров $P = \{P_1, \ldots, P_m\}$, способных параллельно обрабатывать запросы. Прерыва-

ния в обработке задач запрещены. Предположим, что для каждого запроса $R_j([s]_j, p_j)$ можно назначить вектор $T_j = \{T_{j1}, \ldots, T_{jk}\}$, $k \leqslant m$, приближенных оценок времени T_{jk} пребывания подзапросов к отдельным хранилищам в системе. Для времени пребывания в системе T в литературе также встречается обозначение W_s.

Таким образом, можно записать задачу поиска оптимального расписания обработки запросов в виде

$$\left\langle P_m \mid resK11, T_j \in [\underline{t_j}, \overline{t_j}] \mid \sum_j T_j \right\rangle. \tag{1}$$

Обработка запроса в системе происходит по алгоритму, рассмотренному в [14]. Приведем описание статической модели обработки агрегированного запроса, сформулированной в [15].

В [14] показано, что скорость выполнения отдельных операций линейно зависит от количества записей n_{jk}, извлеченных из удаленных хранилищ, поэтому ключевой момент для оценки общего времени выполнения операций — это процедура оценки количества записей n_{jk}, соответствующих запросу $R_j([s]_j, p_j)$ для каждого k-го хранилища из $[s]_j$. Очевидный подход к этому вычислению — выполнить запрос на подсчет количества записей к MDDB. Однако время выполнения такого запроса может значительно отличаться и иногда замедлять обработку задачи. Для снижения нагрузки на хранилище метаданных и ускорения процедуры оценки предлагается оценивать количество записей по квантилям параметров, хранящихся на сервере агрегации. В этом случае T^c, время оценки n_{jk}, пренебрежимо мало.

Пусть T^w_{jk} будет временем ожидания в очереди j-го подзапроса, использующего ресурс k. Для времени ожидания T^w в литературе также встречается обозначение W_q. Общее время обработки предыдущего подзапроса будет считаться равным $T_{j-1,k}$. Тогда время, необходимое для выполнения этого шага, будет:

$$T^w_{jk} = \begin{cases} T^c, & j = 1, \\ T_{j-1,k}, & j > 1. \end{cases} \tag{2}$$

Эффективное время для инициализации запроса к MDDB будет рассчитываться как:

$$T^i_{jk} = \begin{cases} 0, & t^i_{jk} \leqslant T^p_{j-1,k}, \\ t^i_{jk} - T^p_{j-1,k}, & t^i_{jk} > T^p_{j-1,k}, \end{cases} \tag{3}$$

где $t^i_{jk} \in \mathrm{unif}(0, \Theta_i)$, $\Theta_i \in \mathbb{R}$ — фактическое время инициализации запроса, $T^p_{j-1,k}$ — время обработки $(j-1)$-го запроса.

В [14] приведены следующие зависимости:

$$\begin{cases} t_s(n_{jk}) = \nu \cdot n_{jk}, & \nu \in \mathbb{R}, \\ t_f(n_{jk}) = \mu_k \cdot n_{ik}, & \mu = (\mu_1, \ldots, \mu_k) \in \mathbb{R}^s, \\ t_a(n_{jk}) = \tau \cdot n_{jk}, & \tau \in \mathbb{R}, \end{cases} \tag{4}$$

где ν — скорость обработки MDDB, μ_k — скорость обработки информации для хранилища s_k, τ — скорость постобработки события на стороне агрегатора, а t_s, t_f, t_a являются промежуточными обозначениями для времени операций поиска метаданных, извлечения данных из удаленного хранилища и индивидуальной постобработки соответственно. Эти операции могут выполняться параллельно небольшими частями, и общее время выполнения этих шагов можно рассчитать с помощью уравнения

$$T^p_{jk} = n_{jk} \cdot \max(\nu, \mu_k, \tau). \tag{5}$$

Время совместной постобработки событий агрегатором можно оценить как:

$$T^{pp}_j = \sum_k \xi_k \cdot n_{jk}, \quad \xi \in \mathbb{R}^k. \tag{6}$$

Таким образом, требуемое значение T_j можно рассчитать с помощью уравнения:

$$T_j = \max_k \left(T^w_{jk} + T^i_{jk} + T^p_{jk} \right) + T^{pp}_j. \tag{7}$$

4. Результаты и анализ

Существенным ограничением приведенной модели являлось то, что она является статической и описывает простой случай, когда все заявки считаются поступившими на сервер одновременно в момент времени $t_j = 0 \ \forall j$. Для практических задач характерно динамическое поступление запросов, которое вносит в модель стохастические компоненты. Также вероятностным является выбор пользователем сочетания k из K ресурсов, которое считалось случайным, независимым и равномерно распределенным, т.е. выбиралось сочетание C^k_K с вероятностью использования каждого хранилища q_k. Из каждого хранилища извлекается число событий n_{jk}, удовлетворяющих параметрам запроса p_j. Таким образом, задаче соответствует подмножество используемых хранилищ $[s]_j$ и количеств извлекаемых из каждого хранилища событий $[n]_j$. Ожидаемое число запросов в короткий интервал времени распределено по Пуассону, соответственно интервал времени между запросами определяется экспоненциальным распределением. Однако частота запросов на длинных промежутках времени может изменяться. В данной статье при моделировании был принят вариант, когда частота меняется от времени суток по синусоиде

$$r_j = r_{\min} + \frac{1 - \cos(2\pi t/\tau)}{2}(r_{\max} - r_{\min})$$

с минимумом ночью и максимумом днем по внутреннему времени модели. Здесь τ — период колебаний, $t \in \mathbb{R}$ — текущее время.

Поскольку задача состоит в обеспечении онлайн обработки запросов, вероятностные характеристики были внесены в модель в упрощенном виде. Более

точное исследование вероятностных характеристик системы является одним из направлений дальнейших исследований автора.

Далее, была поставлена задача поиска PDR-эвристики, которая бы минимизировала сумму критериев оптимальности расписания. Были рассмотрены следующие PDR:

- Обслуживание в порядке поступления требований — First In First Out — FIFO;
- Обслуживание в порядке, обратном порядку поступления требований — Last In First Out — LIFO;
- Кратчайшее время обработки — Shortest Processing Time — SPT;
- Наибольшее время обработки — Longest Processing Time — LPT;
- Кратчайшее полное время обработки — Shortest Total Processing Time — STPT;
- Наибольшее полное время обработки — Longest Total Processing Time — LTPT;

и критерии оптимальности расписания:

- Время ожидания (Waiting Time, WT) задачи j в очереди i;
- Время обработки (Processing Time, PT): время, требуемое для обработки задачи j с использованием ресурса i;
- Полное время обработки (Total Processing Time, TPT): полное время пребывания задачи j в системе от момента ее поступления до полного окончания обработки (для всех очередей и ресурсов).

Выбор критериев оптимальности производился из критериев, представленных в [37–39], и обусловлен их применимостью в рассматриваемой задаче.

Было произведено математическое моделирование для каждого из PDR по следующему алгоритму. Число хранилищ при моделировании варьировалось от двух до 10. Задачи, генерируемые при моделировании, обращались к каждому из хранилищ с вероятностью $q_k = 50\,\%$ и затребовали из хранилища число событий, распределенное равновероятно от нуля до максимального количества. Использовавшиеся в моделировании хранилища эмулировали хранилище эксперимента KASCADE [40], способное обработать $r_{\max} = 0{,}75$ запросов в час при условии равномерного распределения числа событий. Было промоделировано поведение системы за месяц для различных средних частот запросов r_{ave} от $0{,}1r_{\max}$ до $0{,}9r_{\max}$. Интервалы между событиями считались случайно распределенными в соответствии с экспоненциальным распределением, параметр которого варьировался в зависимости от времени системы от $0{,}2r_{\text{ave}}$ до $1{,}8r_{\text{ave}}$ в период наименьшей и наибольшей загруженности соответственно.

Моделировалась работа системы в течение условного месяца (30 моделируемых суток). Значения критериев были усреднены по 100 запускам для каждого сочетания моделируемых параметров. Новые задачи генерировались в случайные моменты времени, распределенные экспоненциально с параметром распределения r, варьирующимся в течение модельных суток. Обслуживание заявок из очереди производилось в соответствии с выбранными приори-

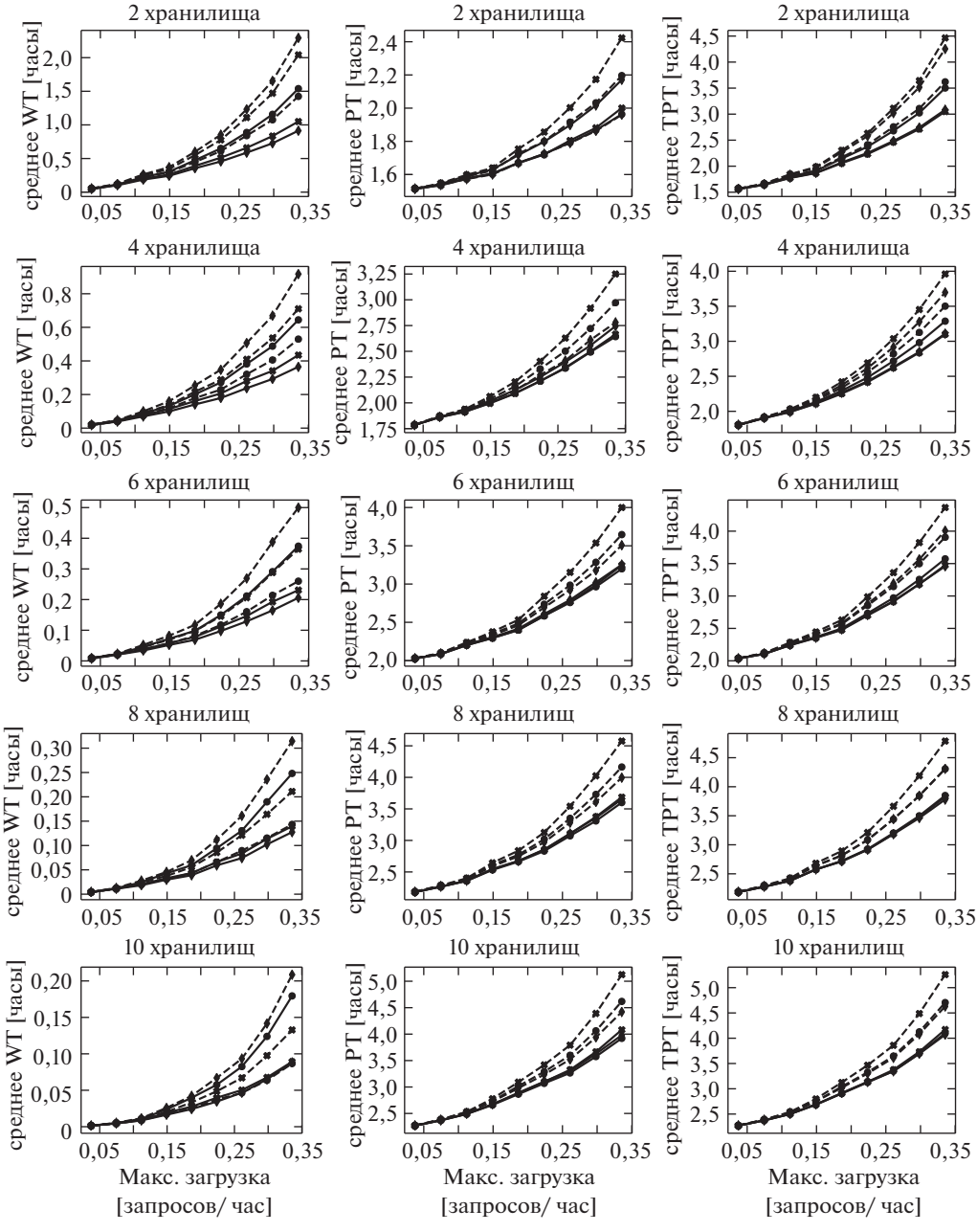

Рис. 3. Результаты моделирования времен-критериев оптимальности расписания для различных правил определения приоритета, количества хранилищ и средних частот запросов. Для маркировки дисциплин обслуживания использованы следующие обозначения. Сплошная линия: круглый маркер — FIFO, крест — SPT, ромб — STPT; штриховая линия: круглый маркер — LIFO, крест — LPT, ромб — LTPT.

тетами. Время обслуживания оценивалось согласно приведенным ранее критериям. Результаты моделирования приведены на рис. 3.

На графиках, приведенных на рис. 3, показаны зависимости значений критериев оптимальности от интенсивности входящего потока заявок для различных PDR. Наименьшее увеличение значения критерия с ростом загруженности системы является показателем удачности выбора PDR.

Можно заметить, что при использовании одного и того же критерия оптимальности увеличение количества хранилищ, подключенных к агрегатору, оказывает влияние на то, какая дисциплина обслуживания оказывается более выгодной. Так, например, в случае двух хранилищ по показателю PT дисциплина LTPT показывает наилучший результат, тогда как с увеличением количества хранилищ данная дисциплина начинает "проигрывать" таким дисциплинам, как STPT и SPT по рассматриваемому критерию.

Выбор критерия оптимальности также влияет на то, какая дисциплина окажется наиболее выгодной для распределения задач в системе. Так, в случае двух хранилищ можно заметить, что дисциплина LTPT оказывается самой выгодной по критерию PT, тогда как по критериям WT и TPT эта стратегия оказывается одной из самых неудачных. Таким образом, конфигурация системы массового обслуживания и выбор критериев оптимизации обслуживания являются существенным фактором, который следует учитывать при выборе используемого PDR.

Вместе с тем для рассматриваемой в рамках данной статьи СМО в случаях, представленных на рис. 3, можно изучить, насколько часто та или иная дисциплина обслуживания оказывается "лучшей" или "худшей" по рассматриваемым критериям. Так, линии, соответствующие дисциплинам LPT и LTPT, оказываются в верхней части графика (среди худших результатов) более чем в 2/3 случаев, что позволяет сделать вывод о том, что применение данных дисциплин является невыгодным в рассмотренных условиях. В то же время дисциплина STPT оказалась среди "лучших" в 0,93% случаев (для некоторых из этих случаев близкий к ней результат показывают FIFO и SPT), что может служить рекомендацией к ее использованию для решения данных классов задач. Качественно данный результат можно объяснить тем, что события, которые стоят в начале очереди, вносят больший вклад в сумму при последующих расчетах.

Результаты данного анализа могут быть использованы при разработке оригинальных планировщиков задач в информационно-вычислительных системах (ИВС) рассмотренной архитектуры и при настройке существующих планировщиков задач в ИВС реального времени.

5. Заключение и выводы

В данной статье была развита динамическая модель онлайн обработки пользовательских заявок сервером для случая агрегированных запросов к K хранилищам за счет добавления стохастической компоненты. Было произведено имитационное моделирование с целью поиска обслуживающей дис-

циплины, позволяющей сократить время обслуживания пользователя в системе.

Развитие данной работы рассматривается в направлении использования математического аппарата сетей массового обслуживания для более точного описания процессов, происходивших в системе. Также автором планируются оценка работы алгоритма на реальных экспериментальных данных и учет неравномерного характера запросов в реальной системе и аппроксимации недостающих измерений для запросов, связанных с определенными сочетаниями запрашиваемых ресурсов.

Автор выражает глубокую признательность коллегам из Научно-исследовательского института ядерной физики им. Д.В. Скобельцына Московского государственного университета им. М.В. Ломоносова (особенно А.П. Крюкову и М.Д. Нгуену), Института астрофизики частиц Технологического института Карлсруэ (в том числе А. Хаунгсу, Д. Вохеле, Ю. Вохеле, Д. Канг и Ф. Полгарту), Иркутского государственного университета и Института динамики систем и теории управления им. В.М. Матросова СО РАН (А.А. Михайлову и А.О. Шигарову) за плодотворную совместную работу в рамках проекта GRADLCI (APPDS) и ценные обсуждения.

СПИСОК ЛИТЕРАТУРЫ

1. *Branchesi M.* Multi-Messenger Astronomy: Gravitational Waves, Neutrinos, Photons, and Cosmic Rays // J. Physics: Conf. Series. 2016. V. 718. P. 022004.
2. *Quix C., Hai R., Vatov I.* GEMMS: A Generic and Extensible Metadata Management System for Data Lakes // CAiSE Forum. 2016. P. 129–136.
3. *Endris K.M., et al.* Ontario: Federated Query Processing Against a Semantic Data Lake / Database and Expert Systems Applications. Eds. Hartmann S., et al. Cham: Springer International Publishing, 2019. P. 379–395.
4. *Villanueva M.J., Valverde F., Levín A.M., Lopez O.P.* Diagen: A Model-Driven Framework for Integrating Bioinformatic Tools // Int. Conf. on Advanced Information Systems Engineering. Heidelberg: Springer, 2011. P. 49–63.
5. *Cohen-Boulakia S., Leser U.* Next Generation Data Integration for Life Sciences // IEEE 27th Int. Conf. on Data Engineering. 2011. P. 1366–1369.
6. *Branco M., et al.* Managing ATLAS Data on a Petabyte-Scale with DQ2 // J. Physics: Conf. Series. 2008. V. 119. P. 062017.
7. *Yzquierdo A.P.-C.* CMS Strategy for HPC Resource Exploitation // EPJ Web of Conferences. 2020. V. 245. P. 09012.
8. *Peters A.J., Janyst L.* Exabyte Scale Storage at CERN // J. Physics: Conf. Series. 2011. V. 331. P. 052015.
9. *Shvachko K., Kuang H., Radia S., Chansler R.* The Hadoop Distributed File System // IEEE 26th Sympos. on Mass Storage Systems and Technologies (MSST). 2010. P. 1–10.
10. *Ayris P., et al.* Realising the European Open Science Cloud. Luxembourg: European Union, 2016.
11. Innovative Digital Technologies for Research on Universe and Matter (ErUM Data IDT). 2020. URL: https://www.erum-data-idt.de/

12. *Bychkov I., et al.* Russian-German Astroparticle Data Life Cycle Initiative // Data. 2018. V. 3. P. 56.
13. *Bolton R., et al.* ESCAPE Prototypes a Data Infrastructure for Open Science // EPJ Web of Conferences. 2020. V. 245. P. 04019.
14. *Tokareva V.* Optimization of Request Processing Times for a Heterogeneous Data Aggregation Platform // J. Physics: Conf. Series. 2021. V. 1740. No. 1. P. 012058.
15. *Tokareva V.* Extended Static Model of User Requests Processing for a Heterogeneous Data Aggregation Platform with K Storages // Proc. of AYSS-2021 Conf. IOP Publishing. 2021. Accepted for publication: Dec. 13 2020.
16. *Чжун К.* Однородные цепи Маркова. М.: Мир, 1964.
17. *Кемени Д., Снелл Д.Л.* Конечные цепи Маркова. М.: Наука, 1970.
18. *Вентцель Е.С.* Исследование операций. М.: Сов. Радио, 1972.
19. *Kleinrock L.* Queuing Systems. N.Y.: Wiley, 1975.
20. *Saaty T.L.* Elements of Queueing Theory, with Applications. N.Y.: McGraw-Hill, 1961.
21. *Хинчин А.Я.* Избранные труды по теории вероятностей. М.: Науч. изд-во ТВП, 1995.
22. *Taha H.A.* Operations Research an Introduction. Pearson Education Limited 2017, 2017.
23. *Gelenbe E., Pujolle G.* Introduction to Queueing Networks. V. 2. N.Y.: Wiley, 1998.
24. *Bramson M.* Stability of Queueing Networks. Heidelberg: Springer, 2008.
25. *Baskett F., Chandy K.M., Muntz R.R., Palacios F.G.* Open, Closed, and Mixed Networks of Queues with Different Classes of Customers // J. the ACM (JACM). 1975. V. 22. No. 2. P. 248–260.
26. *Jackson J.R.* Networks of Waiting Lines // Oper. Res. 1957. V. 5. No. 4. P. 518–521.
27. *Бронштейн О.И., Духовный И.М.* Модели приоритетного обслуживания в информационно-вычислительных системах. М.: Наука, 1976. Т. 2976. С. 221.
28. *Лазарев А.А., Гафаров Е.Р.* Теория расписаний: задачи и алгоритмы. М.: МГУ, 2011.
29. *Błażewicz J., Ecker K., Pesch E., Schmidt G., Sterna M., Weglarz J.* Handbook on Scheduling. Cham: Springer International Publishing, 2019.
30. *Albers S.* Online Scheduling / Introduction to Scheduling. Eds. Robert Y., Vivien F. Boca Raton: Chapman & Hall/CRC Press, 2009. P. 51–78.
31. *Fiat A., Woeginger G.J.* Online Algorithms: The State of the Art. Heidelberg: Springer, 1998. V. 1442.
32. *Pruhs K., Sgall J., Torng E.* Online Scheduling / Hand-Book of Scheduling: Algorithms, Models, and Performance Analysis. Ed. Leung. J.Y.-T. Boca Raton: Chapman & Hall/CRC, 2004. P. 15.1–15.43.
33. *Błażewicz J., Kubiak W., Szwarcfiter J.* Scheduling Independent Fixed-Type Tasks // Advances in Project Scheduling. Elsevier, 1989. P. 225–236.
34. *Błażewicz J., Ecker K.* A Linear Time Algorithm for Restricted Bin Packing and Scheduling Problems // Oper. Res. Lett. 1983. V. 2. No. 2. P. 80–83.
35. *Błażewicz J., Lenstra J.K., Kan A.R.* Scheduling Subject to Resource Constraints: Classification and Complexity // Discrete Appl. Math. 1983. V. 5. No. 1. P. 11–24.
36. *Garey M.R., Johnson D.S.* Complexity Results for Multiprocessor Scheduling under Resource Constraints // SIAM J. on Computing. 1975. V. 4. No. 4. P. 397–411.

37. *Chen B., Matis T.I.* A Flexible Dispatching Rule for Minimizing Tardiness in Job Shop Scheduling // Int. J. Production Economics. 2013. V. 141. No. 1. P. 360–365.

38. *Rajendran C., Holthaus O.* A Comparative Study of Dispatching Rules in Dynamic Flowshops and Jobshops // Eur. J. Oper. Res. 1999. V. 116. No. 1. P. 156–170.

39. *Nguyen S., Zhang M., Johnston M., Tan K.C.* A Computational Study of Representations in Genetic Programming to Evolve Dispatching Rules for the Job Shop Scheduling Problem // IEEE Trans. Evol. Comput. 2013. V. 17. No. 5. P. 621–639.

40. *Wochele J., Wochele D., Haungs A., Kang D.* The KASCADE Cosmic-ray Data Centre KCDC: Releases and Future Perspectives // Proc. 4th Int. Workshop on Data Life Cycle in Physics. 2020. P. 143.

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

Поступила в редакцию 24.01.2021

После доработки 01.06.2021

Принята к публикации 30.06.2021

Оптимизация, системный анализ и исследование операций

© 2021 г. А.Ю. МАТРОСОВА, д-р техн. наук (mau11@yandex.ru),
С.В. ЧЕРНЫШОВ, (svchernyshov@mail.ru),
О.Х. КИМ, (oh.kim@yandex.ru),
Е.А. НИКОЛАЕВА, канд. техн. наук (nikolaeve-ea@yandex.ru)
(Национальный исследовательский Томский государственный университет)

ПОСТРОЕНИЕ ПОСЛЕДОВАТЕЛЬНОСТИ, ОБНАРУЖИВАЮЩЕЙ РОБАСТНО ТЕСТИРУЕМЫЕ НЕИСПРАВНОСТИ ЗАДЕРЖЕК ПУТЕЙ В СХЕМАХ С ПАМЯТЬЮ

Предлагается метод построения последовательности булевых векторов входных переменных, доставляющей тестовые пары (v_1, v_2) соседних векторов в пространстве входных и внутренних переменных для робастно тестируемых неисправностей задержек путей (робастных Path Delay Faults (PDFs)) в логических схемах с памятью. Целью данной работы является выяснение возможности построения тестовой последовательности для заданного подмножества путей без использования технологий сканирования, т.е. без дополнительных аппаратурных затрат в рамках ограничения на длину последовательности для отдельного пути. Проведенные эксперименты показывают, что тестовые последовательности удается построить не для всех путей (иногда ни для одного), для которых существуют тестовые пары в комбинационной составляющей схемы с памятью.

Ключевые слова: логическая схема с памятью, установочная последовательность, Reduced Ordered Binary Decision Diagram (ROBDD), робастно тестируемая неисправность задержки пути (PDF), rising (falling) transition.

DOI: 10.31857/S0005231021110106

1. Введение

Одним из основных параметров логических схем высокой производительности является высокая тактовая частота, т.е. высокая скорость функционирования схемы. Она определяется исходя из скорости прохождения сигналов через логические элементы вдоль путей, соединяющих входы и выходы схемы. Производители стремятся к увеличению тактовой частоты.

Для определения тактовой частоты схемы выделяют путь с максимальной задержкой — критический путь. Задержка этого пути определяет скорость функционирования схемы. Однако при высокой скорости функционирования и высоком уровне интеграции в схемах возникают непредусмотренные

разработчиками емкости, индуктивности, сопротивления, которые не удается рассчитать заранее. Это приводит к дополнительным задержкам в схеме и снижению ее расчетного быстродействия, что нежелательно. Модель неисправности задержки пути (Path Delay Fault (PDF) является одной из наиболее распространенных моделей задержек логической схемы, используемых на практике при тестировании логических схем. При использовании модели неисправности задержки пути предполагают, что задержки отдельных элементов пути и линий связи между ними малы, однако задержка пути в целом может превышать время между соседними синхросигналами тактовой частоты и искажать функционирование схемы. Выделяют робастно и не робастно тестируемые неисправности задержек путей. Под робастно тестируемой неисправностью понимают неисправность задержки пути, которая обнаруживается независимо от существования в схеме задержек других путей, превышающих допустимые значения. В противном случае неисправность называется не робастно тестируемой. При проявлении робастно тестируемой неисправности удается точно определить путь, на котором задержка имеет место, т.е. превышает время между соседними синхросигналами тактовой частоты. Проявление не робастно тестируемой неисправности такой возможности не дает. Информация о полученных задержках позволяет доработать схему с целью сохранения ее расчетного быстродействия или замаскировать действие обнаруженной задержки [1] с той же целью. Для тестирования задержек путей используются пары векторов (v_1, v_2). Векторы каждой пары могут отличаться друг от друга по переменной, отмечающей начало рассматриваемого пути, а возможно, и значениями других переменных. Договоримся в дальнейшем векторы (булевы) и наборы тестовой пары использовать как синонимы. Задержки противоположных перепадов значений сигналов (rising transition и falling transition) могут отличаться, поэтому необходимо для каждого пути в общем случае иметь две тестовые пары. Под rising transition будем, как это принято в зарубежных источниках, понимать последовательность смены значений сигналов вдоль пути, заканчивающуюся на его выходе изменением нулевого сигнала на единичный. Соответственно, под falling transition — последовательность смены значений сигналов вдоль пути, заканчивающуюся на его выходе изменением единичного сигнала на нулевой сигнал. Пусть в момент времени t с приходом синхросигнала на схему поступает вектор v_1 тестовой пары. При поступлении следующего синхросигнала на схему поступает вектор v_2. Если с приходом третьего синхросигнала на выходе схемы наблюдается сигнал, значение которого отличается от ожидаемого, то считаем, что в схеме имеет место задержка в ее путях.

На практике при тестировании задержек путей используют различные методы сканирования. При реализации этих методов в рамках Launch-on-Shift (LOS) технологии одним из векторов пары является тестовый набор для константной неисправности внутреннего полюса схемы, а второй вектор получается сдвигом первого. При использовании Launch-on-Capture (LOC) технологии второй вектор получается на основе реакции комбинационной составляющей схемы на первый вектор. Понятно, что при таких способах формирования пар векторов далеко не всегда удается получать тестовые пары для робаст-

но тестируемых неисправностей задержек путей. Обычно при использовании этих технологий удается обнаруживать около 20% таких неисправностей, в то время как использование точных методов [2], гарантирующих построение тестовой пары для робастно тестируемой неисправности, если она существует, позволяет обнаруживать от 40 до 90% и более таких неисправностей для заданного множества путей комбинационной составляющей схемы с памятью. Точные методы ориентированы на снижение потребляемой мощности. Проблема снижения потребляемой мощности при тестировании задержек исследуется во многих публикациях [3–7]. Другой проблемой при тестировании задержек являются большие аппаратурные затраты, связанные с доставкой тестов в рамках технологий сканирования. Как известно [8, 9], в технологиях сканирования используют в линиях обратных связей специальные триггеры, более сложные, чем D-триггеры. В рабочем режиме функционирования схемы они выполняют, как D-триггеры, задержку на один такт сигнала, сопоставляемого переменной состояния (внутренней переменной), а в режиме тестирования схемы образуют сдвиговый регистр, в который такт за тактом вносится составляющая тестового вектора, соответствующая внутренним переменным. Дополнительная аппаратура требуется также для разделения режимов функционирования и тестирования схемы с памятью. Точные методы могут быть применены в рамках Random Access Scan (RAS) технологий, реализация которых требует, к сожалению, больших аппаратурных затрат, чем использование LOS или LOC технологий. Итак, перечисленные способы тестирования задержек путей связаны с существенными аппаратурными затратами, которых хотелось бы избежать. Исследования, выполненные в данной работе, позволяют выяснить, всегда ли это возможно.

Имеется множество тестовых пар соседних булевых векторов (всех или некоторых), обнаруживающих робастно тестируемую неисправность задержки пути в комбинационной составляющей схемы с памятью. Требуется построить входную последовательность, доставляющую одну (любую) тестовую пару заданного множества из начального состояния q_0 схемы с памятью в рамках ограничения на длину последовательности. Предлагаются алгоритмы построения таких последовательностей в условиях, когда 1) начало пути отмечено входной переменной и 2) когда начало пути отмечено переменной состояний (внутренней переменной) схемы. Если искомые последовательности удается найти для каждого из путей заданного множества, то, совместив их, обнаружим задержки путей без дополнительных аппаратурных затрат.

Во втором разделе обсуждаются свойства пар наборов для робастно тестируемых неисправностей задержек путей. В третьем разделе описывается процедура получения Reduced Ordered Binary Decision Diagram (ROBDD-графа), представляющего множество всех тестовых пар соседних наборов для робастно тестируемых неисправностей задержек путей. В четвертом разделе приводятся алгоритмы построения последовательности, доставляющей из начального состояния тестовую пару для робастно тестируемой неисправности задержки пути в схеме с памятью. В пятом разделе обсуждаются результаты экспериментов.

2. Некоторые свойства тестовых пар наборов для робастно тестируемых неисправностей задержек путей

Имеем одновыходную схему C и соответствующую эквивалентную нормальную форму (ЭНФ). В [10] рассматривается задача построения пар тестовых наборов для робастно и не робастно тестируемых неисправностей задержек путей на основе анализа ЭНФ. Наборы v_2 тестовых пар являются тестовыми наборами для константных неисправностей литер ЭНФ.

В случае rising transition тестируется 0-константная неисправность литеры ЭНФ, соответствующей пути α (a_p-неисправность). В присутствии неисправности все вхождения литеры заменяются в ЭНФ константой 0. Тестовый набор, обнаруживающий эту неисправность, является набором v_2 тестовой пары, который порождает смену сигнала на выходе схемы с нулевого на единичное значение.

В случае falling transition тестируется 1-константная неисправность литеры ЭНФ, соответствующей пути α (b_p-неисправность). В присутствии неисправности все вхождения литеры заменяются в ЭНФ константой 1. Тестовый набор, обнаруживающий эту неисправность, является набором v_2 тестовой пары, который порождает смену сигнала на выходе схемы с единичного на нулевое значение.

В [10] показано, что для тестовых пар, обнаруживающих робастно тестируемые неисправности задержек путей, выполняются следующие условия:

1) если v_2 есть a_p-тестовый набор, то v_1 есть b_p-тестовый набор тестовой пары и наоборот;

2) на наборах тестовой пары функция, реализуемая схемой C, принимает противоположные значения;

3) минимально покрывающий интервал u векторов v_1, v_2 ортогонален всем конъюнкциям ЭНФ, не содержащим литеру, сопоставляемую пути α;

4) наборы (v_1, v_2) тестовой пары порождаются одной и той же непустой конъюнкцией.

При замене набора v_1 набором v_2 схема может попасть в промежуточное состояние, являющееся тестовым набором v_1 для другого пути схемы. В результате можно определить задержку другого пути, а не рассматриваемого. Такая ситуация исключается при выполнении условия 3. Для проверки условия 3 в [10] предлагается воспользоваться ЭНФ схемы. Однако ЭНФ, как правило, оказывается громоздкой даже для небольших схем. Для соседних наборов тестовой пары (v_1, v_2) при переходе от v_1 к v_2 промежуточных состояний не возникает. Использование этого факта избавляет от необходимости анализировать ЭНФ.

Теорема 1. Соседние наборы тестовой пары (v_1, v_2), в которой v_2 есть a_p-тестовый набор, а v_1 — b_p-тестовый набор или наоборот, обнаруживают робастно тестируемую неисправность задержки пути для обоих перепадов значений сигналов этого пути (для rising transition и falling transition).

Теорема 2. Если существует тестовая пара наборов (b_p, a_p), не являющихся соседними и обнаруживающих робастно тестируемую неисправность задержки пути α, то существует тестовая пара соседних наборов, обнаруживающая эту задержку.

Сам факт существования тестовой пары соседних наборов для робастно тестируемых неисправностей при существовании тестовой пары для несоседних наборов, возможно впервые, был отмечен как следствие теорем о свойствах ЭНФ при подстановке в нее одинаковых для соседних наборов констант [11]. От ЭНФ остается дизъюнкция из литер, сопоставляемых переменной, отмечающей начало пути. Конфигурация этих литер позволяет определить, какой тип задержки пути тестируется предъявленной парой соседних наборов. В теореме 2 подтверждается этот результат на основе исследования свойств конъюнкций ЭНФ (в них никакие переменные константами не заменяются), обеспечивающих существование тестовой пары для робастно тестируемых неисправностей задержек путей [10].

Из сказанного следует, что если рассматриваемый путь является робастно тестируемым, то для него существует тестовая пара, состоящая из соседних наборов.

Это также означает, что для установления существования тестовой пары для робастно тестируемой неисправности задержки пути достаточно ограничиться поиском тестовых пар, состоящих из соседних наборов. Предложенный в [12] алгоритм построения всех тестовых пар соседних наборов для обнаружения робастно тестируемых неисправностей задержек путей гарантирует нахождение в схеме всех робастно тестируемых путей.

Для построения множества всех тестовых пар соседних векторов для робастно тестируемых неисправностей рассматриваемого пути в [12] предлагается предварительно вычислять булеву разность (булеву производную) этого пути. Метод вычисления булевой разности и представления ее в виде ROBDD-графа $R(D_{path})$ основан на выполнении операций над ROBDD-графами, извлекаемыми из фрагментов комбинационной схемы, характеризующихся полиномиальной сложностью.

3. Получение тестовых пар соседних наборов для робастно тестируемых неисправностей из ROBDD $R(D_{path})$

Всевозможные тестовые наборы v_2 для обоих перепадов значений сигналов представлены в виде ROBDD $R(D_{path})$ [12]. Напомним, что в случае rising transition v_2 является тестовым набором для a_p-неисправности литеры, отмечающей начало пути α. Для falling transition v_2 является тестовым набором для b_p-неисправности литеры, отмечающей начало пути α. Приведем процедуру извлечения всех тестовых пар [12] для рассматриваемого пути. Пусть начало пути α отмечено переменной x_i без инверсии.

$R_{rise} = R(D_{path}) \,\&\, x_i(R(D_{path}) \,\&\, \overline{x}_i)$ — ROBDD-граф, представляющий все наборы v_2 для rising transition.

$R_{fall} = R(D_{path}) \,\&\, \overline{x}_i(R(D_{path}) \,\&\, x_i)$ — ROBDD-граф, представляющий все наборы v_2 для falling transition.

Для каждого типа перепадов значений сигналов пути α приведены две формулы, поскольку литера ЭНФ, сопоставляемая рассматриваемому пути, может иметь различные знаки инверсии.

Обозначим через R'_{rise} ROBDD-граф, полученный из R_{rise} удалением переменной x_i, а через R'_{fall} ROBDD-граф, полученный из R_{fall} удалением переменной x_i. В обоих случаях знаки переменной x_i не имеют значения.

Обозначим символом R_{rob} ROBDD-граф, представляющий тестовые пары соседних по переменной x_i наборов для робастно тестируемых неисправностей задержек пути α.

$R_{rob} = R'_{rise} \,\&\, R'_{fall}$.

Из процедуры построения R_{rob} следует, что граф не содержит переменную x_i, отмечающую начало пути. Путь от корня графа R_{rob} до его 1-концевой вершины представляет конъюнкцию такую, что булев вектор в пространстве переменных $x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n$ задает тестовую пару в пространстве n переменных для рассматриваемого пути. Один из векторов пары получается приписыванием переменной x_i значения 0, а другой — значения 1.

Если на векторе v_2 и выходе одновыходной схемы (для многовыходной схемы на выходе, сопоставляемом пути α) достигается значение 1, то v_2 — вектор для rising transition, если значение 0, то v_2 — вектор для falling transition.

В последовательностной схеме пары соседних булевых векторов, заданных ROBDD-графом R_{rob}, сопоставляются полным состояниям, т.е. в векторах пары выделяются входные составляющие и составляющие по переменным состояния.

Из тестовой пары формируются тройки векторов, обнаруживающие задержки инверсных перепадов сигналов рассматриваемого пути, либо v_2–v_1–v_2, либо v_1–v_2–v_1. Это значит, что при использовании тестовых пар из соседних булевых векторов имеется возможность обнаруживать противоположные перепады значений сигналов рассматриваемого пути в комбинационной составляющей тремя векторами вместо четырех.

Однако если переменные состояния в комбинационной составляющей не доступны, предлагается из начального состояния q_0 последовательностной схемы доставлять по отдельности пары для каждой из последовательностей перепадов рассматриваемого пути. Дело здесь в том, что необходимо не только попасть в состояние, сопоставляемое началу последовательности из трех векторов, но далее оказаться в специальной цепочке переходов, порождаемой тройкой тестовых векторов, чем длиннее цепочка, тем ниже вероятность попадания в нее.

4. Алгоритмы построения последовательности, обнаруживающей робастно тестируемую неисправность задержки пути в схеме с памятью

Приведем алгоритмы построения последовательности векторов входных переменных последовательностной схемы, доставляющей тестовую пару на

входы комбинационной составляющей схемы из начального состояния q_0. Напомним, что речь идет о соседних булевых векторах тестовой пары по переменной, отмечающей начало исследуемого пути. Эти алгоритмы формируют вместе с соответствующими тестовыми парами последовательности, обнаруживающие робастно тестируемые неисправности задержек путей, по одной входной последовательности для каждой последовательности перепадов значений сигналов рассматриваемого пути из заданного множества путей.

Пусть последовательностная схема получена из State Transition Graph (STG) описания, в котором выполнено кодирование состояний. STG-описание поведения дискретного устройства с памятью используется в зарубежных публикациях с прошлого века. Оно применяется в условиях, когда символы входного и выходного алфавита конечного автомата (абстрактной модели поведения устройства с памятью) уже закодированы разработчиком устройства. В STG-описании условие перехода из одного состояния в другое (с одновременным формированием выходного символа) представлено в виде троичного вектора в пространстве входных переменных и переменных состояний схемы с памятью. Это позволяет в общем случае компактно задавать множество условий этого перехода по сравнению с таблицами переходов-выходов, используемых в теории конечных автоматов. В табл. 1 приведен пример такого описания. Здесь множество состояний представлено символами $\{1, 2, 3, 4\}$.

Сопоставив состояниям булевы векторы в пространстве переменных z_1, z_2 следующим образом: 1 — $\overset{z_1z_2}{0\,0}$, 2 — $\overset{z_1z_2}{0\,1}$, 3 — $\overset{z_1z_2}{1\,0}$, 4 — $\overset{z_1z_2}{1\,1}$, получим таблицу, представляющую систему частичных булевых функций на множестве входных переменных и переменных состояний (табл. 2). В таблице функции переходов отмечены переменными z_1', z_2', а функции выходов переменными y_1, y_2, и т.д.

Эта таблица есть задание на синтез устройства. Для кодирования состояний использованы коды минимальной длины. При получении задания на синтез часто используют другие коды, например равновесные коды, коды Бергера и их модификации [13, 14] и др.

Необходимо принять во внимание следующие факты:

1) если в диаграмме переходов, соответствующей STG-описанию, отсутствуют петли, то невозможно построить последовательность, доставляющую тестовую пару для путей, начало которых отмечено входной переменной схемы;

2) если в результате кодирования состояний в полученном множестве кодов (в рабочей области, задаваемой STG-описанием) отсутствуют соседние векторы, то невозможно построить последовательность, доставляющую тестовую пару для путей, начала которых отмечены переменной состояния (внутренней переменной) в последовательностной схеме.

Итак, множество всех пар соседних булевых векторов, задающих полные состояния схемы и обнаруживающих робастно тестируемые неисправности задержек рассматриваемого пути в комбинационной части последовательностной схемы, представлено ROBDD-графом R_{rob}. В ROBDD R_{rob} снача-

Таблица 1. STG-описание поведения схемы с памятью

x_1	x_2	x_3	q	q'	y_1	y_2	y_3	y_4	y_5
0	–	–	1	1	0	0	0	1	0
–	0	–	1	1	0	0	0	1	0
1	1	–	1	2	1	0	0	1	0
–	–	0	2	2	0	0	1	1	0
–	–	1	2	3	1	0	1	1	0
1	0	–	3	3	0	1	0	0	0
0	–	–	3	4	1	1	0	0	0
–	1	–	3	4	1	1	0	0	0
–	–	0	4	4	0	1	0	0	1
–	–	1	4	1	1	1	0	0	1

Таблица 2. Описание поведения схемы с памятью после кодирования состояний

x_1	x_2	x_3	z_1	z_2	z_1'	z_2'	y_1	y_2	y_3	y_4	y_5
0	–	–	0	0	0	0	0	0	0	1	0
–	0	–	0	0	0	0	0	0	0	1	0
1	1	–	0	0	0	1	1	0	0	1	0
–	–	0	0	1	0	1	0	0	1	1	0
–	–	1	0	1	1	0	1	0	1	1	0
1	0	–	1	0	1	0	0	1	0	0	0
0	–	–	1	0	1	1	1	1	0	0	0
–	1	–	1	0	1	1	1	1	0	0	0
–	–	0	1	1	1	1	0	1	0	0	1
–	–	1	1	1	0	0	1	1	0	0	1

ла выполняется разложение Шеннона по внутренним, а затем по входным переменным. Это следует из метода его построения [12].

Имея в виду, что тестовые пары (v_1, v_2), задаваемые R_{rob}, состоят из входной и внутренней составляющих, представим их как: $v_1 = v_1^{in}, v_1^s$; $v_2 = v_2^{in}, v_2^s$, т.е. тестовую пару далее будем задавать следующим образом: $(v_1^{in}, v_1^s; v_2^{in}, v_2^s)$.

Рассматриваются следующие ситуации:

а) соседние булевы векторы, извлекаемые из R_{rob}, отличаются по входной переменной;

б) соседние булевы векторы, извлекаемые из R_{rob}, отличаются по внутренней переменной.

Случай а).

Рассмотрим тестовую пару $(v_1^{in}, v_1^s; v_2^{in}, v_1^s)$ векторов, которую необходимо подать на комбинационную составляющую последовательностной схемы для обнаружения задержки заданного пути α в условиях отличия векторов тестовой пары (составляющих v_1^{in}, v_2^{in}) по входной переменной x_i и при совпадении составляющих по переменным состояния. Для простоты положим, что пере-

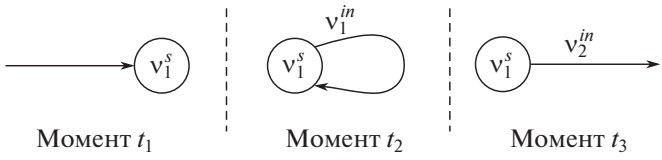

Рис. 1.

менная, отмечающая заданный путь, не имеет инверсии. Опишем процедуру доставки тестовой пары в соответствующие моменты времени:

1) предварительно с помощью подходящей входной последовательности, построение которой будет описано далее, устанавливаем схему (в момент t_1) в состояние v_1^s. В данном случае не важен ни входной сигнал, ни состояние, из которого выполняется переход в состояние v_1^s;

2) из состояния v_1^s под действием входного сигнала v_1^{in}, поступающего в момент t_2 (именно в этот момент обеспечивается подача на входы комбинационной составляющей последовательностной схемы первого вектора v_1^{in}, v_1^s тестовой пары) переходим в состояние v_1^s, т.е. реализуем петлю в автоматной диаграмме переходов с целью достижения состояния v_1^s в момент t_3;

3) в состоянии v_1^s в момент t_3 подаем на входы схемы вектор v_2^{in}, т.е. обеспечиваем поступление второго вектора v_2^{in}, v_1^s тестовой пары на входы комбинационной составляющей. Неважно, каким будет следующее состояние схемы, важно только, что должно произойти изменение сигнала на выходе, сопоставляемом рассматриваемому пути.

В момент t_4 наблюдаем неисправность задержки на соответствующем пути выходе комбинационной составляющей, если неисправность задержки имеет место.

Заметим, что соседние векторы тестовой пары поступают в следующие друг за другом моменты времени. Переходы в моменты времени t_1, t_2, t_3 иллюстрирует рис. 1.

Итак, для доставки тестовой пары необходимо из начального состояния q_0 попасть в состояние v_1^s. Это можно сделать, построив последовательность, устанавливающую схему из начального состояния в некоторое состояние из множества состояний типа v_1^s. Причем представляют интерес не все состояния, имеющие петлю, а только те, в которых петля реализуется по входной составляющей v_1^{in} , содержащейся среди полных состояний, представленных R_{rob}. Обеспечив поступление в этом состоянии входной составляющей, далее формируем вектор v_2^{in}, v_1^s, заменив в v_1^{in} значение переменной x_i инверсным.

Для построения установочной последовательности из начального состояния q_0 в состояние из заданного множества воспользуемся алгоритмом, предложенным в [15]. В нем множество состояний представлено ROBDD-графом R^{s_0}. Построение установочной последовательности из начального состояния является общей частью для различных типов перепадов сигналов

и различных типов переменных, отмечающих начало пути. Для каждой из этих ситуаций приходится строить свой граф R^{s0}.

Итак, строим множество R^{s0} для тестовой пары, в которой соседние векторы отличаются по входной переменной x_i. Из STG-описания выделяем фрагмент, представляющий петли в таблице переходов соответствующего конечного автомата. Фрагмент состоит из строк вида:

$$
\begin{gathered}
k_1 q_i \to q_i \\
k_2 q_i \to q_i \\
\dots \\
\dots \\
k_m q_i \to q_i \\
k_{m+1} q_j \to q_j \\
\dots \\
\dots \\
k_{m+s} q_j \to q_j \\
\dots
\end{gathered}
$$

Здесь k_i — конъюнкции (троичные векторы) на множестве входных переменных схемы, $q_i, q_j, \dots$ — состояния STG-описания (состояния конечного автомата), представленные булевыми векторами при кодировании. Левую часть рассматриваемого фрагмента, т.е. конъюнкции от входных переменных вместе с приписанными им состояниями представляем в виде ROBDD R_p. Этот граф пригодится при рассмотрении всех путей из предъявленного множества, начала которых отмечены входными переменными последовательностной схемы. Перейдем к рассмотрению заданного пути α.

Последовательность шагов для falling transition пути α:

1) из R_{rob} формируем a_p-тестовые наборы (наборы v_1^{in}, v_1^s) тестовой пары, сопоставляемые входной литере x_i, представляя их соответствующим ROBDD-графом: $R_{rob/x_i} = R_{rob} \,\&\, x_i$;

2) среди a_p-тестовых наборов выбираем такие, которые порождают петли: $R_{a/p} = R_{rob/x_i} \,\&\, R_p$, т.е. получаем множество всех векторов вида v_1^{in}, v_1^s, сопоставляемых моменту времени t_2 при тестировании задержки пути;

3) выделяем в $R_{a/p}$ множество внутренних состояний и представляем его графом R^{s0};

4) получив вектор v_1^{in}, v_1^s, далее формируем вектор v_2^{in}, v_1^s, который порождается графом R_{rob}, и подаем его в момент t_3. Вектор v_2^{in} отличается от вектора v_1^{in} по переменной x_i.

При нахождении тестовой пары для rising transition выполняем следующие шаги алгоритма для b_p-тестовых наборов.

Последовательность шагов для rising transition пути α:

1) из R_{rob} формируем b_p-тестовые наборы (наборы v_1^{in}, v_1^s) тестовой пары, сопоставляемые входной литере $\overline{x}_i$, представляя их соответствующим ROBDD-графом: $R_{rob/\overline{x}_i} = R_{rob} \,\&\, \overline{x}_i$;

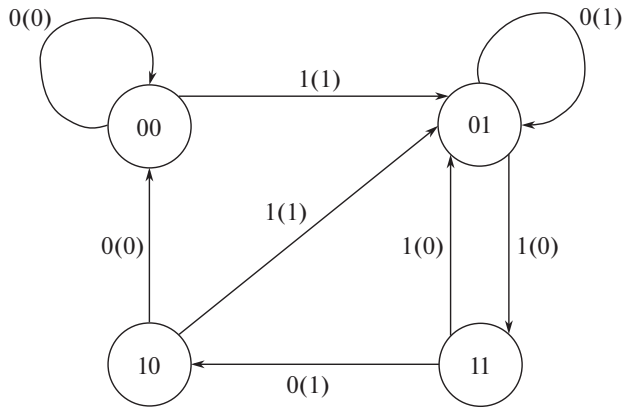

Рис. 2. Диаграмма переходов схемы с памятью.

2) среди b_p-тестовых наборов выбираем такие, которые порождают петли: $R_{b/p} = R_{rob/\overline{x}_i}\,\&\,R_p$, т.е. получаем множество всех векторов вида v_1^{in}, v_1^s, сопоставляемых моменту времени t_2, при тестировании задержки пути;

3) выделяем в $R_{b/p}$ множество внутренних состояний и представляем его графом R^{s_0};

4) получив вектор v_1^{in}, v_1^s, далее формируем вектор v_2^{in}, v_1^s, который порождается графом R_{rob}, и подаем его в момент t_3. Вектор v_2^{in} отличается от вектора v_1^{in} по переменной x_i.

Рассмотрим иллюстрирующий пример. Пусть поведение автомата представляется диаграммой переходов, представленной на рис. 2. В диаграмме переходов состояния закодированы следующим образом: $1 - \overset{z_1z_2}{0\,0}$, $2 - \overset{z_1z_2}{0\,1}$, $3 - \overset{z_1z_2}{1\,0}$, $4 - \overset{z_1z_2}{1\,1}$. Имеем частный случай STG-описания, когда условия переходов представлены булевыми векторами. Положим, начальное состояние q_0 представляется вектором $\overset{z_1z_2}{0\,0}$, сопоставляемым символу 1.

В диаграмме на дугах записаны входные и выходные (в скобках) векторы. В данном случае это однокомпонентные векторы. При кодировании состояний это описание превращается в задание системы булевых функций (см. табл. 3).

Таблица 3. Табличное задание системы булевых функций

x	z_1	z_2	z_1'	z_2'	y
0	0	0	0	0	0
0	0	1	0	1	1
1	0	0	0	1	1
0	1	0	0	0	0
1	1	0	0	1	1
0	1	1	1	0	1
1	1	1	0	1	0
1	0	1	1	1	0

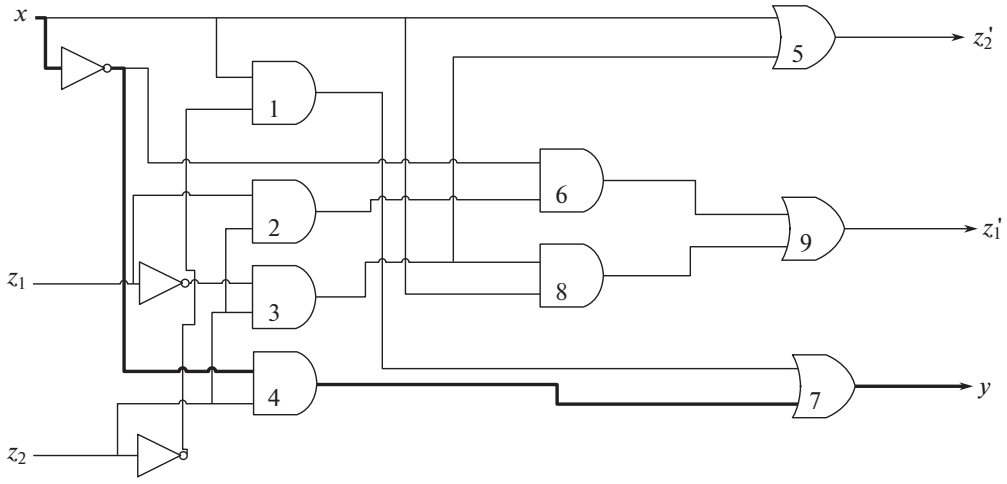

Рис. 3. Начало пути в схеме отмечено переменной x.

Из таблицы получаем реализацию системы булевых функций в виде системы ДНФ от входных переменных и переменных состояний:

$$z_1' = \overline{x}z_1z_2 \vee x\overline{z}_1z_2;$$
$$z_2' = x \vee \overline{z}_1z_2;$$
$$y = x\overline{z}_2 \vee \overline{x}z_2.$$

Эта система реализуется комбинационной составляющей схемы с памятью, показанной на рис. 3.

Рассмотрим путь $x \to 4 \to 7 \to y$.

Вычисляем для него булеву разность. В примере будем использовать операции над ДНФ, а не над ROBDD-графами для простоты. Чтобы сохранить соответствие тексту алгоритма, символ D (вместо ДНФ) будем сопровождать в скобках обозначением соответствующего ROBDD-графа из текста. Здесь символами $U_1, U_2, \ldots$ обозначены выходы соответствующих элементов схемы.

$$D_{U_7}/D_{U_4} = (U_1 \vee U_4)|_{U_4=0} \oplus (U_1 \vee U_4)|_{U_4=1} = \overline{U}_1;$$
$$D_{U_4}/D_x = (\overline{x}z_2)|_{x=0} \oplus (\overline{x}z_2)|_{x=1} = z_2;$$
$$D_{path} = (\overline{x\overline{z}_2})z_2 = (\overline{x} \vee z_2)z_2 = z_2.$$

Итак, $D_{path} = z_2$.

Вычисляем далее $D(R_{rob})$.

$$D(R_{rise}) = \overline{x}z_2;$$
$$D(R_{fall}) = xz_2;$$
$$D(R'_{rise}) = z_2;$$
$$D(R'_{fall}) = z_2;$$
$$D(R_{rob}) = z_2.$$

Из табл. 3 выделяем переходы, сопоставляемые петлям в диаграмме переходов. Условия, обеспечивающие эти переходы, задаем соответствующей ДНФ.

$$D(R_p) = \overline{xz}_1\overline{z}_2 \vee \overline{xz}_1 z_2.$$

Для falling transition выполняем пп. 1–3 алгоритма и получаем состояние $D(R^{s_0}) = \overset{z_1 z_2}{0\,1}$:

1. $D(R_{rob/\overline{x}}) = R_{rob}\overline{x} = \overline{x}z_2$;
2. $D(R_{a/p}) = \overline{xz}_1 z_2$;
3. $D(R^{s_0}) = \overline{z}_1 z_2$.

Далее строим последовательность, доставляющую схему из начального состояния $q_0(\overset{z_1 z_2}{0\,0})$ в состояние $\overset{z_1 z_2}{0\,1}$.

Пытаемся найти последовательность длины один, перемножая ДНФ, соответствующие переменным состояния: $D(R^{s_1}) = xz_1 \vee x\overline{z}_2 \vee \overline{xz}_1 z_2$.

Из полученной ДНФ следует, что из состояния $\overset{z_1 z_2}{0\,0}$ под действием входного символа $\overset{x}{1}$ попадаем в состояние $\overset{z_1 z_2}{0\,1}$. Из состояния $\overset{z_1 z_2}{0\,1}$ под действием входного символа $\overset{x}{0}$ оказываемся в том же состоянии $\overset{z_1 z_2}{0\,1}$ (рис. 2). Это значит, что, воспользовавшись последовательностью длины один, в момент времени t_1 попадаем в состояние, в котором можем организовать поступление нужной тестовой пары. А именно, под действием входного сигнала $\overset{x}{0}$ формируем первый вектор тестовой пары в момент t_2. Оказавшись в следующий момент t_3 в том же состоянии $\overset{z_1 z_2}{0\,1}$, подаем входной сигнал $\overset{x}{1}$ и формируем второй вектор тестовой пары. Это можно увидеть из диаграммы переходов (рис. 2).

Для rising transition выполняем пп. 1, 2 алгоритма.

1. $D(R_{rob/x}) = xz_2$;
2. $D(R_{b/p}) = 0$.

Поскольку $D(R_{b/p}) = 0$, приходим к заключению, что невозможно доставить тестовую пару для rising transition для рассматриваемого пути.

Случай б).

Рассмотрим тестовую пару $(v_1^{in}, v_1^s; v_1^{in}, v_2^s)$ векторов, которую необходимо подать на комбинационную составляющую последовательностной схемы для обнаружения задержки заданного пути α в условиях отличия векторов тестовой пары (составляющих v_1^s, v_2^s) по переменной состояния z_i и при совпадении составляющих по входным переменным. Для простоты положим, что переменная, отмечающая заданный путь, не имеет инверсии. Опишем процедуру доставки тестовой пары в соответствующие моменты времени:

1) предварительно с помощью подходящего входного воздействия устанавливаем схему (в момент t_1) в состояние v_1^s. В данном случае не важен ни входной сигнал, ни состояние, из которого выполняется переход в состояние v_1^s;

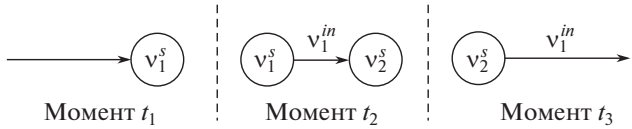

Рис. 4.

2) из состояния v_1^s под действием входного сигнала v_1^{in}, поступающего в момент t_2 (именно в этот момент обеспечивается подача на входы комбинационной составляющей последовательностной схемы первого вектора v_1^{in}, v_1^s тестовой пары), переходим в состояние v_2^s, т.е. переходим в соседнее состояние, отличающееся от v_1^s по переменной z_i;

3) в состоянии v_2^s в момент t_3 подаем на входы схемы вектор v_1^{in}, т.е. обеспечиваем поступление второго вектора v_1^{in}, v_2^s тестовой пары на входы комбинационной составляющей. Неважно, каким будет следующее состояние схемы, важно только, что должно произойти изменение сигнала на выходе, сопоставляемом рассматриваемому пути.

В момент t_4 наблюдаем неисправность задержки на соответствующем пути выходе комбинационной составляющей, если неисправность задержки имеет место.

Заметим, что соседние векторы тестовой пары поступают в следующие друг за другом моменты времени. Переходы в моменты времени t_1, t_2, t_3 иллюстрируются рис. 4.

Итак, для доставки тестовой пары во всех отмеченных выше ситуациях необходимо из начального состояния q_0 попасть в состояние v_1^s. Причем интерес представляют не все состояния, имеющие переходы в соседние состояния, а только те, в которых переход реализуется по входной составляющей v_1^{in}, содержащейся среди полных состояний, представленных R_{rob}. Обеспечив поступление в этом состоянии входной составляющей, далее формируем вектор v_1^{in}, v_2^s, заменив в v_1^s значение переменной z_i на инверсное.

Из STG-описания выделяем фрагмент, представляющий переходы в соседние состояния. Фрагмент состоит из строк вида:

$$
\begin{gathered}
k_1 q_{i_1} \to q_{j_1} \\
k_2 q_{i_1} \to q_{j_1} \\
\dots \\
\dots \\
k_m q_{i_1} \to q_{j_1} \\
k_{m+1} q_{i_2} \to q_{j_2} \\
\dots \\
\dots \\
k_{m+s} q_{i_2} \to q_{j_2} \\
\dots
\end{gathered}
$$

Здесь k_i – конъюнкции (троичные векторы) на множестве входных переменных схемы, $q_{i_1}, q_{j_1}, q_{i_2}, q_{j_2} \ldots$ – состояния STG-описания (состояния конечного автомата), представленные булевыми векторами при кодировании. Левую часть рассматриваемого фрагмента, т.е. конъюнкции от входных переменных вместе с приписанными им состояниями, представляем в виде ROBDD R_s. Этот граф пригодится при рассмотрении всех путей из предъявленного множества, начала которых отмечены переменными состояний последовательностной схемы. Перейдем к рассмотрению заданного пути α.

Последовательность шагов для falling transition пути α:

1) из R_{rob} формируем a_p-тестовые наборы (наборы v_1^{in}, v_1^s) тестовой пары, сопоставляемые переменной состояний z_i, представляя их соответствующим ROBDD-графом: $R_{rob/z_i} = R_{rob} \,\&\, z_i$;

2) среди a_p-тестовых наборов выбираем такие, которые порождают переходы в соседние состояния по переменной z_i: $R_{a/s} = R_{rob/z_i} \,\&\, R_s$, т.е. получаем множество всех векторов вида v_1^{in}, v_1^s, сопоставляемых моменту времени t_2 при тестировании задержки пути;

3) выделяем в $R_{a/s}$ множество внутренних состояний и представляем его графом R^{s_0};

4) получив вектор v_1^{in}, v_1^s, далее формируем вектор v_1^{in}, v_2^s, который порождается R_{rob}, и подаем его в момент t_3. Вектор v_2^s отличается от вектора v_1^s по переменной z_i.

При нахождении тестовой пары для rising transition выполняем следующие шаги алгоритма для b_p-тестовых наборов.

Последовательность шагов для rising transition пути α:

1) из R_{rob} выбираем b_p-тестовые наборы (наборы v_1^{in}, v_1^s) тестовой пары, сопоставляемые входной литере $\overline{z}_i$, представляя их соответствующим ROBDD-графом: $R_{rob/\overline{z}_i} = R_{rob} \,\&\, \overline{z}_i$;

2) среди b_p-тестовых наборов выбираем такие, которые порождают переходы в соседние состояния по переменной z_i: $R_{b/s} = R_{rob/\overline{z}_i} \,\&\, R_s$, т.е. получаем множество всех векторов вида v_1^{in}, v_1^s, сопоставляемых моменту времени t_2 при тестировании задержки пути;

3) выделяем в $R_{b/s}$ множество внутренних состояний и представляем его графом R^{s_0};

4) получив вектор v_1^{in}, v_1^s, далее формируем вектор v_1^{in}, v_2^s, который порождается R_{rob}, и подаем его в момент t_3. Вектор v_2^s отличается от вектора v_1^s по переменной z_i.

Итак, во всех выше перечисленных алгоритмах (п. 3) для отыскания тестовых пар формируется соответствующее множество R^{s_0}.

Находим последовательность [15], устанавливающую схему из начального состояния q_0 в одно из состояний соответствующего отыскиваемой паре множества R^{s_0}, т.е. в состояние v_1^s. Получив это состояние, используем для тестовой пары этого же типа ROBDD-граф из множества $\{R_{a/p}, R_{a/s}, R_{b/p}, R_{b/s}\}$ и

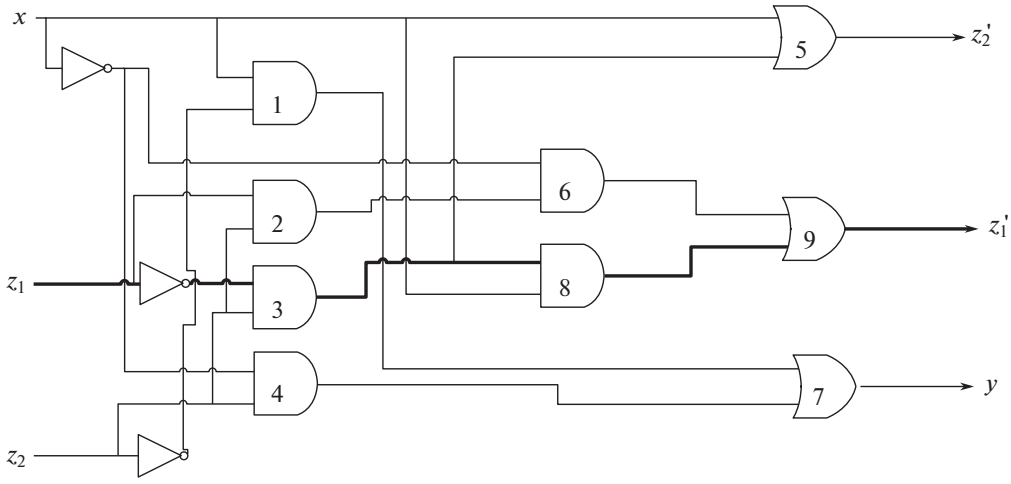

Рис. 5. Начало пути в схеме отмечено переменной z_1.

по нему находим первый вектор тестовой пары, а затем второй вектор способом, указанными в п.4 соответствующего отыскиваемой паре алгоритма.

Рассмотрим пример. В схеме задан путь $z_1 \to 3 \to 8 \to 9 \to z_1'$ (см. рис. 5).

Начало пути отмечено переменной z_1.

Вычисляем для пути булеву разность.

$$D_{U_9}/D_{U_8} = (U_8 \vee U_6)|_{U_8=0} \oplus (U_8 \vee U_6)|_{U_8=1} = U_6 \oplus 1 = \overline{U}_6;$$
$$D_{U_8}/D_{U_3} = (U_3 x)|_{U_3=0} \oplus (U_3 x)|_{U_3=1} = x;$$
$$D_{U_3}/D_{z_1} = (\overline{z}_1 z_2)|_{z_1=0} \oplus (\overline{z}_1 z_2)|_{z_1=1} = z_2;$$
$$D_{path} = (\overline{\overline{x} z_1 z_2}) x z_2 = (x \vee \overline{z}_1 \vee \overline{z}_2) x z_2 = x z_2.$$

Итак, булева разность представляется в виде $D_{path} = xz_2$.

Вычисляем далее $D(R_{rob})$.

$$D(R_{rise}) = x\overline{z}_1 z_2;$$
$$D(R_{fall}) = x z_1 z_2;$$
$$D(R'_{rise}) = x z_2;$$
$$D(R'_{fall}) = x z_2;$$
$$D(R_{rob}) = x z_2.$$

Получили единственную конъюнкцию, не содержащую переменную, отмечающую начало рассматриваемого пути. Конъюнкция порождает единственную тестовую пару. Получена $(\overset{xz_1z_2}{1\,1\,1}, \overset{xz_1z_2}{1\,0\,1})$.

Из табл. 3 выделяем переходы, представляемые соседними векторами на множестве переменных z_1, z_2. Условия, обеспечивающие эти переходы, задаем в виде ДНФ.

$$D(R_s) = x\overline{z}_1\overline{z}_2 \vee x\overline{z}_1 z_2 \vee x z_1 z_2 \vee \overline{x} z_1 z_2 \vee \overline{x} z_1 \overline{z}_2.$$

Рассматриваем falling transition. Выбираем вектор $\overset{xz_1z_2}{1\,0\,1}$. Это a_p-тестовый набор, поскольку он обращает в единицу ДНФ одновыходной подсхемы, которой принадлежит рассматриваемый путь.

Для falling transition выполняем пп. 1–3 алгоритма и получаем состояние $D(R^{s_0}) = \overset{z_1z_2}{0\,1}$.

1. $D(R_{rob/\overline{z}_1}) = x\overline{z}_1z_2$;
2. $D(R_{a/s}) = x\overline{z}_1z_2$;
3. $D(R^{s_0}) = \overline{z}_1z_2$.

Для обеспечения доставки тестовой пары необходимо из начального состояния q_0, представляемого вектором $\overset{z_1z_2}{0\,0}$, попасть в состояние $\overset{z_1z_2}{0\,1}$. Ранее было показано, что в состояние $\overset{z_1z_2}{0\,1}$ можно попасть из начального состояния $\overset{z_1z_2}{0\,0}$ по входному символу $\overset{x}{1}$ ($D(R^{s_1}) = xz_1 \vee x\overline{z}_2 \vee \overline{xz}_1z_2$.). В состоянии $\overset{z_1z_2}{0\,1}$ формируем тестовые пары $\overset{xz_1z_2}{1\,0\,1}$ и $\overset{xz_1z_2}{1\,1\,1}$. Эту ситуацию можно видеть из диаграммы переходов (рис. 2).

Рассматриваем rising transition. Выбираем вектор $\overset{xz_1z_2}{1\,1\,1}$. Это b_p-тестовый набор, поскольку он обращает в ноль ДНФ одновыходной подсхемы, которой принадлежит рассматриваемый путь.

Пытаемся найти последовательность длины один, перемножая ДНФ, соответствующие переменным состояния: $D(R^{s_1}) = x\overline{z}_1z_2$. Полученная ДНФ в начальном состоянии $\overset{z_1z_2}{0\,0}$ не обращается в единицу. Следовательно, не существует последовательности длины 1, доставляющей нужную тестовую пару. Попробуем найти последовательность длины два. Это значит, что надо найти условия перехода из начального состояния в состояние $\overset{z_1z_2}{0\,1}$. Ранее эти условия уже находили и представляли в виде ДНФ. Под действием входного символа $x = 1$ из начального состояния $\overset{z_1z_2}{0\,0}$ попадаем в состояние $\overset{z_1z_2}{0\,1}$. Далее из состояния $\overset{z_1z_2}{0\,1}$ под действием входного символа $\overset{x}{1}$ попадаем в состояние $\overset{z_1z_2}{1\,1}$. Итак, получили входную последовательность $\overset{x}{1}, \overset{x}{1}$, приводящую схему в состояние $\overset{z_1z_2}{1\,1}$. Это можно видеть по диаграмме переходов (рис. 2). Формируем первый вектор тестовой пары $\overset{xz_1z_2}{1\,1\,1}$. Под действием входного сигнала $x = 1$ переходим в состояние $\overset{xz_1z_2}{1\,0\,1}$, т.е. обеспечиваем поступление тестовой пары для rising transition. Эту ситуацию также можно видеть из диаграммы переходов (рис. 2).

Строим для каждого пути из заданного множества и типа перепадов значений сигналов соответствующие последовательности, обнаруживающие задержку пути, объединяем их и получаем тестовую последовательность для заданного множества путей. Отметим, что полученная тестовая последовательность состоит из фрагментов, начинающихся в состоянии q_0.

5. Результаты экспериментов

Описанные выше алгоритмы были реализованы в виде программы. Для операций над ROBDD-графами использовалась библиотека CUDD. Были проведены эксперименты на некоторых бенчмарках, результаты которых приведены в табл. 4. В схемах, представленных в бенчмарках, рассматривались только самые длинные пути. Длина установочной последовательности ограничивалась тысячью входными векторами, однако в процессе эксперимента установочные последовательности не достигали и сотни входных векторов. В последнем столбце таблицы показана доля (в процентах) путей, для которых из начального состояния последовательностной схемы удалось доставить тестовую пару, обнаруживающую робастно тестируемую неисправность задержки пути. Эта доля определяется по отношению ко всем путям, для которых существуют тестовые пары для тех же неисправностей в условиях доступности входов, сопоставляемых переменным состояний комбинационной составляющей.

Результаты экспериментов, проведенных на контрольных схемах, показали, что доля путей, для которых возможна доставка тестовых пар, обнаруживающих робастно тестируемые неисправности задержек путей, в лучшем случае составляет сорок с небольшим процентов. Из девяти схем для двух вообще не удалось доставить тестовые пары ни для одного из рассмотренных путей. Это значит, что отказываться от технологий сканирования, несмотря на связанные с ними большие аппаратурные затраты, нецелесообразно.

Обозначения:

— name – название бенчмарка;

— i – количество входов;

— s – количество переменных состояний;

— xr – количество путей, начало которых помечено входной переменной и для которых можно доставить тестовую пару из начального состояния схемы. Тестовая пара предназначена для rising transition;

— xf – количество путей, начало которых помечено входной переменной и для которых можно доставить тестовую пару из начального состояния схемы. Тестовая пара предназначена для falling transition;

Таблица 4. Результаты экспериментов

name	i	s	xf	xr	zf	zr	xob	zob	ob	N	%
s27	4	3	8	8	18	2	8	2	10	58	17,2%
s208	11	8	0	0	0	3	0	0	0	643	0%
s298	3	14	40	0	0	0	0	0	0	1179	0%
s386	7	6	45	57	320	572	0	160	160	1351	11,8%
s510	19	6	0	0	906	724	0	575	575	1373	41,9%
s820	18	5	34	451	1860	1392	0	1313	1313	3055	42,87%
s832	18	5	34	450	1859	1391	0	1312	1312	3052	42,98%
s1488	8	6	0	58	2578	2209	0	1612	1612	5384	29,94%
s1494	8	6	0	58	2628	2247	0	1626	1626	5351	30,39%

— zr – количество путей, начало которых помечено переменной состояния и для которых можно доставить тестовую пару из начального состояния схемы. Тестовая пара предназначена для rising transition;

— zf – количество путей, начало которых помечено переменной состояния и для которых можно доставить тестовую пару из начального состояния схемы. Тестовая пара предназначена для falling transition;

— xob – количество путей, начало которых помечено входной переменной и для которых можно доставить тестовые пары для обоих перепадов значений сигналов из начального состояния схемы;

— zob – количество путей, начало которых помечено переменной состояний и для которых можно доставить тестовые пары для обоих перепадов значений сигналов из начального состояния схемы;

— ob – общее количество робастно тестируемых путей, для каждого из которых обнаружимы оба перепада значений сигналов;

— N – количество путей, для которых существуют непустые графы R_{rob};

— % – доля (в процентах) робастно тестируемых путей, для которых удалось доставить тестовую пару, среди путей, для которых существуют непустые графы R_{rob}.

6. Заключение

Ранее был предложен метод отыскания всех тестовых пар, состоящих из соседних наборов булевых векторов, обнаруживающих робастно тестируемые неисправности задержек пути в условиях их доставки на входы комбинационной составляющей схемы с памятью. В данной работе предложен способ доставки таких тестовых пар из начального состояния q_0 схемы с памятью в условиях ограничения на длину последовательности, основанный на операциях над ROBDD-графами. Работа ориентирована на исследование возможности отказа от использования техник сканирования, требующих больших аппаратурных затрат. В работе выделены классы автоматов и способы кодирования состояний, для которых невозможна доставка тестовых пар, существующих в комбинационной составляющей. Проведенные эксперименты на контрольных примерах (benchmarks) показали, что далеко не всегда удается доставить хотя бы одну тестовую пару, не важно какую именно, для пути, имеющего тестовые пары для комбинационной составляющей в условиях доступности ее переменных состояния. Это значит, что для некоторых схем с памятью обнаружение робастно тестируемых неисправностей задержек путей без существенных аппаратурных затрат может оказаться невозможным.

ПРИЛОЖЕНИЕ

Доказательство теоремы 1. Будем иметь в виду, что a_p-тестовый набор тестовой пары для робастно тестируемой неисправности задержки пути обращает в единицу в общем случае несколько непустых конъюнкций ЭНФ (в непустых конъюнкциях ЭНФ отсутствуют взаимно инверсные переменные). Каждая из таких конъюнкций содержит литерал ЭНФ, сопоставляемый пути α (литерал ЭНФ — это символ переменной с подходящим знаком

инверсии и последовательностью индексов, представляющих путь α). Сопоставляемая литералу переменная присутствует в такой конъюнкции только один раз. Это объясняется тем, что b_p-тестовый набор тестовой пары (b_p, a_p) порождается конъюнкцией, которая не содержит литералов, отличающихся от литерала, сопоставляемого пути α только последовательностью индексов, сопоставляемых другому пути схемы. В то же время оба тестовых набора рассматриваемой пары порождаются одной и той же непустой конъюнкцией ЭНФ (условие 4). Отметим, что тестовый набор для a_p-неисправности (по построению) ортогонален каждой конъюнкции ЭНФ, не содержащей литерала, сопоставляемого пути α. Тестовый набор для b_p-неисправности (по построению) ортогонален всем конъюнкциям ЭНФ. Примем во внимание, что любой тестовый набор является булевым вектором и, следовательно, он ортогонален пустой конъюнкции. Это значит, что минимально покрывающий интервал u, порожденный соседними векторами (b_p, a_p), ортогонален каждой конъюнкции ЭНФ, не содержащей литерал, сопоставляемый пути α. Итак, для рассматриваемых соседних наборов выполняются условия быть тестовой парой для робастно тестируемой неисправности задержки пути, перечисленные ранее в работе. Имея в виду теорему 5 работы [7], приходим к заключению, что найденная тестовая пара может быть использована для обнаружения противоположных перепадов значений сигналов рассматриваемого пути. Теорема доказана.

Доказательство теоремы 2. Пусть u — минимально покрывающий интервал наборов (b_p, a_p) тестовой пары для робастно тестируемой неисправности задержки пути. Рассмотрим тестовый набор a_p. Этот набор (по построению) ортогонален каждой конъюнкции, не содержащей литерал, сопоставляемой пути α, и, возможно, пересекается с некоторыми остальными конъюнкциями рассматриваемой ЭНФ. Построим для a_p соседний по переменной x_i набор b'. Здесь x_i — переменная, отмечающая начало пути, для которого построена тестовая пара (b_p, a_p). Набор b' поглощается интервалом u, следовательно, тоже ортогонален каждой конъюнкции, не содержащей литерал, сопоставляемой пути α. Кроме того, b' ортогонален всем конъюнкциям, содержащим литерал, сопоставляемый пути α, по переменной x_i, т.е. b' ортогонален конъюнкциям ЭНФ в целом. Это значит, что он принадлежит области нулевых значений функции, представляемой ЭНФ, и является b_p-тестовым набором. Теорема доказана.

СПИСОК ЛИТЕРАТУРЫ

1. *Matrosova A., Ostanin S., Chernyshov S.* Masking Robust Testable PDFs // Proceedings of 2019 IEEE East-West Design & Test Symposium (EWDTS), 13–16 september 2019, Batumi. Kharkov: IEEE, 2019. P. 420–423.
2. *Matrosova A.Yu., Andreeva V.V., Tychinskiy V.Z., Goshin G.G.* Applying ROBDDs for delay testing of logical circuits. // Russ. Physics J. 2019. V. 62. No. 5. P. 615–621.
3. *Lindgren P., Kerttu M., Thornton M., Drechsler R.* Low power optimization technique for BDD mapped circuits // In Proceedings of the 2001 Asia and South Pacific Design Automation Conference (ASP-DAC '01), Association for Computing Machinery, New York, NY, USA, P. 615–621.

4. *Shelar R.S., Sapatnekar S.S.* An efficient algorithm for low power pass transistor logic synthesis // Proceedings of ASP-DAC/VLSI Design 2002, 7th Asia and South Pacific Design Automation Conference and 15h International Conference on VLSI Design, Bangalore, India, 2002, P. 87–92.
5. *Gekas G., Nikolos D., Kalligeros E., Kavousianos X.* "Power aware test-data compression for scan-based testing", ICECS 2005 // 12th IEEE International Conference on Electronics, Circuits and Systems, Gammarth, Tunisia, 2005, P. 1–4.
6. *Tudu J.T., Larsson E., Singh V., Agrawal V.D.* On Minimization of Peak Power for Scan Circuit during Test // Test Symposium 2009 14th IEEE European, 2009, P. 25–30.
7. *Kotasek Z., Skarvada J., Strnadel J.* Reduction of Power Dissipation Through Parallel Optimization of Test Vector and Scan Register Sequences // IEEE International Symposium on Design and Diagnostics of Electronic Circuits and Systems, 2010, P. 364–369.
8. *Eichelberger E.B., Williams T.W.* A Logic Design Structure for LSI // Proc. 14th Design Automation Conference. 1977. P. 462–468.
9. *Agrawal V.D., Cheng K.T., Johnson D.D.* Designing Circuits with Partial Scan // IEEE Design and Test of Computers. 1988. P. 8–15.
10. *Матросова А.Ю., Липский В.Б.* Свойства пар тестовых наборов, обнаруживающих неисправности задержек путей в логических схемах VLSI высокой производительности // АиТ. 2015. № 4. С. 135–148.

    Matrosova A.Yu., Lipskii V.B. Properties of pairs of test vectors detecting path delay faults in high performance VLSI logical circuits // Autom. Remote Control. 2015. No. 4. P. 135–148.
11. *Сапожников В.В., Сапожников Вл.В., Лыков А.А.* Теоремы анализа для обнаружения неисправности типа "временная задержка" // Электрон. моделирование. 2004. Т. 26б. No. 3. С. 83–93.
12. *Matrosova A.Yu., Andreeva V.V., Nikolaeva E.A.* Finding Test Pairs for PDFs in Logic Circuits Based on Using Operations on ROBDDs // Russ. Physics J. 2018. V. 61. No. 5. P. 994–999.
13. *Efanov D., Sapozhnikov V., Sapozhnikov Vl., Nikitin D.* Sum Code Formation with Minimum Total Number of Undetectable Errors in Data Vectors // Proceedings of 13th IEEE East-West Design & Test Symposium (EWDTS'2015), Batumi, Georgia, September 26–29, 2015, P. 141–148.
14. *Efanov D.V., Sapozhnikov V., Sapozhnikov Vl.* Two-Modulus Codes with Summation of One-Data Bits for Technical Di-agnostics of Discrete Systems // Automatic Control and Computer Sciences. 2018. V. 52. Issue 1. P. 5–21.
15. *Matrosova A., Andreeva V., Melnikov A.* ROBDDs Application for Finding the Shortest Transfer Sequence of Sequential Circuit or Only Revealing Existence of this Sequence without Deriving the Sequence itself //Proceedings of IEEE East-West Design & Test Symposium (EWDTS'2016). Yerevan: IEEE Computer Society, 2016. P. 513–516.

Статья представлена к публикации членом редколлегии М.Ф. Караваем.

Поступила в редакцию 22.05.2020

После доработки 01.02.2021

Принята к публикации 16.03.2021

СОДЕРЖАНИЕ

CONTENTS

Optimization, System Analysis, and Operations Research