ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ НИЖЕГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМ. Н. И. ЛОБАЧЕВСКОГО»

На правах рукописи

Емелин Максим Денисович

Комбинирование решений аксиальной задачи о назначениях

Научная специальность
1.2.2. Математическое моделирование, численные методы и комплексы программ

Диссертация на соискание ученой степени кандидата физико-математических наук

Научный руководитель
доктор физико-математических наук, доцент,
Афраймович Лев Григорьевич

Нижний Новгород - 2026

Оглавление

Введение

Актуальность темы исследования и степень её разработанности

Большое количество прикладных задач формализуется в виде многоиндексных аксиальных задач о назначениях:

- оптимизация логистических процессов и планирования – при управлении цепочками поставок, распределении ресурсов и планировании производства [1–3];
- многоцелевое отслеживание объектов – при обработке координатной информации, поступающей от множества разнородных сенсоров [4–7];
- многопараметрическое связывание записей – при разрешении сущностей и связывании данных из нескольких независимых крупномасштабных наборов данных [8];
- организация спутниковой связи – при маршрутизации каналов и распределении ресурсов в группировках высокопроизводительных спутников [9];
- компьютерное зрение – при межкадровом сопоставлении дескрипторов, отслеживании множества объектов и реконструкции трехмерных сцен [10];
- управление человеческими ресурсами – при оптимальном распределении исполнителей по проектам и задачам с учетом их компетенций [11];
- теория расписаний – при формировании неконфликтных графиков учебных занятий и распределении аудиторий в образовательных учреждениях [12,13];
- роботизированная логистика – при бесконфликтной диспетчеризации роботов на автоматизированных складах [14];
- биомедицина – для перепрофилирования лекарственных средств [15];
- молекулярный спектральный анализ – при назначении метильных групп для минимизации затрат на результирующее резонансное назначение [16,17].

Обзор результатов анализа подклассов многоиндексных задач о назначениях приведен в [1]. Известно, что класс многоиндексных аксиальных задач о назначениях является NP-трудным уже в трехиндексном случае [18]. В [19]

доказано отсутствие полиномиальных ε-приближенных алгоритмов решения трехиндексной аксиальной задачи о назначениях (здесь ε – произвольная константа), в противном случае P = NP. В [19] так же показано, что данный результат справедлив в случае, когда матрица стоимостей задачи о назначениях имеет декомпозиционную структуру (т.е. матрица стоимостей представима в виде суммы трех двухиндексных матриц). При этом задача о назначениях в случае, когда декомпозиционная матрица стоимостей удовлетворяет неравенству треугольника, остается NP-трудной [19].

Известны приближенные и эвристические алгоритмы решения NP-трудной аксиальной задачи о назначениях [2,15,20–25]. Такие алгоритмы, как правило, строят серию допустимых решений задачи. На финальном шаге таких алгоритмов общепринятым подходом является выбор рекорда среди построенных допустимых решений. В диссертационной работе в качестве улучшения финального шага выбора рекорда предлагается решение задачи оптимального комбинирования найденных допустимых решений.

В работах [19,26–29] исследуются аксиальные задачи о назначениях со специальной структурой многоиндексной матрицы стоимостей, в [30–32] обсуждаются вопросы построения потоковых алгоритмов решения многоиндексных задач, в [25] описывается метод ветвей и границ решения аксиальной задачи о назначениях, в [20] строится генетический алгоритм, в [33] обсуждается нижняя оценка задачи, в [34] исследуются асимптотически оптимальные решения. В планарной постановке задача о назначениях исследуется в работах [35–37]. Стохастические постановки многоиндексных задач исследуются в [38–40]. В многокритериальной постановке многоиндексные задачи о назначениях исследуются в [41–50].

В работе [19] предложен алгоритм, применимый при решении задач с декомпозиционной матрицей системы ограничений. Т.е. с матрицей стоимостей представимой в виде суммы трех двухиндексных матриц, т.е. $c_{ijk} = u^1_{ij} + u^2_{ik} + u^3_{jk}$, $i \in I, j \in J, k \in K$. Данный алгоритм основан на решении пары зависимых

двухиндексных задач. Всего существует 3 таких пары, таким образом, предложенный в [19] алгоритм позволяет построить 3 различных допустимых решения. В [19] показано, что рекорд из данных 3 решений является ε-приближенным решением задачи в случае, когда декомпозиционная матрица стоимостей удовлетворяет неравенству треугольника: $u_{ij}^1 \le u_{ik}^2 + u_{jk}^3, u_{ik}^2 \le u_{ij}^1 + u_{jk}^3, u_{jk}^3 \le u_{ij}^1 + u_{ik}^2,\ i \in I, j \in J, k \in K$. В [2] предложен эвристический подход, основанный на аппроксимации матрицы стоимостей исходной задачи матрицей стоимостей заданной структуры и решении задачи с аппроксимационной матрицей стоимостей. Для ряда аппроксимаций в [2] строятся оценки отклонения от оптимума.

В работе [15] предложен адаптированный эвристический подход из [44] с последовательным решением двухиндексных задач о назначениях, рассматриваются различные стратегии для порядка выбора решаемых задач. Такой же подход рассматривается для многоиндексных задач в работе [45]. В работе [20] построен гибридный генетический алгоритм решения задачи о назначениях, интерес представляет предложенная стратегия локальной оптимизации, которая может быть применена к любым допустимым решениям (в том числе полученных случайной генерацией). В работе [46] представлен алгоритм с разбиением задачи на подклассы, на каждом подклассе решается трехиндексная задача о назначениях, специфика алгоритма предполагает возможность его использования в качестве стратегии локальной оптимизации. Ещё одним вариантом поиска локального оптимума является алгоритм из [47], в [48] предлагается вариант улучшения этого алгоритма. В [49] рассматривается подход с решением двойственной задачи.

Значительный вклад в становление и развитие теории многоиндексных задач о назначениях внесли фундаментальные труды таких зарубежных и отечественных исследователей, как Гимади Э.Х., Емеличев В.А., Коркишко Н.М., Кравцов М.К., Медведева О.А., Медведев С.Н., Перепелица В.А., Прилуцкий М.Х., Тизик А.П., Цурков В.И., Balas E., Bandelt H.J., Burkard R.E., Crama Y., Frieze A.M., Gabrovšek B., Garey M.R., Gutin G., Hahn P.M., Huang G., John M., Johnson D.S., Kammerdiner

A.R., Karapetyan D., Kim B.J., Lim A., Magos D., Nagi R., Otting G., Pasiliao E.L., Pattipati K.R., Poore A.B., Rudolf R., Saltzman M.J., Semenov A., Vadrevu S., Spieksma F.C.R., Veenman C.J., Woeginger G., Yadegar J., Zhang L., Zhang Y. и др.

Цель и задачи работы

Целью диссертационной работы является построение математических моделей комбинирования решений трехиндексной аксиальной задачи о назначениях, улучшение точности решений, полученных различными эвристическими и приближенными алгоритмами. Для достижения целей поставлены и решены следующие задачи:

- разработка полиномиальных алгоритмов решения поставленных задач комбинирования;

- определение классов NP-трудных задач комбинирования;

- разработка программного комплекса решения трехиндексных аксиальных задач о назначениях с использованием разработанных алгоритмов комбинирования.

Научная новизна

Построены математические модели комбинирования допустимых решений трехиндексной аксиальной задачи о назначениях. Разработаны полиномиальные алгоритмы комбинирования, на основе которых предложен подход для постобработки допустимых решений трехиндексной аксиальной задачи о назначениях, демонстрирующий повышение точности решений, полученных различными эвристическими и приближенными алгоритмами. Выделены классы NP-трудных задач комбинирования.

Теоретическая и практическая ценность

Определены классы NP-трудных задач комбинирования. Найдены классы задач, для которых существуют полиномиальные алгоритмы решения этих задач.

Разработанные алгоритмы позволяют улучшить качество исходных решений аксиальных задач о назначениях при применении их к практическим задачам. Разработанный комплекс программ можно использовать для решения задач о назначениях. Предложенный подход применен при разработке программного модуля для моделирования выполнения производственного плана при различных графиках работы производства, входящий в состав АИС «Ока-план». Экономический эффект от внедрения результатов диссертации заключается в снижении затрат на проведение планирования работ по изготовлению изделий микроэлектроники на величину до 5%. Акт внедрения представлен в приложении 3.

Методология и методы исследования

В работе используются методы теории графов, дискретной оптимизации, а также теория вычислительной сложности.

Положения, выносимые на защиту

Предложен подход для постобработки допустимых решений трехиндексной аксиальной задачи о назначениях, основанный на комбинировании решений, в рамках которого построены математические модели комбинирования решений трехиндексной аксиальной задачи о назначениях. Для реализации данного подхода разработаны следующие алгоритмы:

- алгоритм решения задачи комбинирования допустимых решений трехиндексной аксиальной задачи о назначениях;
- алгоритм решения задачи комбинирования допустимых решений трехиндексной аксиальной задачи о назначениях с минимаксным критерием;
- эвристический алгоритм решения задачи комбинирования допустимых решений трехиндексной аксиальной задачи о назначениях с квадратичным критерием, показан подкласс задачи, в котором алгоритм находит оптимальное решение задачи;

- алгоритм решения задачи комбинирования допустимых решений многокритериальной трехиндексной аксиальной задачи о назначениях в случае лексикографической свертки критериев;

- алгоритм построения подмножества Парето-оптимальных решений задачи комбинирования допустимых решений многокритериальной трехиндексной аксиальной задачи о назначениях.

Доказана NP-трудность следующих задач:

- комбинирования четырех допустимых решений трехиндексной аксиальной задачи о назначениях;

- комбинирования двух допустимых решений трехиндексной аксиальной задачи о назначениях с минимаксным критерием;

- комбинирования четырех допустимых решений многокритериальной трехиндексной аксиальной задачи о назначениях в случае лексикографической и линейной сверток критериев.

Разработан программный комплекс, реализующий разработанные алгоритмы комбинирования, обеспечивающий автоматизацию решения задач о назначениях и подтвердивший эффективность разработанных методов в ходе вычислительных экспериментов.

Личный вклад автора

Все основные результаты получены автором самостоятельно. Научному руководителю (соавтору) принадлежит постановка задач. Интерпретация результатов, представленных в диссертационной работе, выполнялась совместно с научным руководителем.

Раздел 4.1 диссертационной работы выполнен при финансовой поддержке Министерства науки и высшего образования РФ в рамках Государственного задания по технологическому запросу (соглашение FSWR-2026-0019).

Соответствие паспорту специальности

Соответствие паспорту специальности 1.2.2. Математическое моделирование, численные методы и комплексы программ: в диссертационной работе представлены новые математические модели комбинирования допустимых решений трехиндексной аксиальной задачи о назначениях, что соответствует п. 1. «Разработка новых математических методов моделирования объектов и явлений (физико-математические науки)», разработаны алгоритмы для решения новых задач комбинирования, что соответствует п. 2. «Разработка, обоснование и тестирование эффективных вычислительных методов с применением современных компьютерных технологий», разработан комплекс программ, что соответствует п. 3. «Реализация эффективных численных методов и алгоритмов в виде комплексов проблемно-ориентированных программ для проведения вычислительного эксперимента».

Достоверность и апробация результатов

Достоверность результатов обусловлена строгостью математических доказательств. Полученные в диссертационный работе результаты обсуждались на XIII Международном семинаре «Дискретная математика и её приложения» имени академика О. Б. Лупанова (Москва, 2019), Международной школе-конференции «Высокопроизводительные вычисления и искусственный интеллект» (Нижний Новгород, 2019), 13-й Международной конференции «Интеллектуализация обработки информации» (Москва, 2020), Российском форуме «МОЛОДЁЖЬ и НАУКА» (Нижний Новгород, 2021), Форуме молодых ученых государств – участников СНГ «НАУКА БЕЗ ГРАНИЦ» (Нижний Новгород, 2022), XIV Международном семинаре «Дискретная математика и её приложения» имени академика О. Б. Лупанова (Москва, 2022). Основные результаты диссертационной работы представлены в 5 статьях, опубликованных в ведущих рецензируемых научных журналах (Автоматика и телемеханика, Mathematics) из списка,

рекомендованного ВАК [54,55,56,59,60]. Разработанный комплекс программ зарегистрирован в Федеральной службе по интеллектуальной собственности (РОСПАТЕНТ) [61].

Структура и содержание работы

Работа состоит из введения, четырех глав, заключения, списка литературы и приложения.

Во введении представлена актуальность класса многоиндексных аксиальных задач о назначениях, приведен обзор основных результатов по теме аксиальных задач о назначениях, сформулированы цели, показана научная новизна работы, продемонстрирована теоретическая и практическая ценность работы, описаны методы исследования, представлены положения, выносимые на защиту, и описана достоверность и апробация результатов.

В главе 1 приводятся математические модели комбинирования решений однокритериальной трехиндексной аксиальной задачи о назначениях.

В главе 2 приводятся математические модели комбинирования решений многокритериальной аксиальной задачи о назначениях с различными вариантами сверток критериев, а также математическая модель задачи нахождения Парето-оптимальных решений при комбинировании решений двухкритериальной трехиндексной аксиальной задачи о назначениях.

В главе 3 описываются алгоритмы решения поставленных задач комбинирования и найденные классы NP-трудных задач комбинирования.

В главе 4 описывается разработанный комплекс программ, реализующий разработанные алгоритмы комбинирования, с помощью которого был проведен вычислительный эксперимент, результаты которого приводятся в этой главе.

В заключении подведены итоги проведенных исследований.

1. Математические модели комбинирования в однокритериальных задачах

1.1. Прикладные задачи о назначениях

Задачи о назначениях имеют широкую область применений, в частности, аксиальные задачи о назначениях лежат в основе математического моделирования следующих прикладных задач:

Задача назначения исполнителей на работы

Содержательная постановка

Имеются работы, исполнители и орудия труда. Известна стоимость оплаты и время выполнения работы исполнителя на каждой работе с выбранным орудием труда. Каждый исполнитель должен быть назначен на одну работу и иметь в своем распоряжении одно орудие труда, каждое орудие труда должно быть назначено одному исполнителю и на каждую работу должен быть назначен один исполнитель. Требуется найти эффективное назначение исполнителей на работы с орудиями труда. [51]

Исходные параметры

I – множество исполнителей, J – множество орудий труда, K – множество работ.

Варьируемые параметры

x_{ijk} показывает назначен ли исполнитель i на работу k и выдано ли ему орудие труда $j, i \in I, j \in J, k \in K$.

Ограничения математической модели

$$\sum_{i \in I} \sum_{j \in J} x_{ijk} = 1, k \in K,$$

(на каждую работу должен быть назначен исполнитель с орудием труда)

$$\sum_{i \in I} \sum_{k \in K} x_{ijk} = 1, j \in J,$$

(каждое орудие труда должно быть выдано исполнителю для выполнения работы)

$$\sum_{j\in J}\sum_{k\in K} x_{ijk} = 1, i\in I,$$

(каждый исполнитель должен быть назначен на работу и ему должно быть выдано орудие труда)

$$x_{ijk} \in \{0,1\}, i \in I, j \in J, k \in K,$$

(условие неделимости исполнителей, работ и орудий труда)

Критерий оптимальности

Критерии оптимальности, определяющие эффективность назначения, могут зависеть от различных показателей. К примеру, c_{ijk} - стоимость оплаты работы исполнителя i на работе k с орудием труда j, d_{ijk} – время выполнения работы исполнителем i на работе k с орудием труда $j, i \in I, j \in J, k \in K$.

Тогда задача минимизации стоимости работ заключается в поиске решения, на котором оптимальное значение принимает критерий

$$\sum_{i\in I}\sum_{j\in J}\sum_{k\in K} c_{ijk}x_{ijk} \to min.$$

Задача минимизации времени выполнения работ заключается в поиске решения, на котором оптимальное значение принимает критерий

$$\max_{i\in I, j\in J, k\in K} d_{ijk}x_{ijk} \to min.$$

Задача назначения метильных групп

Содержательная постановка

Даны химические сдвиги метильных групп ^{1}H и ^{13}C измеренные в $^{13}C - HSQC$ спектре и химический сдвиг метильной группы ^{13}C, слишком близкий к парамагнетическому центру, чтобы их можно было наблюдать в спектре ядерного магнитного резонанса методом ^{1}H. [16]

Исходные параметры

I – множество диамагнетических резонансов, J – множество резонансов в присутствии лантаноида, K – множество остатков, возникающих из парамагнетизма лантаноида.

Варьируемые параметры

x_{ijk} показывает назначен ли диамагнетический резонанс i на резонанс в присутствии лантаноида j и остаток, возникающий из парамагнетизма лантаноида, $k, i \in I, j \in J, k \in K$.

Ограничения математической модели

$$\sum_{i \in I} \sum_{j \in J} x_{ijk} = 1, k \in K,$$

(на каждый остаток должна быть назначена пара резонансов)

$$\sum_{i \in I} \sum_{k \in K} x_{ijk} = 1, j \in J,$$

(на каждый резонанс в присутствии лантаноида должна быть назначена пара из диамагнетического резонанса и остатка, возникающего из парамагнетизма лантаноида)

$$\sum_{j \in J} \sum_{k \in K} x_{ijk} = 1, i \in I,$$

(на каждый диамагнетический резонанс должна быть назначена пара из резонанса в присутствии лантаноида и остатка, возникающего из парамагнетизма лантаноида)

$$x_{ijk} \in \{0,1\}, i \in I, j \in J, k \in K,$$

(условие неделимости резонансов и остатков)

Критерий оптимальности

Среди всех возможных резонансных назначений необходимо найти назначение с минимальной остаточной стоимостью.

$$\mathrm{C}(l) = \sum_{i \in I, j \in J, k \in K} x_{ijk}(c(\,^{1}H, i, j, k, l) + c(\,^{13}C, i, j, k, l)), \text{ где}$$

$$c(S,i,j,k,l) = \frac{\left[PCS^{S}_{calc}(k,l) - \left({}^{para}\delta^{S}_{exp}(j,l) - {}^{dia}\delta^{S}_{exp}(i)\right)\right]^2}{[e(l)]^2 + \left[{}^{para}\delta^{S}_{exp}(j,l) - {}^{dia}\delta^{S}_{exp}(i)\right]^2}, \text{где}$$

$PCS^{S}_{calc}(k,l) -$ предсказанное значение PCS (Pseudocontact shifts) для вращения S метильной группы в остатке k возникшего из парамагнетизма лантаноида l.

${}^{para}\delta^{S}_{exp}(j,l) -$ химический сдвиг резонанса j в присутствии лантаноида l для вращения S.

${}^{dia}\delta^{S}_{exp}(i) -$ диамагнетический резонанс i для вращения S.

$e(l) -$ постоянная величина, предотвращающая появление сингулярности в функции затрат.

Пусть $c_{ijk} = (c({}^{1}H,i,j,k,l) + c({}^{13}C,i,j,k,l))$, тогда критерий задачи принимает вид:

$$\sum_{i\in I}\sum_{j\in J}\sum_{k\in K} c_{ijk}x_{ijk} \to min.$$

Задача отслеживания целей по сенсорам

Содержательная постановка

Имеются сенсоры, считывающие данные о положении различных целей в дискретные моменты времени в определенной области. Задача заключается в сопоставлении положений целей в различные моменты времени реальным объектам. [4]

Исходные параметры

Пусть k – количество моментов времени, в которые были произведены собраны данные со сканеров, I_t – множество целей в момент времени $t, t = \overline{1,k}$.

Варьируемые параметры

$x_{i_1 \dots i_k}$ – переменная, обозначающая будут ли считаться цели $i_1 \dots i_k$ одним объектом, $i_1 \in I_1 \dots i_k \in I_k$.

Ограничения математической модели

$$\sum_{i_1\in I_1} \dots \sum_{i_{t-1}\in I_{t-1}} \sum_{i_{t+1}\in I_{t+1}} \dots \sum_{i_k\in I_k} x_{i_1 \dots i_k} = 1, i_t \in I_t,$$

(каждая цель должна быть назначена на какой-то объект)

$$x_{i_1 \dots i_k} \in \{0,1\}, i_1 \in I_1 \dots i_k \in I_k,$$

(целям $i_1 \dots i_k$ соответствует один объект либо не соответствует ни один)

Критерий оптимальности

Для задачи распознавания целей по расположениям в различные моменты времени существует функция вероятности $P(i_1 \dots i_k)$ того, что положения целей $i_1 \dots i_k$ соответствуют одному объекту, тогда $c_{i_1 \dots i_k}$ – можно определить как $c_{i_1 \dots i_k} = -\ln(P(i_1 \dots i_k))$, тогда критерий задачи может быть представлен в линейном виде

$$\sum_{i_1 \in I_1} \dots \sum_{i_k \in I_k} c_{i_1 \dots i_k} x_{i_1 \dots i_k} \to min$$

1.2. Аксиальные задачи о назначениях

Приведем постановку классической трехиндексной аксиальной задачи о назначениях.

Пусть I, J, K – непересекающиеся множества индексов, $I \cap J = \emptyset, I \cap K = \emptyset, J \cap K = \emptyset$ и $|I| = |J| = |K| = n$; c_{ijk}, $i \in I, j \in J, k \in K$ – трехиндексная матрица стоимостей; $x_{ijk}, i \in I, j \in J, k \in K$ – трехиндексная матрица неизвестных. Тогда трехиндексная аксиальная задача о назначениях ставится как следующая задача целочисленного линейного программирования:

$$\sum_{i \in I} \sum_{j \in J} x_{ijk} = 1, k \in K, \tag{1.1}$$

$$\sum_{i \in I} \sum_{k \in K} x_{ijk} = 1, j \in J, \tag{1.2}$$

$$\sum_{j \in J} \sum_{k \in K} x_{ijk} = 1, i \in I, \tag{1.3}$$

$$x_{ijk} \in \{0,1\}, i \in I, j \in J, k \in K, \tag{1.4}$$

$$\sum_{i\in I}\sum_{j\in J}\sum_{k\in K} c_{ijk}x_{ijk} \to min. \qquad (1.5)$$

Рассмотрим иные постановки трехиндексной задачи о назначениях.

Трехиндексная аксиальная задача о назначениях с минимаксным критерием ставится как следующая задача целочисленного линейного программирования (1.1)–(1.4), (1.6):

$$\max_{i\in I, j\in J, k\in K} \mathrm{c}_{ijk}x_{ijk} \to min. \qquad (1.6)$$

Обозначим задачу о назначениях с минимаксным критерием через Z^{minmax}.

Приведем постановку задачи о назначениях с квадратичным критерием.

Пусть I, J, K – непересекающиеся множества индексов, $I \cap J = \emptyset, I \cap K = \emptyset, J \cap K = \emptyset$ и $|I| = |J| = |K| = n$; $a_{ijk}, \mathrm{c}_{ijk},\ i\in I, j\in J, k\in K$ – трехиндексные матрицы стоимостей; $x_{ijk}, i\in I, j\in J, k\in K$ – трехиндексная матрица неизвестных. Тогда трехиндексная аксиальная задача о назначениях с квадратичным критерием ставится как следующая задача целочисленного линейного программирования (1.1)-(1.4), (1.7):

$$\sum_{i\in I}\sum_{j\in \mathrm{J}}\sum_{\mathrm{k}\in \mathrm{K}}\sum_{i\prime\in I\prime}\sum_{j\prime\in \mathrm{J}\prime}\sum_{\mathrm{k}\prime\in \mathrm{K}\prime} a_{ijk}a_{i\prime j\prime k\prime}x_{ijk}x_{i\prime j\prime k\prime} + \sum_{i\in I}\sum_{j\in J}\sum_{k\in K} c_{ijk}x_{ijk} \to \min \qquad (1.7)$$

Обозначим задачу о назначениях с квадратичным критерием через Z^2.

Приведенные задачи о назначениях являются NP-трудными [4,52]. При этом существует ряд эффективных алгоритмов (эвристических или ε-приближенных), показывающих хорошие практические результаты при решении численных задач. В [19] предложен алгоритм, применимый при решении задач с декомпозиционной матрицей системы ограничений, а именно с матрицей стоимостей представимой в виде суммы трех двухиндексных матриц: $c_{ijk} = u^1_{ij} + u^2_{ik} + u^3_{jk}, i \in I, j \in J, k \in K$. Данный алгоритм основан на решении пары зависимых двухиндексных задач. Всего существует три таких пары, таким образом, предложенный в [19] алгоритм позволяет построить три различных допустимых решения. В [19] показано, что рекорд из данных трех решений является ε-приближенным решением задачи в

случае, когда декомпозиционная матрица стоимостей удовлетворяет неравенству треугольника: $u_{ij}^1 \le u_{ik}^2 + u_{jk}^3, u_{ik}^2 \le u_{ij}^1 + u_{jk}^3, u_{jk}^3 \le u_{ij}^1 + u_{ik}^2, i \in I, j \in J, k \in K$. В [2] предложен эвристический подход, основанный на аппроксимации матрицы стоимостей исходной задачи матрицей стоимостей заданной структуры и решении задачи с аппроксимационной матрицей стоимостей. Для ряда аппроксимаций в [2] строятся оценки отклонения от оптимума. В [20] построен гибридный генетический алгоритм решения задачи о назначениях, интерес представляет предложенная стратегия локальной оптимизации, которая может быть применена к любым допустимым решениям (в том числе полученным случайной генерацией). В данной работе предлагается алгоритм, который может быть использован для постобработки начальных допустимых решений задачи (1.1) –(1.5), которые могут быть получены при помощи эвристик, описанных, например, в [2, 19, 20] и др.

1.3. Математическая модель комбинирования допустимых решений аксиальной задачи о назначениях

Опишем новую математическую модель комбинирования допустимых решений аксиальной задачи о назначениях, которая описывает процесс постобработки за счет комбинирования компонент ранее найденных решений. В рамках данной модели строится решение позволяющее повысить качество в соответствии с заданным критерием. Данный подход представляет альтернативу общепринятой стратегии выбора рекорда.

Пусть задано множество $W \subseteq I \times J \times K$, которое определяет подмножество разрешенных назначений. Тогда рассмотрим задачу (1.1)–(1.4), (1.8), (1.5)

$$x_{ijk} = 0, (i, j, k) \notin W. \qquad (1.8)$$

Задачу (1.1)–(1.4), (1.8), (1.5) для заданного множества W будем обозначать через $Z(W)$. Очевидно, задача (1.1)–(1.5) соответствует задаче $Z(I \times J \times K)$. Обозначим $C(x) = \sum_{i \in I} \sum_{j \in J} \sum_{k \in K} c_{ijk} x_{ijk}$.

В общем случае задача $Z(W)$ является NP-трудной [1]. Более того, проблема проверки совместности системы (1.1)–(1.4), (1.8) для произвольного множества W является NP-полной [1]. Далее будем рассматривать такие множества W, которые соотвествуют набору назначений некоторых допустимых решений задачи (1.1)–(1.5).

Введем вспомогательные обозначения. Пусть x – допустимое решение системы ограничений (1.1)–(1.4). Тогда через $W(x)$ обозначим следующее множество разрешенных назначений:

$W(x) = \{(i,j,k) | \, x_{ijk} = 1, i \in I, j \in J, k \in K\}$.

Пусть $x^1, x^2, \dots, x^m$ – произвольные допустимые решения системы ограничений (1.1)–(1.4). Обозначим $W(x^1, x^2, \dots, x^m) = \bigcup_{t=1}^{m} W(x^t)$. Тогда задача оптимального комбинирования m допустимых решений $x^1, x^2, \dots, x^m$ представляет собой задачу $Z\big(W(x^1, x^2, \dots, x^m)\big)$.

Для задачи с минимаксным критерием соответствующую задачу комбинирования (1.1)–(1.4), (1.8), (1.6) обозначим через $Z^{minmax}\big(W(x^1, \dots, x^r)\big)$ и задачу с квадратичным критерием (1.1)–(1.4), (1.8), (1.7) через $Z^2(W(x^1, \dots, x^r))$ соотвветственно.

Большое количество известных эвристических и приближенных алгоритмов решения аксиальной задачи о назначениях в ходе своей работы строят некоторое множество допустимых решений. Для удобства построенные допустимые решения обозначим $x^1, x^2, \dots, x^m$. Общепринятым подходом в подобных алгоритмах является выбор рекорда среди найденных допустимых решений на финальном шаге, т.е. : $C' = \min_{t=\overline{1,m}} C(x^t)$. В качестве улучшения финального шага выбора рекорда предлагается решение задачи $Z\big(W(x^1, x^2, \dots, x^m)\big)$. Т.е. выбор рекорда

среди допустимых решений предлагается заменить на поиск решения, скомбинированного из компонент найденных допустимых решений.

2. Математические модели комбинирования в многокритериальных задачах

2.1. Многокритериальная трехиндексная аксиальная задача о назначениях

Пусть I, J, K – непересекающиеся множества индексов, $I \cap J = \emptyset, I \cap K = \emptyset, J \cap K = \emptyset$ и $|I| = |J| = |K| = n$; M – фиксированное количество критериев задачи; $c_{ijk}^{u}, i \in I, j \in J, k \in K, u = \overline{1, M}$ – трехиндексные матрицы стоимостей; $x_{ijk}, i \in I, j \in J, k \in K$ – трехиндексная матрица неизвестных. Тогда многокритериальная трехиндексная аксиальная задача о назначениях ставится как следующая задача целочисленного линейного программирования:

$$\sum_{i \in I} \sum_{j \in J} x_{ijk} = 1, k \in K, \tag{2.1}$$

$$\sum_{i \in I} \sum_{k \in K} x_{ijk} = 1, j \in J, \tag{2.2}$$

$$\sum_{j \in J} \sum_{k \in K} x_{ijk} = 1, i \in I, \tag{2.3}$$

$$x_{ijk} \in \{0,1\}, i \in I, j \in J, k \in K, \tag{2.4}$$

$$\sum_{i \in I} \sum_{j \in J} \sum_{k \in K} c_{ijk}^{u} x_{ijk} \to min, u = \overline{1, M}. \tag{2.5}$$

Для удобства многокритериальную задачу (2.1)–(2.5) обозначим через Z_M.

2.2. Математическая модель комбинирования допустимых решений многокритериальной задачи о назначениях

Опишем новую математическую модель комбинирования допустимых решений аксиальной задачи о назначениях, которая описывает процесс постобработки за счет комбинирования компонент ранее найденных решений. В рамках данной модели строится решение позволяющее повысить качество в соответствии с заданным критерием. Данный подход представляет альтернативу общепринятой стратегии выбора рекорда.

Пусть множество $W \subseteq I \times J \times K$ определяет подмножество разрешенных назначений

$$x_{ijk} = 0, (i,j,k) \notin W. \tag{2.6}$$

Тогда определим через $Z_M(W)$ оптимизационную задачу с критерием (2.5) и системой ограничений (2.1)–(2.4), (2.6) для заданного множества W. Нетрудно заметить, что задача (2.1)–(2.5) соответствует задаче $Z_m(I \times J \times K)$. Проблема проверки совместности системы ограничений задачи $Z_M(W)$, т.е. системы (2.1)–(2.4), (2.6), для произвольного множества W является NP-полной [1]. Мы будем рассматривать такие множества W, которые соответствуют назначениям заданного подмножества допустимых решений.

Пусть $x_{ijk}, i \in I, j \in J, k \in K$ – допустимое решение системы ограничений (2.1)-(2.4). Тогда через $W(x)$ обозначим следующее множество разрешенных назначений:

$W(x) = \{(i,j,k) | x_{ijk} = 1, i \in I, j \in J, k \in K\}$.

Рассмотрим $x^1_{ijk}, x^1_{ijk}, \dots, x^r_{ijk},, i \in I, j \in J, k \in K$ – произвольные r допустимых решений системы ограничений (2.1)–(2.4). Тогда

$$W(x^1, x^2, \dots, x^r) = W(x^1) \cup W(x^2) \cup \dots \cup W(x^r).$$

Тогда задача оптимального комбинирования r допустимых решений $x^1, x^2, \dots, x^r$ представляет собой задачу $Z_M\big(W(x^1, x^2, \dots, x^r)\big)$.

Обозначим $C^u(x) = \sum_{i\in I}\sum_{j\in J}\sum_{k\in K} c^u_{ijk}x_{ijk}$.

Введем следующее обозначение: если $p \in \{1,2\}$, то $\bar{p} = 3 - p$.

2.3. Лексикографическая свертка критериев

Будем предполагать, что порядок значимости критериев (2.5) совпадает с их исходной индексацией. Тогда введем отношение предпочтения на множестве допустимых решений задачи о назначениях. Обозначим через P множество допустимых решений системы ограничений (2.1)–(2.4). Пусть $x^1, x^2 \in P$, тогда будем записывать $x^1 \preccurlyeq x^2$ тогда и только тогда, когда существует $y \in \{0, \dots, M\}$ такой, что выполняются условия (2.7), (2.8):

$$\sum_{i\in I}\sum_{j\in J}\sum_{k\in K} \mathrm{c}^u_{ijk}x^1_{ijk} = \sum_{i\in I}\sum_{j\in J}\sum_{k\in K} \mathrm{c}^u_{ijk}x^2_{ijk}, u \in \{1, \dots, y\}, \quad (2.7)$$

$$\sum_{i\in I}\sum_{j\in J}\sum_{k\in K} \mathrm{c}^u_{ijk}x^1_{ijk} < \sum_{i\in I}\sum_{j\in J}\sum_{k\in K} \mathrm{c}^u_{ijk}x^2_{ijk}, u \in \{1, \dots, M\} \cap \{y+1\}. \quad (2.8)$$

Тогда многокритериальная задача с лексикографической сверткой критериев заключается в поиске x^*, удовлетворяющего системе ограничений:

$$x^* \in P, \quad (2.9)$$

$$x^* \preccurlyeq x, x \in P. \quad (2.10)$$

Задачу (2.9), (2.10) обозначим $Z_{\preccurlyeq}$. Несложно увидеть, что к задаче $Z_{\preccurlyeq}$ полиномиально сводима NP-трудная трехиндексная аксиальная задача о назначениях. В соответствующей задаче $Z_{\preccurlyeq}$ определим $\mathrm{c}^1_{ijk} = \mathrm{c}_{ijk}, \mathrm{c}^u_{ijk} = 0$, $i\in I, j\in J, k\in K, u = \overline{2, M}$. Отсюда следует утверждение:

Утверждение 2.1. Задача $Z_{\preccurlyeq}$ NP-трудна.

В случае лексикографической свертки критериев соответствующая задача принимает вид (2.9), (2.10), где в качестве множества P применяется множество

допустимых решений системы ограничений (2.1)–(2.4), (2.6). Данную задачу обозначим $Z_{\leqslant}(W)$.

2.4. Минимаксная свертка критериев

Минимаксная свертка критериев будет иметь вид

$$\max_{u\in\{1,\dots,M\}}\left(\sum_{i\in I}\sum_{j\in J}\sum_{k\in K}c_{ijk}^{u}x_{ijk}\right)\to min. \tag{2.11}$$

Для удобства задачу (2.1)–(2.4), (2.11) обозначим через Z_{minmax}. Легко увидеть, что к задаче Z_{minmax} полиномиально сводима NP-трудная трехиндексная аксиальная задача о назначениях. В соответствующей задаче Z_{minmax} определим $c_{ijk}^{u}=c_{ijk}, i\in I, j\in J, k\in K, u=\overline{1,M}$. Отсюда следует утверждение:

Утверждение 2.2. Задача Z_{minmax} NP-трудна.

В случае минимаксной свертки критериев соответствующая задача принимает вид (2.1)–(2.4), (2.6), (2.11), для удобства обозначаемая $Z_{minmax}(W)$.

2.5. Линейная свертка критериев

Рассмотрим многокритериальную задачу о назначениях в случае линейной свертки критериев.

Пусть заданы веса $\alpha_u, u=\overline{1,M}$. Тогда линейная свертка критериев будет иметь вид

$$\sum_{u=1}^{M}\alpha_u\sum_{i\in I}\sum_{j\in J}\sum_{k\in K}c_{ijk}^{u}x_{ijk}\to min. \tag{2.12}$$

Для удобства задачу (2.1)–(2.4), (2.12) обозначим через Z_L. Легко увидеть, что к задаче Z_L полиномиально сводима NP-трудная (однокритериальная) трехиндексная аксиальная задача о назначениях. Действительно, рассмотрим трехиндексную аксиальную задачу о назначениях с матрицей стоимостей $c_{ijk}, i \in I, j \in J, k \in K$. В соответствующей задаче Z_L определим $c^1_{ijk} = c_{ijk}, c^u_{ijk} = 0$, $\alpha_1 = 1$, $\alpha_u = 0$, $i \in I, j \in J, k \in K$, $u = \overline{2, M}$. Отсюда следует утверждение:

Утверждение 2.3. Задача Z_L NP-трудна.

В случае линейной свертки критериев соответствующая задача принимает вид (2.1)–(2.4), (2.6), (2.12), для удобства обозначаемая $Z_L(W)$.

2.6. Математическая модель задачи нахождения Парето-оптимальных решений

Для удобства запишем двухкритериальную задачу в следующем виде:

Пусть I, J, K – непересекающиеся множества индексов, $I \cap J = \emptyset, I \cap K = \emptyset, J \cap K = \emptyset$ и $|I| = |J| = |K| = n$; $c_{ijk}, d_{ijk}, i \in I, j \in J, k \in K$ – трехиндексные матрицы стоимостей; $x_{ijk}, i \in I, j \in J, k \in K$ – трехиндексная матрица неизвестных. Тогда двухкритериальная трехиндексная аксиальная задача о назначениях ставится как следующая задача целочисленного линейного программирования (2.1)–(2.4), (2.13), (2.14):

$$\sum_{i \in I} \sum_{j \in J} \sum_{k \in K} c_{ijk} x_{ijk} \to min, \tag{2.13}$$

$$\sum_{i \in I} \sum_{j \in J} \sum_{k \in K} d_{ijk} x_{ijk} \to min.$$

Обозначим

$$C(x) = \sum_{i \in I} \sum_{j \in J} \sum_{k \in K} c_{ijk} x_{ijk},$$

$$D(x) = \sum_{i \in I} \sum_{j \in J} \sum_{k \in K} d_{ijk} x_{ijk}.$$

Тогда x^* называется Парето-оптимальным решением задачи Z_2, если x^* удовлетворяет системе ограничений (2.1)–(2.4); не существует x', удовлетворяющего системе ограничений (2.1)–(2.4), что при этом $C(x^*) > C(x')$ и $D(x^*) \geq D(x')$ или $C(x^*) \geq C(x')$ и $D(x^*) > D(x')$.

Как известно, (однокритериальная) аксиальная задача о назначениях (2.1)–(2.4), (2.13) является NP-трудной [4]. Отсюда несложно увидеть, что задача построения Парето-оптимального решения двухкритерильной аксиальной задачи о назначениях Z_2 также является NP-трудной. Для доказательства необходимо в соответствующей задаче Z_2 определить $c_{ijk} = d_{ijk}, i \in I, j \in J, k \in K$.

Утверждение 2.4. Задача построения Парето-оптимального решения задачи Z_2 является NP-трудной.

Таким образом актуальным является вопрос построения эффективных эвристических подходов при оценке Парето-области задачи Z_2. Предлагаемый в данной работе эвристический подход основан на идее комбинирования допустимых решений задачи Z_2. Под комбинированием здесь понимается решение задачи на множестве решений, которые содержат только компоненты заданных допустимых решений.

x^* называется Парето-оптимальным решением задачи $Z_2(W)$, если x^* удовлетворяет системе ограничений (2.1)–(2.4), (2.7); не существует x', удовлетворяющего системе ограничений (2.1)–(2.4), (2.7), что при этом $C(x^*) > C(x')$ и $D(x^*) \geq D(x')$ или $C(x^*) \geq C(x')$ и $D(x^*) > D(x')$.

3. Численные методы комбинирования решений

3.1. Комбинирование пары решений аксиальной задачи о назначениях

Для задачи комбинирования двух допустимых решений трехиндексной аксиальной задачи о назначениях был разработан полиномиальный алгоритм, находящий оптимальное решение поставленной задачи.

Алгоритм 3.1[54]. *Решение задачи* $Z\left(W(x^1, x^2)\right)$.

Шаг 1. Построить граф $G = (V, A)$, где $V = \{I \cup J \cup K\}$, $A = \{(i,j),(i,k),(j,k)|(i,j,k) \in W(x^1, x^2)\}$.

Шаг 2. Найти компоненты связности графа $G_l = (V_l, A_l), l = \overline{1,q}$.

Шаг 3. Построить следующие множества:

$$D_l^1 = \{(i,j,k)|(i,j,k) \in W(x^1),(i,j),(i,k),(j,k) \in A_l\},$$

$$D_l^2 = \{(i,j,k)|(i,j,k) \in W(x^2),(i,j),(i,k),(j,k) \in A_l\}.$$

Шаг 4. Оптимальное значение критерия задачи $Z\left(W(x^1, x^2)\right)$ определяется как

$$c^* = \sum_{l=1}^{q} \min\left(\sum_{(i,j,k)\in D_l^1} c_{ijk}, \sum_{(i,j,k)\in D_l^2} c_{ijk}\right).$$

Шаг 5. Оптимальное решение задачи определяется по следующему алгоритму. Пусть $x_{ijk} := 0, i \in I, j \in J, k \in K$. Далее для каждого $l = \overline{1,q}$ выполнить

$x_{ijk} := 1, (i,j,k) \in D_l^{p^*}$, где $p^* = \underset{p\in\{1,2\}}{\operatorname{argmin}} \sum_{(i,j,k)\in D_l^p} c_{ijk}$.

Пусть $V_l^1 = \bigcup_{(i,j,k)\in D_l^1}\{i,j,k\}, V_l^2 = \bigcup_{(i,j,k)\in D_l^2}\{i,j,k\}, l = \overline{1,q}$.

Лемма 3.1. $V_l^1 = V_l^2 = V_l,\ l = \overline{1,q}$.

Доказательство. Покажем, что $V_l^1 = V_l,\ l = \overline{1,q}$. Докажем методом от противного. Предположим, что найдется $l \in \{1,2,\dots,q\}$, что выполняется условие $V_l^1 \neq V_l$. По построению $V_l^1 \subseteq V_l$, тогда существует v, что $v \in V_l$, и $v \notin V_l^1$.

Тогда согласно условию (1.1)–(1.4) существует тройка (i,j,k), что $x^1_{ijk}=1$ и $v\in\{i,j,k\}$. Отсюда $(i,j,k)\in W(x^1)$ и, следовательно, $(i,j),(i,k),(j,k)\in A$. Тогда т.к. $v\in V_l$ и $v\in\{i,j,k\}$, то $(i,j),(i,k),(j,k)\in A_l$ и $(i,j,k)\in D^1_l$. Отсюда $i,j,k\in V^1_l$ и, следовательно, $v\in V^1_l$. Получаем противоречие.

Соотношение $V^2_l=V_l,\ l=\overline{1,q}$, доказывается аналогично. Лемма 3.1 доказана.

Лемма 3.2. $|V_l\cap I|=|V_l\cap J|=|V_l\cap K|,\ l=\overline{1,q}$.

Доказательство. Согласно условию (1.1)–(1.4) выполняется $\left|V^1_l\cap I\right|=\left|V^1_l\cap J\right|=\left|V^1_l\cap K\right|$. По лемме 3.1 $V^1_l=V_l$, отсюда $|V_l\cap I|=|V_l\cap J|=|V_l\cap K|$. Лемма 3.2 доказана.

Теорема 3.1. Оптимальное значение критерия задачи $Z\left(W(x^1,x^2)\right)$ определяется как $\sum_{l=1}^{q} min\left(\sum_{(i,j,k)\in D^1_l} c_{ijk}\,,\sum_{(i,j,k)\in D^2_l} c_{ijk}\right)$.

Доказательство. Рассмотрим компоненту связности $V_l,\ l\in\{1,2,\dots,q\}$. Согласно лемме 3.2 выполняется $|V_l\cap I|=|V_l\cap J|=|V_l\cap K|$. Тогда упорядочим множества $I\cap V_l$, $J\cap V_l$, $K\cap V_l$ следующим образом.

В качестве i_1,j_1,k_1 выберем произвольную тройку $(i_1,j_1,k_1)\in D^1_l$. Далее пусть мы упорядочили первые m элементов множеств:

$$i_1,i_2,\dots,i_m,$$
$$j_1,j_2,\dots,j_m,$$
$$k_1,k_2,\dots,k_m.$$

Тогда выберем тройку i_{m+1},j_{m+1},k_{m+1}, удовлетворяющую следующим правилам:

- $i_{m+1}\in(I\cap V_l)\backslash\{i_1,i_2,\dots,i_m\}$, $j_{m+1}\in(J\cap V_l)\backslash\{j_1,j_2,\dots,j_m\}$, $k_{m+1}\in(K\cap V_l)\backslash\{k_1,k_2,\dots,k_m\}$,
- $(i_{m+1},j_{m+1},k_{m+1})\in D^1_l$,
- существует тройка $(i_{m+1},j,k)\in D^2_l$, что $j\in\{j_1,j_2,\dots,j_m\}$ или $k\in\{k_1,k_2,\dots,k_m\}$.

Далее покажем, что такая тройка $(i_{m+1}, j_{m+1}, k_{m+1})$ всегда существует, в противном случае множества упорядочены.

Построим множества $V_m^1 = \bigcup_{q=1}^{m}\{i_q, j_q, k_q\}$, $V_m^2 = \bigcup_{q=\overline{1,m},(i_q,j,k)\in D_l^2}\{i_q, j, k\}$. Возможны следующие два случая.

1) $V_m^1 \neq V_m^2$, тогда существует вершина $j_q \notin V_m^2$ или $k_q \notin V_m^2, q = \overline{1,m}$, следовательно существует тройка $(i,j,k)\in D_l^2$, что $j_q = j$ или $k_q = k$, данную вершину i выберем в качестве i_{m+1}. В качестве j_{m+1}, k_{m+1} выберем такие индексы, что $(i_{m+1}, j_{m+1}, k_{m+1}) \in D_l^1$.

2) $V_m^1 = V_m^2$. По построению для каждой вершины $i \in \{i_1, i_2, \dots, i_m\}$ существует тройка $(i,j,k)\in D_l^2$, причем $j \in \{j_1, j_2, \dots, j_m\}$ и $k \in \{k_1, k_2, \dots, k_m\}$. Тогда согласно (1.1)–(1.4) не существует тройки $(i,j,k)\in D_l^2$, что $\{i,j,k\} \cap V_m^1 \neq \emptyset$ и $\{i,j,k\}\backslash V_m^1 \neq \emptyset$. Тогда V_m^1 образует компоненту связанности графа G и, следовательно, $V_m^1 = V_l$. Таким образом множества упорядочены.

Далее рассмотрим произвольное допустимое решение задачи $x_{ijk}, i \in I, j \in J, k \in K$, задачи $W(x^1, x^2)$. Пусть $R_l = \{(i,j,k)\big| x_{ijk} = 1, \{i,j,k\} \in V_l\}$, $l = \overline{1,q}$. Предположим, что для некоторого $l \in \{1,2,\dots,q\}$ найдутся $(i',j',k'), (i'',j'',k'') \in R_l$, что $(i',j',k') \neq (i'',j'',k'')$, $(i',j',k') \in D_l^1$, $(i'',j'',k'') \in D_l^2$. Пусть множества $I \cap V_l, J \cap V_l, K \cap V_l$ упорядочены согласно процедуре, описанной выше:

$$i_1, i_2, \dots, i_m,$$
$$j_1, j_2, \dots, j_m,$$
$$k_1, k_2, \dots, k_m.$$

Здесь $I \cap V_l = \{i_1, i_2, \dots, i_m\}$, $J \cap V_l = \{j_1, j_2, \dots, j_m\}$, $K \cap V_l = \{k_1, k_2, \dots, k_m\}$. Не уменьшая общности будем считать, что $(i_1, j_1, k_1) = (i', j', k')$.

Как показано выше $(i_1, j_1, k_1) \in R_l$. Пусть для некоторого $t < m$ выполняется $(i_1, j_1, k_1), (i_2, j_2, k_2), \dots, (i_t, j_t, k_t) \in R_l$. Тогда согласно описанному выше правилу упорядочивания множеств существует тройка $(i_{t+1}, j, k) \in D_l^2$, что $j \in \{j_1, j_2, \dots, j_t\}$ или $k \in \{k_1, k_2, \dots, k_t\}$. Тогда с учетом условия (1.1)–(1.4) выполняется

$(i_{t+1}, j_{t+1}, k_{t+1}) \in R_l$. Следовательно, $(i_1, j_1, k_1), (i_2, j_2, k_2), \dots, (i_m, j_m, k_m) \in R_l$. Получаем противоречие.

Таким образом для каждого $l \in \{1,2, \dots, q\}$ выполняется $R_l = D_l^1$ или $R_l = D_l^2$. Отсюда оптимальное значение критерия задачи $Z\big(W(x^1, x^2)\big)$ определяется как $\sum_{l=1}^{q} \min\left(\sum_{(i,j,k)\in D_l^1} c_{ijk}, \sum_{(i,j,k)\in D_l^2} c_{ijk}\right)$. Теорема 3.1 доказана.

Следствие 3.1. Любое допустимое решение x, удовлетворяющее системе ограничений задачи $Z\big(W(x^1, x^2)\big)$, может быть построено путем выбора троек только из первого или только из второго решения для каждой компоненты связности независимо.

Доказательство. Данное следствие было показано в ходе доказательства Теоремы 3.1.

Для допустимых решений x, удовлетворяющих системе ограничений задачи $Z\big(W(x^1, x^2)\big)$ введем:

$$p_l(x) = \begin{cases} 1, \text{если } x_{ijk} = 1, \text{для всех } (i,j,k) \in D_l^1, \\ 2, \text{если } x_{ijk} = 1, \text{для всех } (i,j,k) \in D_l^2. \end{cases}$$

Теорема 3.2. Алгоритм 3.1 требует $O(n)$ вычислительных операций.

Доказательство. Пусть входными данными алгоритма 3.1 являются допустимые решения x^1, x^2, представленные в виде коллекции троек (i,j,k) и стоимостей c_{ijk}, для которых соответствующие переменные принимают значение 1. Согласно (1.1)–(1.4) для каждого из допустимых решений количество таких троек составляет n. Тогда на шаге 1 алгоритма строится граф $G = (V, A)$, такой что $|V| = O(n)$, $|A| = O(n)$.

На шаге 2 граф G разбивается на компоненты связности. Это осуществляется при помощи обхода графа в ширину, который требует $O(|V| + |A|) = O(n)$ вычислительных операций.

На шаге 3 для входных троек алгоритма определяются соответствующие компоненты связанности, этот шаг требует $O(n)$ вычислительных операций.

И наконец на шаге 4 необходимо определить оптимальное значение критерия. Согласно шагу 4 по каждой из компонент связанности выбирается минимум среди подмножества троек, соответствующих входным допустимым решениям. Данный шаг требует $O(n)$ вычислительных операций.

Таким образом алгоритм 3.1 требует $O(n)$ вычислительных операций. Теорема 3.2 доказана.

3.2. Эвристические стратегии комбинирования произвольного числа решений

Покажем, что решение задачи комбинирования $m > 2$ допустимых решений $Z(W(x^1, x^2, \dots, x^m))$ в общем случае не может быть сведено к последовательному комбинированию пар допустимых решений. В качестве базового алгоритма последовательного комбинирования пар допустимых решений рассмотрим следующий алгоритм.

Алгоритм 3.2. Последовательное комбинирование пар допустимых решений.

Вход: Исходные допустимые решения $x^1, x^2, \dots, x^m$ и перестановка $p \in S_m$.

Шаг 1. y^1 является решением задачи $Z\left(W(x^{p_1}, x^{p_2})\right)$.

Шаг 2. y^t является решением задачи $Z\left(W(y^{t-1}, x^{p_{t+1}})\right), t = \overline{2, m-1}$.

Выход: y^{m-1}.

Замечание. На шагах 1 и 2 алгоритма 3.2 применяется алгоритм 3.1 оптимального комбинирования пар решений (случай $m = 2$).

Теорема 3.3. Алгоритм 3.2 не является алгоритмом решения задачи $Z\left(W(x^1, x^2, \dots, x^m)\right)$ при $m > 2$.

Доказательство. Для доказательства данной теоремы приведем контрпример. Пусть $n = 3,\ m = 3$,

$$c = \left[\begin{bmatrix} 1 & 2 & 2 \\ 2 & 2 & 2 \\ 2 & 2 & 2 \end{bmatrix} \begin{bmatrix} 2 & 2 & 2 \\ 2 & 1 & 2 \\ 2 & 2 & 2 \end{bmatrix} \begin{bmatrix} 2 & 2 & 2 \\ 2 & 2 & 2 \\ 2 & 2 & 1 \end{bmatrix}\right],$$

$$x^1 = \left[\begin{bmatrix}1 & 0 & 0\\0 & 0 & 0\\0 & 0 & 0\end{bmatrix}\ \begin{bmatrix}0 & 0 & 0\\0 & 0 & 0\\0 & 1 & 0\end{bmatrix}\ \begin{bmatrix}0 & 0 & 0\\0 & 0 & 1\\0 & 0 & 0\end{bmatrix}\right],$$

$$x^2 = \left[\begin{bmatrix}0 & 0 & 0\\0 & 0 & 0\\1 & 0 & 0\end{bmatrix}\ \begin{bmatrix}0 & 0 & 0\\0 & 1 & 0\\0 & 0 & 0\end{bmatrix}\ \begin{bmatrix}0 & 0 & 1\\0 & 0 & 0\\0 & 0 & 0\end{bmatrix}\right],$$

$$x^3 = \left[\begin{bmatrix}0 & 0 & 0\\1 & 0 & 0\\0 & 0 & 0\end{bmatrix}\ \begin{bmatrix}0 & 1 & 0\\0 & 0 & 0\\0 & 0 & 0\end{bmatrix}\ \begin{bmatrix}0 & 0 & 0\\0 & 0 & 0\\0 & 0 & 1\end{bmatrix}\right].$$

Несложно увидеть, что граф $G = (V, A)$, где $V = \{I \cup J \cup K\}$, $A = \{(i,j),(i,k),(j,k)\big|(i,j,k) \in W(x^{l_1}, x^{l_2})\}$, $l_1 \neq l_2, l_1, l_2 \in \{1,2,3\}$ имеет одну компоненту связности.

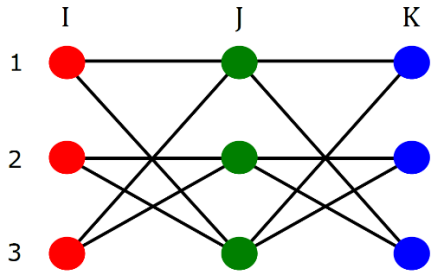

Рисунок 3.1. Граф, построенный для $l_1 = 1, l_2 = 2$

Отсюда согласно [54] решением задачи $Z\left(W(x^{l_1}, x^{l_2})\right)$ является x^{l_1} или x^{l_2}. Следовательно, при применении алгоритма 3.2 комбинирования решений x^1, x^2, x^3 с произвольной перестановкой $p \in S_3$ мы получим на выходе одно из этих решений. При этом оптимальным решением задачи $Z(W(x^1, x^2, x^3))$ является

$$x^* = \left[\begin{bmatrix}1 & 0 & 0\\0 & 0 & 0\\0 & 0 & 0\end{bmatrix}\ \begin{bmatrix}0 & 0 & 0\\0 & 1 & 0\\0 & 0 & 0\end{bmatrix}\ \begin{bmatrix}0 & 0 & 0\\0 & 0 & 0\\0 & 0 & 1\end{bmatrix}\right],$$

в котором тройка (1,1,1) выбрана из решения x^1, тройка (2,2,2) из x^2, тройка (3,3,3) из x^3. И выполняется условие $C(x^1) = C(x^2) = C(x^3) = 5 > 3 = C(x^*)$. Теорема 3.3 доказана.

По причине отсутствия на данный момент известного эффективного алгоритма решения задачи $Z(W(x^1, x^2, \dots, x^m))$ при $m > 2$ предлагается ряд эвристических стратегий решения данной задачи, основанных на последовательном комбинировании пар решений. Согласно теореме 3.3, данные стратегии не гарантируют построение оптимального решения, однако на

численных задачах предложенные стратегии показывают улучшение результатов по сравнению с общепринятым выбором рекорда.

Предлагаются следующие эвристические стратегии комбинирования решений:

Стратегия 1.

Применить алгоритм 3.2 к решениям $x^1, x^2, \dots, x^m$ со случайной перестановкой $p \in S_m$. Полученное алгоритмом 3.2 решение выбрать в качестве выходного.

Стратегия 2.

Применить алгоритм 3.2 к решениям $x^1, x^2, \dots, x^m$ с перестановкой $p \in S_m$, удовлетворяющей следующему свойству $C(x^{p_i}) \le C(x^{p_{i+1}}), i = \overline{1, m-1}$, т.е. решения упорядочены в порядке неубывания критерия. Полученное алгоритмом 3.2 решение выбрать в качестве выходного.

Стратегия 3.

Применить алгоритм 3.2 к решениям $x^1, x^2, \dots, x^m$ с перестановкой $p \in S_m$, удовлетворяющей следующему свойству $C(x^{p_i}) \le C(x^{p_{i+1}}), i = \overline{1, m-1}$. Обозначим полученное алгоритмом 3.2 решение через y_1.

Получить k решений $y_t,\ t = \overline{2, k+1}$ следующим способом. Для каждого $t = \overline{2, k+1}$ создать перестановку $p \in S_m$, удовлетворяющую свойству $C(x^{p_i}) \le C(x^{p_{i+1}}),\ i = \overline{1, m-1}$, выбрать случайным образом $d = \left\lfloor \frac{m}{2} \right\rfloor$ элементов перестановки и случайно поменять их местами. Применить Алгоритм 3.2 к решениям $x^1, x^2, \dots, x^m$ с перестановкой $p \in S_m$. Полученное решение обозначим через y_t.

Применить Алгоритм 3.2 к решениям $y_1, y_2, \dots, y_{k+1}$ с тождественной перестановкой $e = (1,2, \dots, k+1)$. Полученное алгоритмом 3.2 решение выбрать в качестве выходного.

3.3. Комбинирование пары решений задачи о назначениях с минимаксным критерием

Для задачи комбинирования двух допустимых решений трехиндексной аксиальной задачи о назначениях с минимаксным критерием был разработан алгоритм по аналогии с алгоритмом 3.1.

Алгоритм 3.3. Решение задачи $Z^{minmax}\left(W(x^1, x^2)\right)$.

Шаг 1. Построить граф $G = (V, A)$, где $V = \{I \cup J \cup K\}$, $A = \{(i,j),(i,k),(j,k)|(i,j,k) \in W(x^1, x^2)\}$.

Шаг 2. Найти компоненты связности $G_l = (V_l, A_l), l = \overline{1,q}$.

Шаг 3. Построить следующие множества:

$$D_l^1 = \{(i,j,k)|(i,j,k) \in W(x^1),(i,j),(i,k),(j,k) \in A_l\},$$

$$D_l^2 = \{(i,j,k)|(i,j,k) \in W(x^2),(i,j),(i,k),(j,k) \in A_l\}.$$

Шаг 4. Оптимальное значение критерия задачи $Z^{minmax}\left(W(x^1, x^2)\right)$ определяется как

$$c^* = \max_{l=\overline{1,q}} \min\left(\max_{(i,j,k)\in D_l^1} c_{ijk}, \max_{(i,j,k)\in D_l^2} c_{ijk} \right).$$

Шаг 5. Оптимальное решение задачи определяется по следующему алгоритму. Пусть $x_{ijk} := 0, i \in I, j \in J, k \in K$. Далее для каждого $l = \overline{1,q}$ выполнить

$x_{ijk} := 1, (i,j,k) \in D_l^{p^*}$, где $p^* = \underset{p\in\{1,2\}}{\operatorname{argmin}} \max\limits_{(i,j,k)\in D_l^p} c_{ijk}$.

Сформулируем следующие утверждения:

Утверждение 3.1. Алгоритм 3.3 находит решение задачи $Z^{minmax}\left(W(x^1, x^2)\right)$.

Утверждение 3.2. Алгоритм 3.3 требует $O(n)$ вычислительных операций.

Доказательства проводятся по аналогии с соответствующими теоремами для алгоритма 3.1.

3.4. Комбинирование пары решений задачи о назначениях с квадратичным критерием

Для задачи комбинирования двух допустимых решений трехиндексной задачи о назначениях с квадратичным критерием разработан эвристический алгоритм решения и найден подкласс задач, в котором алгоритм находит оптимальное решение.

Алгоритм 3.4. Решение задачи $Z^2\big(W(x^1,x^2)\big)$.

Шаг 1. Построить граф $G=(V,A)$, где $V=\{I\cup J\cup K\}$, $A=\{(i,j),(i,k),(j,k)|(i,j,k)\in W(x^1,x^2)\}$.

Шаг 2. Найти компоненты связности $G_l=(V_l,A_l), l=\overline{1,q}$.

Шаг 3. Построить следующие множества:

$$D_l^1=\{(i,j,k)|(i,j,k)\in W(x^1),(i,j),(i,k),(j,k)\in A_l\},$$

$$D_l^2=\{(i,j,k)|(i,j,k)\in W(x^2),(i,j),(i,k),(j,k)\in A_l\}.$$

Шаг 4. Пусть

$$A_l^p=\sum_{(i,j,k)\in D_l^p} a_{ijk}\,, l=\overline{1,q}, p=\overline{1,2},$$

$$C_l^p=\sum_{(i,j,k)\in D_l^p} c_{ijk}\,, l=\overline{1,q}, p=\overline{1,2},$$

$$P=\left\{p\middle|p=\underset{p\in\{1,2\}}{\operatorname{argmin}} A_l^p\right\},$$

$$p_l^*=\underset{p\in P}{\operatorname{argmin}} C_l^p\,, l=\overline{1,q}.$$

Шаг 5. Оптимальное решение задачи определяется по следующему алгоритму. Пусть $x_{ijk}:=0, i\in I, j\in J, k\in K$. Далее для каждого $l=\overline{1,q}$ выполнить

$$x_{ijk}:=1,(i,j,k)\in D_l^{p_l^*}.$$

Теорема 3.4. Алгоритм 3.4 находит решение задачи $Z^2\big(W(x^1,x^2)\big)$, если

$$A_l^p\geq 0,\ \mathrm{l}=\overline{1,q}, p\in\{1,2\},$$

$$A_l^1 \le A_l^2 \text{ и } C_l^1 \le C_l^2 \text{ или } A_l^1 \ge A_l^2 \text{ и } C_l^1 \ge C_l^2,\ l = \overline{1,q}$$

Доказательство (от противного). Предположим, что решение. Полученное алгоритмом 3.4 не является решением задачи $Z^2\big(W(x^1,x^2)\big)$. Тогда существует оптимальное решение x задачи $Z^2\big(W(x^1,x^2)\big)$ достигающее меньшего значения критерия.

Следствие 3.1 будет верно и для задачи $Z^2\big(W(x^1,x^2)\big)$, доказательство проводится аналогично. Тогда для любого допустимого решения x' задачи комбинирования критерий можно переписать в следующем виде:

$$\sum_{\mathrm{l}=\overline{1,q}} \sum_{l'=\overline{1,q}} A_l^{p_l(x')} A_{l'}^{p_{l'}(x')} + \sum_{l=\overline{1,q}} C_l^{p_l(x')} \to \min$$

В соответствии со следствием 3.1 найдется компонента связности $l \in \{1,\dots,q\}$ такая, что тройки в ней были выбраны не из того же решения, что и в решении x^*, и при этом согласно предположению $A_l^{p_l(x)} \le A_l^{p_l(x^*)}$ и $C_l^{p_l(x)} \le C_l^{p_l(x^*)}$. Тогда, так как $A_l^p \ge 0$, $\mathrm{l} = \overline{1,q}, p \in \{1,2\}$, то можно заменить тройки в решении x, уменьшив значение критерия, то есть x не является оптимальным решением задачи $Z^2\big(W(x^1,x^2)\big)$, что приводит к противоречию, теорема доказана.

3.5. Комбинирование пары решений в случае лексикографической свертки критериев

Для многокритериальной задачи о назначениях в случае лексикографической свертки критериев при комбинировании двух допустимых решений x^1, x^2 был разработан полиномиальный алгоритм её решения.

Алгоритм 3.5. Решение задачи $Z_{\preccurlyeq}\big(W(x^1,x^2)\big)$.

Шаг 1. Построить граф $G = (V,A)$, где $V = \{I \cup J \cup K\}$, $A = \{(i,j),(i,k),(j,k) | (i,j,k) \in W(x^1,x^2)\}$.

Шаг 2. Найти компоненты связности $G_l = (V_l, A_l), l = \overline{1,q}$.

Шаг 3. Построить следующие множества

$$D_l^1 = \{(i,j,k) | (i,j,k) \in W(x^1), (i,j), (i,k), (j,k) \in A_l\},$$

$$D_l^2 = \{(i,j,k) | (i,j,k) \in W(x^2), (i,j), (i,k), (j,k) \in A_l\}.$$

Шаг 4. Пусть

$$P_l^1 = \left\{ p \middle| p = \underset{p \in \{1,2\}}{\operatorname{argmin}} \sum_{(i,j,k) \in D_l^p} c_{ijk}^1 \right\},$$

$$P_l^u = \left\{ p \middle| p = \underset{p \in P_l^{u-1}}{\operatorname{argmin}} \sum_{(i,j,k) \in D_l^p} c_{ijk}^u \right\}, u = \overline{2, M-1},$$

$$p_l^* = \underset{p \in P_l^{M-1}}{\operatorname{argmin}} \sum_{(i,j,k) \in D_l^p} c_{ijk}^M, l = \overline{1,q}.$$

Шаг 5. Решение задачи $Z_{\preccurlyeq}\left(W(x^1, x^2)\right)$ определяется по следующему алгоритму. Пусть $x_{ijk}^* := 0, i \in I, j \in J, k \in K$. Далее для каждого $l = \overline{1,q}$ выполнить $x_{ijk}^* := 1, (i,j,k) \in D_l^{p_l^*}$.

При этом значение критериев (2.5) на решении x^* определяется как

$$\sum_{l=1}^{q} \sum_{(i,j,k) \in D_l^{p_l^*}} c_{ijk}^u, u = \overline{1,M}.$$

Теорема 3.5. Алгоритм 3.5 находит решение задачи $Z_{\preccurlyeq}\left(W(x^1, x^2)\right)$.

Доказательство (от противного). Предположим, что решение, полученное алгоритмом 3.5 не является решением задачи $Z_{\preccurlyeq}\left(W(x^1, x^2)\right)$. Тогда существует допустимое решение x задачи $Z_{\preccurlyeq}\left(W(x^1, x^2)\right)$ такое, что не выполняется условие $x^* \preccurlyeq x$.

Согласно следствию 3.1 в каждой компоненте связности $V_l, l = \overline{1,q}$, любое допустимое решение задачи $Z_{\preccurlyeq}\left(W(x^1, x^2)\right)$ может содержать назначения, состоящие из троек только первого или только второго решения.

По предположению, существует компонента связности V_l, в которой назначения решения x не совпадают с назначениями решения x^*. Тогда построим решение x' следующим образом:

Шаг 1. $x^0 = x, l = 1$

Шаг 2. ${x^t}_{ijk} = \begin{cases} 1, \text{если } (i,j,k) \in D_l^{p_l^*} \\ 0, \text{если } (i,j,k) \in D_l^{3-p_l^*}, i \in I, j \in J, k \in K \\ {x^{t-1}}_{ijk}, \text{иначе} \end{cases}$

Шаг 3. Если $l = q$, закончить,

иначе $l = l + 1$, перейти на Шаг 2.

Тогда по построению $x^{l+1} \preccurlyeq x^l, l = \overline{0, q-1}$. При этом $x^q = x^*$, $x^0 = x$. Отсюда $x^* \preccurlyeq x$. Получили противоречие. Предположение неверно. Теорема доказана.

Утверждение 3.3. Алгоритм 3.5 требует $O(n)$ вычислительных операций.

Доказательство проводится аналогично теореме 3.2.

3.6. Комбинирование пары решений двухкритериальной задачи о назначениях при построении Парето-области

Для двухкритериальной задачи о назначениях был разработан полиномиальный алгоритм нахождения подмножества Парето-оптимальных решений задачи комбинирования двух допустимых решений x^1, x^2 исходной задачи.

Алгоритм 3.6. Построение подмножества Парето-оптимальных решений задачи $Z_2(W(x^1, x^2))$.

Шаг 1. Построить граф $G = (V, A)$, где $V = \{I \cup J \cup K\}$, $A = \{(i,j), (i,k), (j,k) | (i,j,k) \in W(x^1, x^2)\}$.

Шаг 2. Найти компоненты связности $G_l = (V_l, A_l), l = \overline{1, q}$.

Шаг 3. Построить следующие множества:

$$D_l^1 = \{(i,j,k) | (i,j,k) \in W(x^1), (i,j), (i,k), (j,k) \in A_l\},$$

$$D_l^2 = \{(i,j,k) | (i,j,k) \in W(x^2), (i,j), (i,k), (j,k) \in A_l\}.$$

Шаг 4. Пусть $P_l = \left\{p \middle| \sum_{(i,j,k)\in D_l^p} c_{ijk} = \min\limits_{p'\in\{1,2\}} \sum_{(i,j,k)\in D_l^{p'}} c_{ijk}, p \in \{1,2\}\right\}$, $p_l = \underset{p\in P_l}{\operatorname{argmin}} \sum_{(i,j,k)\in D_l^p} d_{ijk}, l = \overline{1,q}$.

Шаг 5. Построим Парето-оптимальное решение x^{*0} по следующему алгоритму. Пусть $x_{ijk}^{*0} \coloneqq 0, i \in I, j \in J, k \in K$. Далее для каждого $l = \overline{1,q}$ выполнить $x_{ijk}^{*0} \coloneqq 1, (i,j,k) \in D_l^{p_l}$.

Шаг 6. Построим следующее множество индексов компонент связности

$L = \left\{l \middle| \sum_{(i,j,k)\in D_l^{\overline{p_l}}} c_{ijk} > \sum_{(i,j,k)\in D_l^{p_l}} c_{ijk} \text{ и } \sum_{(i,j,k)\in D_l^{\overline{p_l}}} d_{ijk} < \sum_{(i,j,k)\in D_l^{p_l}} d_{ijk}, l \in \{1,\dots,q\}\right\}$,

здесь для удобства обозначим $\bar{p} = 3 - p$, при $p \in \{1,2\}$.

Шаг 7. Упорядочим это множество в порядке невозрастания величины $tg(l)$, где

$$tg(l) = \frac{\sum_{(i,j,k)\in D_l^{p_l}} d_{ijk} - \sum_{(i,j,k)\in D_l^{\overline{p_l}}} d_{ijk}}{\sum_{(i,j,k)\in D_l^{\overline{p_l}}} c_{ijk} - \sum_{(i,j,k)\in D_l^{p_l}} c_{ijk}}.$$

То есть пусть $L = \{l_1, \dots, l_{|L|}\}$ и $tg(l_s) \geq tg(l_{s+1}), \mathrm{s} = \overline{1, |L|-1}$.

Шаг 8. Построим $|L|$ Парето-оптимальных решений $x^{*s}, s = \overline{1,|L|}$ следующим способом.

Пусть $x_{ijk}^{*s} \coloneqq 0, i \in I, j \in J, k \in K$, далее для каждого $t = \overline{1,|q|}$ выполнить:

если $t \in \{l_1, \dots, l_s\}$, то $x_{ijk}^{*s} \coloneqq 1, (i,j,k) \in D_t^{\overline{p_t}}$,

иначе $x_{ijk}^{*\mathrm{s}} \coloneqq 1, (i,j,k) \in D_t^{p_t}$.

Приведем численный пример, иллюстрирующий работу Алгоритма 3.6.

Пример 3.1. Пример работы Алгоритма 3.6.

Пусть $n = 6$, матрицы c_{ijk}, d_{ijk} заданы следующим образом:

$$c_{ijk} = \begin{cases} 0, (i,j,k) \in \{(1,1,1),(1,1,2),(2,2,1),(3,3,3),(4,4,4),(3,3,4),(5,5,5),(5,5,6),(6,6,5)\}, \\ 1, (i,j,k) \in \{(4,4,3)\}, \\ 2, (i,j,k) \in \{(6,6,6)\}, \\ 3, (i,j,k) \in \{(2,2,2)\}, \\ 10 \text{ иначе.} \end{cases}$$

$$d_{ijk} = \begin{cases} 0, (i,j,k) \in \{(1,1,1),(1,1,2),(2,2,2),(3,3,3),(4,4,3),(3,3,4),(5,5,5),(5,5,6),(6,6,6)\}, \\ 1, (i,j,k) \in \{(2,2,1),(4,4,4)\}, \\ 4, (i,j,k) \in \{(6,6,5)\}, \\ 10, \text{иначе}. \end{cases}$$

Рассмотрим два допустимых решения x^1, x^2:

$$x^1_{ijk} = \begin{cases} 1, (i,j,k) \in \{(1,1,1),(2,2,2),(3,3,3),(4,4,4),(5,5,5),(6,6,6)\}, \\ 0 \text{ иначе}. \end{cases}$$

$$x^2_{ijk} = \begin{cases} 1, (i,j,k) \in \{(1,1,2),(2,2,1),(3,3,4),(4,4,3),(5,5,6),(6,6,5)\}, \\ 0 \text{ иначе}. \end{cases}$$

Опишем работу Алгоритма 3.6 на данном численном примере.

Граф G, полученный на шаге 1 будет иметь следующий вид:

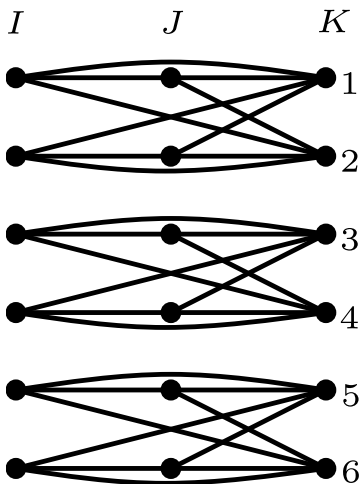

Рисунок 3.2. Граф G полученный на шаге 1.

На шаге 2 строятся компоненты связности графа G. Как мы видим граф G имеет три компоненты связности одинаковой структуры. Соответственно множества, получаемые на шаге 3 имеют следующий вид:

$$D^1_1 = \{(1,1,1),(2,2,2)\}, D^1_2 = \{(3,3,3),(4,4,4)\}, D^1_3 = \{(5,5,5),(6,6,6)\},$$

$$D^2_1 = \{(1,1,2),(2,2,1)\}, D^2_2 = \{(3,3,4),(4,4,3)\}, D^2_3 = \{(5,5,6),(6,6,5)\}.$$

На шаге 4 получим:

$$P_1 = \{2\}, p_1 = 2,$$

$$P_2 = \{1\}, p_2 = 1,$$

$$P_3 = \{2\}, p_3 = 2.$$

На шаге 5 первое полученное Парето-оптимальное решение x^{*0} будет построено следующим способом:

$$x_{ijk}^{*0} = \begin{cases} 1, & (i,j,k) \in \{(1,1,2),(2,2,1),(3,3,3),(4,4,4),(5,5,6),(6,6,5), \\ & 0 \text{ иначе.} \end{cases}$$

Значения критериев, достигаемые этим решением, равны $C(x^{*0}) = 0, D(x^{*0}) = 6$.

На шаге 6 множество L будет содержать все три компоненты связности и будет упорядочено на шаге 7 следующим образом: $l_1 = 3, l_2 = 2, l_3 = 1$. Следующие три Парето-оптимальные решения, построенные на шаге 8, имеют вид:

$$x_{ijk}^{*1} = \begin{cases} 1, & (i,j,k) \in \{(1,1,2),(2,2,1),(3,3,3),(4,4,4),(5,5,5),(6,6,6), \\ & 0 \text{ иначе,} \end{cases}$$

$$x_{ijk}^{*2} = \begin{cases} 1, & (i,j,k) \in \{(1,1,2),(2,2,1),(3,3,4),(4,4,3),(5,5,5),(6,6,6), \\ & 0 \text{ иначе,} \end{cases}$$

$$x_{ijk}^{*3} = \begin{cases} 1, & (i,j,k) \in \{(1,1,1),(2,2,2),(3,3,4),(4,4,3),(5,5,5),(6,6,6), \\ & 0 \text{ иначе.} \end{cases}$$

И соответствующие им значения критериев:

$$C(x^{*1}) = 2, D(x^{*1}) = 2,$$

$$C(x^{*2}) = 3, D(x^{*2}) = 1,$$

$$C(x^{*3}) = 6, D(x^{*2}) = 0.$$

Полученные решения отображены на плоскости CD, согласно значению их критерия (см. Рис. 3.3). Здесь x^1, x^2 – исходные решения, $x^{*0}, x^{*1}, x^{*2}, x^{*3}$ – найденные алгоритмом 3.6 Парето-оптимальные решения задачи комбинирования, x'^1, x'^2- возможные решения задачи комбинирования. Как можно увидеть на данном примере, кроме найденных алгоритмом решений существует ещё одно Парето-оптимальное решение x^1.

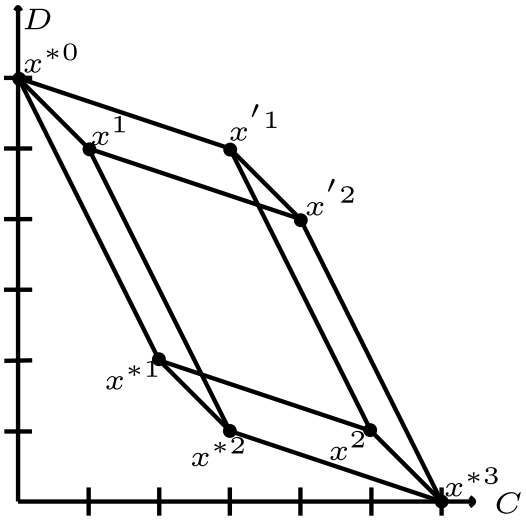

Рисунок 3.3. Найденные решения на плоскости CD.

Далее докажем корректность Алгоритма 3.6.

Рассмотрим многокритериальную задачу. Пусть $X \subseteq R^n -$ множество допустимых решений задачи, $\overrightarrow{f(x)} \in R^m -$ целевая функция задачи, $f_i(x) \to min, i = \overline{1,m}$.

Утверждение 3.4. Пусть множество $X \subseteq R^n$ – конечно. Если $x \in X$ не является Парето-оптимальным решением, то существует Парето-оптимальное решение $y \in X$, доминирующее x, то есть $f_i(y) \le f_i(x), i = \overline{1,m},$ и существует $j \in \{1, \dots, m\}$ такой, что $f_j(y) < f_j(x)$.

Доказательство. Пусть выполняются условия утверждения и $x \in X$ не является Парето-оптимальным решением. Тогда построим алгоритм на нахождения такого решения y.

Шаг 1. Найти решение x', доминирующее решение x. Если бы такого решения бы не нашлось, то решение x было бы Парето-оптимальным, что привело бы к противоречию.

Шаг 2. $x := x'$, если x Парето-оптимальное вернуть x, иначе перейти на Шаг 1.

Из определения Парето-оптимальности следует, что количество Шагов 2 алгоритма не превысит мощности множества решений. Так как множество конечно, то алгоритм найдет Парето-оптимальное решение за конечное число шагов. Утверждение доказано.

Утверждение 3.5. Для любого Парето-оптимального решения x' задачи $Z_2\big(W(x^1, x^2)\big)$ выполняется:

$$\sum_{(i,j,k)\in D_l^{p_l(x')}} c_{ijk} = \sum_{(i,j,k)\in D_l^{p_l(x^{*0})}} c_{ijk} \text{ и } \sum_{(i,j,k)\in D_l^{p_l(x')}} d_{ijk} = \sum_{(i,j,k)\in D_l^{p_l(x^{*0})}} d_{ijk}, l \notin L.$$

Доказательство (от противного). Предположим, что существует Парето-оптимальное решение x', для которого найдется $l \notin L$, что

$\sum_{(i,j,k)\in D_l^{p_l(x')}} c_{ijk} \neq \sum_{(i,j,k)\in D_l^{p_l(x^{*0})}} c_{ijk}$ или $\sum_{(i,j,k)\in D_l^{p_l(x')}} d_{ijk} \neq \sum_{(i,j,k)\in D_l^{p_l(x^{*0})}} d_{ijk}$.

Рассмотрим два случая:

1. $\sum_{(i,j,k)\in D_l^{p_l(x')}} c_{ijk} \neq \sum_{(i,j,k)\in D_l^{p_l(x^{*0})}} c_{ijk}$. Согласно Шагу 5 Алгоритма 3.6 выполняется

$\sum_{(i,j,k)\in D_l^{p_l(x')}} c_{ijk} > \sum_{(i,j,k)\in D_l^{p_l(x^{*0})}} c_{ijk}$.

Тогда

a. если $\sum_{(i,j,k)\in D_l^{p_l(x')}} d_{ijk} < \sum_{(i,j,k)\in D_l^{p_l(x^{*0})}} d_{ijk}$, то $l \in L$, получаем противоречие,

b. иначе выполняется $\sum_{(i,j,k)\in D_l^{p_l(x')}} d_{ijk} \geq \sum_{(i,j,k)\in D_l^{p_l(x^{*0})}} d_{ijk}$, тогда в решении x' тройки, соответствующие компоненте связности l можно заменить следующим образом

$$x'_{ijk} := 0, (i,j,k) \in D_l^{p_l(x')},$$

$$x'_{ijk} := 1, (i,j,k) \in D_l^{p_l(x^{*0})},$$

уменьшив при этом значение обоих критериев, следовательно x' не Парето-оптимальное решение, получаем противоречие.

2. $\sum_{(i,j,k)\in D_l^{p_l(x')}} c_{ijk} = \sum_{(i,j,k)\in D_l^{p_l(x^{*0})}} c_{ijk}$, $\sum_{(i,j,k)\in D_l^{p_l(x')}} d_{ijk} \neq \sum_{(i,j,k)\in D_l^{p_l(x^{*0})}} d_{ijk}$. Согласно Шагу 5 Алгоритма 3.6 выполняется

$\sum_{(i,j,k)\in D_l^{p_l(x')}} d_{ijk} > \sum_{(i,j,k)\in D_l^{p_l(x^{*0})}} d_{ijk}$,

тогда в решении x' тройки, соответствующие компоненте связности l можно заменить следующим образом

$$x'_{ijk} \coloneqq 0, (i,j,k) \in D_l^{p_l(x')},$$

$$x'_{ijk} \coloneqq 1, (i,j,k) \in D_l^{p_l(x^{*0})},$$

уменьшив при этом значение обоих критериев, следовательно, x' не Парето-оптимальное решение, получаем противоречие. Утверждение доказано.

Обозначим прямую, соединяющую точки x^{*t}, x^{*t+1} на плоскости CD, как $P_t, t = \overline{1,|L|}$. Запишем уравнение прямой P_t:

$$\frac{D - D(x^{*t})}{D(x^{*t-1}) - D(x^{*t})} = \frac{C - C(x^{*t})}{C(x^{*t-1}) - C(x^{*t})}.$$

Для удобства приведем уравнение к виду:

$$D = D(x^{*t}) + \frac{D(x^{*t}) - D(x^{*t-1})}{C(x^{*t-1}) - C(x^{*t})}(C(x^{*t}) - C) = D(x^{*t}) + tg(l_t)(C(x^{*t}) - C).$$

Уравнение прямой, также, можно записать в виде:

$$D = D(x^{*t-1}) + tg(l_t)(C(x^{*t-1}) - C).$$

Будем говорить, что допустимое решение x задачи $Z_2(W(x^1, x^2))$ лежит не ниже прямой P_t, если

$$D(x) \geq D(x^{*t}) + tg(l_t)\big(C(x^{*t}) - C(x)\big).$$

Несложно увидеть, что если решение x лежит не ниже прямой P_t, то решение x не будет доминировать ни одно из решений x^{*t-1}, x^{*t}.

Утверждение 3.6. Если для некоторого $e \in \{1, \dots, |L|\}$ решение x лежит не ниже прямой P_e и $C(x) \leq C(x^{*e})$, то оно лежит не ниже любой прямой $P_t, t = \overline{e,|L|}$.

Доказательство. Т.к. x лежит не ниже прямой P_e, то

$$D(x) \geq D(x^{*e}) + tg(l_e)\big(C(x^{*e}) - C(x)\big).$$

По построению $tg(l_t) \geq tg(l_{t+1}) \geq 0, t = \overline{1,|L|-1}$. Из условия утверждения $\big(C(x^{*e}) - C(x)\big) \geq 0$. Отсюда

$$tg(l_e)\big(C(x^{*e}) - C(x)\big) \geq tg(l_{e+1})\big(C(x^{*e}) - C(x)\big)$$

и следовательно

$$D(x) \geq D(x^{*e}) + tg(l_{e+1})\big(C(x^{*e}) - C(x)\big).$$

То есть решение x лежит не ниже прямой P_{e+1}. Отсюда по индукции получили, что x лежит не ниже прямой $P_t, t \geq e$. Утверждение доказано.

Теорема 3.6. Не существует Парето-оптимального решения x' задачи $Z_2\big(W(x^1, x^2)\big)$, для которого выполняется

$\sum_{(i,j,k)\in D_l^{p_l(x')}} c_{ijk} = \sum_{(i,j,k)\in D_l^{p_l(x^{*0})}} c_{ijk}$ и $\sum_{(i,j,k)\in D_l^{p_l(x')}} d_{ijk} =$

$\sum_{(i,j,k)\in D_l^{p_l(x^{*0})}} d_{ijk}, l \notin L,$

доминирующего какое-либо из решений, построенных Алгоритмом 3.6.

Доказательство (от противного). Предположим, что существует Парето-оптимальное решение x', доминирующее хотя бы одно из решений $x^{*0}, \dots, x^{*|L|}$, построенных алгоритмом.

Построим $|L| + 1$ решение $x^{"s}, s = \overline{0, |L|}$ следующим способом:

Шаг 1. Пусть $x_{ijk}^{"s} := 0, i \in I, j \in J, k \in K$.

Шаг 2. Далее для каждого $t = \overline{1, |q|}$ выполнить:

- если $t \in \{l_1, \dots, l_s\}$ и $p_t(x') \neq p_t(x^{*0})$, то $x_{ijk}^{*s} := 1, (i,j,k) \in D_t^{\overline{p_t}}$,

- иначе $x_{ijk}^{*s} := 1, (i,j,k) \in D_t^{p_t}$.

По построению $x^{"0} = x^{*0}$, отсюда решение $x^{"0}$ лежит не ниже прямой P_1.

Покажем, что если решение $x^{"s}$ лежит не ниже прямых P_t, $t = \overline{1, |L|}$, то и решение $x^{"s+1}$ лежит не ниже прямых $P_t, t = \overline{1, |L|}$.

Для каждой из прямых $P_t, t = \overline{1, r+1}$ возможны 2 случая:

1. $p_{l_{s+1}}(x') = p_{l_{s+1}}(x^{*0})$, тогда решение $x^{"s+1}$ совпадает с решением $x^{"s}$, а значит оно не ниже прямой P_t,

2. $p_{l_{s+1}}(x') \neq p_{l_{s+1}}(x^{*0})$, тогда из условия, что решение $x^{"s}$ лежит не ниже прямой P_t следует, что

$$\mathrm{D}\big(x^{"s}\big) \geq D(x^{*t}) + tg(l_t)\left(C(x^{*t}) - C\big(x^{"s}\big)\right). \quad (3.1)$$

Из построения:

$$\mathrm{D}\left(x^{"s+1}\right) = D\left(x^{"s}\right) + tg(l_{s+1})\left(C\left(x^{"s}\right) - C\left(x^{"s+1}\right)\right).$$

Применим неравенство (3.1):

$$\mathrm{D}\left(x^{"s+1}\right) \geq D(x^{*t}) + tg(l_t)\left(C(x^{*t}) - C\left(x^{"s}\right)\right) + tg(l_{s+1})\left(C\left(x^{"s}\right) - C\left(x^{"s+1}\right)\right).$$

Отсюда:

$$\mathrm{D}\left(x^{"s+1}\right) \geq D(x^{*t}) + tg(l_t)\left(C(x^{*t}) - C\left(x^{"s+1}\right)\right) +$$

$$+(tg(l_t) - tg(l_{s+1}))\left(C\left(x^{"s+1}\right) - C\left(x^{"s}\right)\right).$$

По построению $C\left(x^{"s+1}\right) - C\left(x^{"s}\right) \geq 0$ и $tg(l_t) - tg(l_{s+1}) \geq 0$ при $s+1 \geq t$. Тогда:

$$D\left(x^{"s+1}\right) \geq D(x^{*t}) + tg(l_t)\left(C(x^{*t}) - C\left(x^{"s+1}\right)\right).$$

То есть, решение $x^{"s+1}$ не ниже прямой P_t.

Отсюда если решение $x^{"s}$ лежит не ниже прямых P_t, $t = \overline{1,|L|}$, то и решение $x^{"s+1}$ лежит не ниже прямых $P_t, t = \overline{1,s+1}$.

По построению $C\left(x^{"s+1}\right) \leq C(x^{*s+1})$. Тогда по утверждению 4 решение $x^{"s+1}$ лежит не ниже прямых $P_t, t = \overline{s+1,|L|}$ (здесь $e = s+1$). Следовательно, если решение $x^{"s}$ лежит не ниже прямых $P_t, t = \overline{1,|L|}$, то решение $x^{"s+1}$ не ниже прямых $P_t, t = \overline{1,|L|}$. Отсюда по индукции получаем, что $x^{"|L|}$ не ниже прямых $P_t, t = \overline{1,|L|}$.

По построению $x^{"|L|} = x'$, то есть x' лежит не ниже прямых $P_t, t = \overline{1,|L|}$, а следовательно решение x' не доминирует ни одно из решений $x^{*0}, \dots, x^{*|L|}$, построенных алгоритмом 1. Получаем противоречие, предположение неверно. Теорема доказана.

Теорема 3.7. Решения $x^{*0}, \dots, x^{*|L|}$, найденные Алгоритмом 3.6 являются Парето-оптимальными решениями задачи $Z_2\left(W(x^1, x^2)\right)$.

Доказательство (от противного). Предположим, что это не так, тогда существует решение $x^{*q'}$, $q' \in \{0, \dots, L\}$, не являющееся Парето-оптимальным решением задачи $Z_2\left(W(x^1, x^2)\right)$. Тогда согласно Утверждению 3.3 существует Парето-оптимальное решение x' задачи $Z_2\left(W(x^1, x^2)\right)$, доминирующее $x^{*q'}$:

$$C(x') < C(x^{*q'}) \text{ и } D(x') < \mathrm{D}(x^{*q'}).$$

Тогда в соответствии с утверждением 3.4 выполняется

$\sum_{(i,j,k)\in D_l^{p_l(x')}} c_{ijk} = \sum_{(i,j,k)\in D_l^{p_l(x^{*0})}} c_{ijk}$ и $\sum_{(i,j,k)\in D_l^{p_l(x')}} d_{ijk} = \sum_{(i,j,k)\in D_l^{p_l(x^{*0})}} d_{ijk}, l \notin L.$

Т.е. решение x' может отличаться от x^{*0} только в тройках компонент $l \in L$. Но согласно Теореме 3.6 такого решения x' не существует. Получаем противоречие. Предположение неверно. Теорема доказана.

Утверждение 3.7. Алгоритм 3.6 не гарантирует нахождение всего множества Парето-оптимальных решений задачи $Z_2\big(W(x^1, x^2)\big)$.

Доказательство. Построим численный пример. Пусть $n = 4$ и

$$x_{ijk}^1 = \begin{cases} 1, \text{если } (i,j,k) \in \{(1,1,1),(2,2,2),(3,3,3),(4,4,4)\}, \\ 0 \text{ иначе}, \end{cases}$$

$$x_{ijk}^2 = \begin{cases} 1, \text{если } (i,j,k) \in \{(1,1,2),(2,2,1),(3,3,4),(4,4,3)\}, \\ 0, \text{иначе}. \end{cases}$$

Пусть

$$c_{ijk} = \begin{cases} 0, \text{если } (i,j,k) \in \{(1,1,1),(2,2,2),(3,3,3),(4,4,4),(2,2,1),(4,4,3)\}, \\ 1, \text{если } (i,j,k) = (1,1,2), \\ 2, \text{если } (i,j,k) = (3,3,4), \\ 5 \text{ иначе}, \end{cases}$$

$$d_{ijk} = \begin{cases} 0, \text{если } (i,j,k) \in \{(2,2,2),(4,4,4),(1,1,2),(2,2,1),(3,3,4),(4,4,3)\}, \\ 1, \text{если } (i,j,k) = (1,1,1), \\ 2, \text{если } (i,j,k) = (3,3,3), \\ 5 \text{ иначе}. \end{cases}$$

Тогда существует 4 допустимых решения задачи комбинирования:

$$x_{ijk}^{*1} = \begin{cases} 1, \text{если } (i,j,k) \in \{(1,1,1),(2,2,2),(3,3,3),(4,4,4)\}, \\ 0 \text{ иначе}, \end{cases}$$

здесь $C(x^{*1}) = 0, D(x^{*1}) = 3$.

$$x_{ijk}^{*2} = \begin{cases} 1, \text{если } (i,j,k) \in \{(1,1,2),(2,2,1),(3,3,4),(4,4,3)\}, \\ 0 \text{ иначе}, \end{cases}$$

здесь $C(x^{*2}) = 3, D(x^{*2}) = 0$.

$$x_{ijk}^{*3} = \begin{cases} 1, \text{если } (i,j,k) \in \{(1,1,2),(2,2,1),(3,3,3),(4,4,4)\}, \\ 0 \text{ иначе}, \end{cases}$$

здесь $C(x^{*3}) = 1, D(x^{*3}) = 2.$

$$x_{ijk}^{*4} = \begin{cases} 1, \text{если } (i,j,k) \in \{(1,1,1),(2,2,2),(3,3,4),(4,4,3)\}, \\ 0 \text{ иначе,} \end{cases}$$

здесь $C(x^{*1}) = 2, D(x^{*1}) = 1.$

Каждое из построенных 4 решений является Парето-оптимальным решением задачи $Z_2\big(W(x^1, x^2)\big)$. При этом $q = 2$, $|L| = 2$ и, следовательно, Алгоритм 3.6 найдет только 3 решения. Утверждение доказано.

Таким образом предложенный алгоритм 3.6 гарантирует построение Парето-оптимальных решений задачи $Z_2\big(W(x^1, x^2)\big)$, однако не гарантирует построение всей области Парето.

Теорема 3.8. Алгоритм 3.6 требует $O(n^2)$ вычислительных операций.

Доказательство. Пусть входными данными алгоритма 3.6 являются допустимые решения x^1, x^2, представленные в виде коллекции троек (i, j, k) и стоимостей c_{ijk}, для которых соответствующие переменные принимают значение 1. Согласно (2.1)–(2.4) для каждого из допустимых решений количество таких троек составляет n. Тогда на шаге 1 алгоритма строится граф $G = (V, A)$, что $|V| = O(n), |A| = O(n)$. На шаге 2 граф G разбивается на компоненты связности. Это осуществляется при помощи обхода графа в ширину, который требует $O(|V| + |A|) = O(n)$ вычислительных операций. На шаге 3 для входных троек алгоритма определяются соответствующие компоненты связности, этот шаг требует $O(n)$ вычислительных операций. На шаге 4 для каждой компоненты связности определяется из какого решения будет взяты тройки в первое найденное Парето-оптимальное решение, данный шаг требует $O(n)$ вычислительных операций. На шаге 5 строится первое Парето-оптимальное решение, так как мы можем представить решение в виде коллекции из n троек (i, j, k), то этот шаг требует $O(n)$ вычислительных операций. На шаге 6 из всех компонент связности выбираются только те, что соответствуют критерию, что требует $O(n)$ вычислительных операций. На шаге 7 подмножество компонент связности упорядочивается, что требует $O\big(nlog(n)\big)$ вычислительных операций. И, наконец, на шаге 8 строится

$O(n)$ решений, каждое из которых строится за $O(n)$ вычислительных операций. Таким образом, алгоритм 3.6 требует $O(n^2)$ вычислительных операций. Теорема доказана.

3.7. NP-трудность задачи комбинирования четырех решений аксиальной задачи о назначениях

Покажем, что класс задач оптимального комбинирования m допустимых решений является NP-трудным уже при $m = 4$.

Лемма 3.3. Класс задач 3-SAT [53] полиномиально сводим к классу задач 3-SAT без повторяющихся переменных в каждом дизъюнкте.

Доказательство. Рассмотрим формулу из произвольной задачи класса 3-SAT. К каждому дизъюнкту формулы применим следующий алгоритм:

a. Если дизъюнкт не содержит повторяющихся переменных, то дизъюнкт остается без изменений.

b. Если повторяющаяся переменная дизъюнкта входит в дизъюнкт только с отрицанием или только без отрицания, то дизъюнкт имеет вид: $(x \vee x \vee y)$ или $(x \vee x \vee x)$, где x, y - литералы (булевы переменные или их отрицание).

Дизъюнкт $u(x, y) = (x \vee x \vee y)$ заменим на $u'(x, y, z) = (x \vee y \vee z) \wedge (x \vee y \vee \bar{z})$, где z – новая булева переменная. Очевидно, $u(x, y) = u'(x, y, z)$, $\forall z$.

Дизъюнкт $u(x) = (x \vee x \vee x)$ заменим на $u'(x, z, w) = (x \vee z \vee w) \wedge (x \vee z \vee \overline{w}) \wedge (x \vee \bar{z} \vee w) \wedge (x \vee \bar{z} \vee \overline{w})$, где z, w – новые булевы переменные. Очевидно, $u(x) = u'(x, z, w)$, $\forall z, w$.

c. Если повторяющаяся переменная дизъюнкта одновременно входит в него с отрицанием и без отрицания, то дизъюнкт имеет вид $(x \vee \bar{x} \vee y)$, где x – булева переменная, y – литерал. Тогда $(x \vee \bar{x} \vee y) \equiv 1$ и дизъюнкт можно исключить из формулы.

Отсюда класс задач 3-SAT полиномиально сводим к классу задач 3-SAT без повторяющихся переменных в каждом дизъюнкте. Лемма доказана.

Теорема 3.9. Класс задач оптимального комбинирования 4 допустимых решений является NP-трудным.

Доказательство. Для доказательства покажем, что классическая NP-трудная задача 3-SAT [53] полиномиально сводима к задаче оптимального комбинирования 4 допустимых решений.

Рассмотрим формулу из произвольной задачи класса 3-SAT, содержащую N булевых переменных и M дизъюнктов. Согласно лемме 3.3, не уменьшая общности, будем предполагать, что каждый дизъюнкт не содержит повторяющихся переменных.

Для удобства введем следующие обозначения

- $l_1(s), l_2(s), l_3(s)$ – номера булевых переменных входящий в s-ый дизъюнкт,
- $L(s) = \{l_1(s) \cup l_2(s) \cup l_3(s)\}$ – множество номеров булевых переменных входящих в s-ый дизъюнкт,
- $L^+(s) \subseteq L(s)$ – множество номеров булевых переменных входящих в s-ый дизъюнкт без отрицания,
- $L^-(s) \subseteq L(s)$ – множество номеров булевых переменных входящих в s-ый дизъюнкт с отрицанием,
- $\bar{L}(s) = \{1, \dots, N\} \backslash L(s)$ – множество номеров булевых переменных не входящих в s-ый дизъюнкт,

$s = \overline{1, M}$.

Тогда построим непересекающиеся множества индексов I, J, K следующим образом:

- $I = \{a_{ls}^1 \big| l = \overline{1, N}, s = \overline{1, M}\} \cup \{d_s^1, q_s^1, w_s^1 | s = \overline{1, M}\} \cup \{e_{ls}^1 | l \in \bar{L}(s), s = \overline{1, M}\}$,
- $J = \{a_{ls}^2 \big| l = \overline{1, N}, s = \overline{1, M}\} \cup \{d_s^2, q_s^2, w_s^2 | s = \overline{1, M}\} \cup \{e_{ls}^2 | l \in \bar{L}(s), s = \overline{1, M}\}$,
- $K = \{b_{ls}^1, b_{ls}^2 \big| l = \overline{1, N}, s = \overline{1, M}\}$.

По построению $|I| = |J| = NM + 3M + (N - 3)M = 2NM, \quad |K| = 2NM$. Следовательно, $|I| = |J| = |K| = 2NM$.

Далее построим множество $R \subseteq I \times J \times K$, которое будет использовано для определения многоиндексной матрицы стоимостей аксиальной задачи о назначениях, следующим образом:

- $R_1 = \{(a_{ls}^1, a_{ls}^2, b_{ls}^1), (a_{l(s \bmod M+1)}^1, a_{ls}^2, b_{ls}^2) | l = \overline{1,N}, s = \overline{1,M}\}$,
- $R_2 = \{(d_s^1, d_s^2, b_{ls}^1) | l \in L^-(s), s = \overline{1,M}\} \cup \{(d_s^1, d_s^2, b_{ls}^2) | l \in L^+(s), s = \overline{1,M}\}$,
- $R_3 = \{(q_s^1, q_s^2, b_{l_1(s)s}^1), (q_s^1, q_s^2, b_{l_1(s)s}^2), (q_s^1, q_s^2, b_{l_2(s)s}^1), (q_s^1, q_s^2, b_{l_2(s)s}^2) | s = \overline{1,M}\}$,
- $R_4 = \{(w_s^1, w_s^2, b_{l_2(s)s}^1), (w_s^1, w_s^2, b_{l_2(s)s}^2), (w_s^1, w_s^2, b_{l_3(s)s}^1), (w_s^1, w_s^2, b_{l_3(s)s}^2) | s = \overline{1,M}\}$,
- $R_5 = \{(e_{ls}^1, e_{ls}^2, b_{ls}^1), (e_{ls}^1, e_{ls}^2, b_{ls}^2) | l \in \bar{L}(s), s = \overline{1,M}\}$,
- $R = R_1 \cup R_2 \cup R_3 \cup R_4 \cup R_5$.

Определим трехиндексную матрицу стоимостей $c_{ijk} = \begin{cases} 0, (i,j,k) \in R \\ 1, \text{иначе} \end{cases}$, $i \in I, j \in J, k \in K$.

Таким образом построенные множества I, J, K и трехиндексная матрица стоимостей $\|c_{ijk}\|$ определяют многоиндексную аксиальную задачу о назначениях (1.1) - (1.5).

Далее построим 4 подмножества $P_1, P_2, P_3, P_4 \subseteq I \times J \times K$, которые будут определять 4 допустимых решения задачи (1.1) - (1.5):

- $P_1 = \{(a_{ls}^1, a_{ls}^2, b_{ls}^1) | l = \overline{1,N}, s = \overline{1,M}\} \cup \{(q_s^1, q_s^2, b_{l_2(s)s}^2), (w_s^1, w_s^2, b_{l_3(s)s}^2) | s = \overline{1,M}\} \cup \{(d_s^1, d_s^2, b_{l_1(s)s}^2) | s = \overline{1,M}\} \cup \{(e_{ls}^1, e_{ls}^2, b_{ls}^2) | l \in \bar{L}(s), s = \overline{1,M}\}$,
- $P_2 = \{(a_{l(s \bmod M+1)}^1, a_{ls}^2, b_{ls}^2) | l = \overline{1,N}, s = \overline{1,M}\} \cup \{(q_s^1, q_s^2, b_{l_2(s)s}^1), (w_s^1, w_s^2, b_{l_3(s)s}^1) | s = \overline{1,M}\} \cup \{(d_s^1, d_s^2, b_{l_1(s)s}^1) | s = \overline{1,M}\} \cup \{(e_{ls}^1, e_{ls}^2, b_{ls}^1) | l \in \bar{L}(s), s = \overline{1,M}\}$,
- $P_3 = \{(q_s^1, q_s^2, b_{l_1(s)s}^2), (w_s^1, w_s^2, b_{l_2(s)s}^2), (a_{l_1(s)s}^1, a_{l_1(s)s}^2, b_{l_1(s)s}^1) | s = \overline{1,M}\} \cup \cup \{(d_s^1, d_s^2, b_{l_2(s)s}^1), (a_{l_2(s)s}^1, a_{l_2(s)s}^2, b_{l_3(s)s}^1), (a_{l_3(s)s}^1, a_{l_3(s)s}^2, b_{l_3(s)s}^2) | l_2(s) \in L^-(s), s = \overline{1,M}\} \cup \cup \{(d_s^1, d_s^2, b_{l_3(s)s}^1), (a_{l_3(s)s}^1, a_{l_3(s)s}^2, b_{l_3(s)s}^2) | l_2(s) \in L^+(s), l_3(s) \in L^-(s), s = \overline{1,M}\} \cup$

$$\cup \left\{\left(d_s^1, d_s^2, b_{l_3(s)s}^2\right), \left(a_{l_3(s)s}^1, a_{l_3(s)s}^2, b_{l_3(s)s}^1\right) | l_2(s) \in L^+(s), l_3(s) \in L^+(s), s = \overline{1,M}\right\} \cup$$

$$\cup \left\{\left(a_{l_2(s)s}^1, a_{l_2(s)s}^2, b_{l_2(s)s}^1\right) | l_2(s) \in L^+(s), s = \overline{1,M}\right\} \cup$$

$$\cup \left\{\left(a_{ls}^1, a_{ls}^2, b_{ls}^1\right), \left(e_{ls}^1, e_{ls}^2, b_{ls}^2\right) \middle| l \in \bar{L}(s), s = \overline{1,M}\right\}$$

- $P_4 = \left\{\left(q_s^1, q_s^2, b_{l_1(s)s}^1\right), \left(w_s^1, w_s^2, b_{l_2(s)s}^1\right), \left(a_{l_1(s)s}^1, a_{l_1(s)s}^2, b_{l_1(s)s}^2\right) | s = \overline{1,M}\right\} \cup$

$$\cup \left\{\left(d_s^1, d_s^2, b_{l_2(s)s}^2\right), \left(a_{l_3(s)s}^1, a_{l_3(s)s}^2, b_{l_3(s)s}^1\right), \left(a_{l_2(s)s}^1, a_{l_2(s)s}^2, b_{l_3(s)s}^2\right) | l_2(s) \in L^+(s), s = \overline{1,M}\right\} \cup$$

$$\cup \left\{\left(d_s^1, d_s^2, b_{l_3(s)s}^1\right), \left(a_{l_3(s)s}^1, a_{l_3(s)s}^2, b_{l_3(s)s}^2\right) | l_2(s) \in L^-(s), l_3(s) \in L^-(s), s = \overline{1,M}\right\} \cup$$

$$\cup \left\{\left(d_s^1, d_s^2, b_{l_3(s)s}^2\right), \left(a_{l_3(s)s}^1, a_{l_3(s)s}^2, b_{l_3(s)s}^1\right) | l_2(s) \in L^-(s), l_3(s) \in L^+(s), s = \overline{1,M}\right\} \cup$$

$$\cup \left\{\left(a_{l_2(s)s}^1, a_{l_2(s)s}^2, b_{l_2(s)s}^2\right) | l_2(s) \in L^-(s), s = \overline{1,M}\right\} \cup$$

$$\cup \left\{\left(a_{ls}^1, a_{ls}^2, b_{ls}^2\right), \left(e_{ls}^1, e_{ls}^2, b_{ls}^1\right) | l \in \bar{L}(s), s = \overline{1,M}\right\}.$$

Соответствующие допустимые решения x^t, $t = \overline{1,4}$, определим следующим образом:

$$x_{ijk}^t = \begin{cases} 1, \text{если } (i,j,k) \in P_t, i \in I, j \in J, k \in K, \\ 0 \text{ иначе} \end{cases}$$

где $t \in \{1,2,3,4\}$. Очевидно, значение критерия построенной задачи комбинирования $Z\left(W(x^1, x^2, x^3, x^4)\right)$ принимает неотрицательное значение.

Покажем, что оптимальное значение критерия задачи $Z\left(W(x^1, x^2, x^3, x^4)\right)$ равно 0 тогда и только тогда, когда формула выполнима.

1. Пусть x^* оптимальное решение задачи $Z\left(W(x^1, x^2, x^3, x^4)\right)$ и $C(x^*) = 0$. Построим следующее множество $W(x^*) = \left\{(i,j,k) \middle| x_{ijk}^* = 1, i \in I, j \in J, k \in K\right\}$.

Т.к. x^* удовлетворяет системе ограничений (2.1)–(2.4), то $|W(x^*)| = 2NM$.

Далее легко увидеть, что для каждого $l \in \{1,2,\dots,N\}$ выполняется одно из следующих двух условий:

$$\left(a_{ls}^1, a_{ls}^2, b_{ls}^1\right) \in P(x^*), s = \overline{1,M}, \tag{3.2}$$

или

$$\left(a^1_{l(s\ mod\ M+1)}, a^2_{ls}, b^2_{ls}\right) \in P(x^*), s = \overline{1,M}, \quad (3.3)$$

Действительно, предположим, что для произвольного $l \in \{1,2,\dots,N\}$ выполняется $(a^1_{ls}, a^2_{ls}, b^1_{ls}) \in W(x^*)$, но $\left(a^1_{l(s\ mod\ M+1)}, a^2_{l(s\ mod\ M+1)}, b^1_{l(s\ mod\ M+1)}\right) \notin W(x^*)$. По построению

$$P \cap \left\{\left(a^1_{(l\ mod\ M+1)s}, j, k\right) | j \in J, k \in K\right\} =$$

$$= \left\{\left(a^1_{l(s\ mod\ M+1)}, a^2_{l(s\ mod\ M+1)}, b^1_{l(s\ mod\ M+1)}\right), \left(a^1_{l(s\ mod\ M+1)}, a^2_{ls}, b^2_{ls}\right)\right\}.$$

Т.к. $(a^1_{ls}, a^2_{ls}, b^1_{ls}) \in W(x^*)$ и x^* удовлетворяет системе ограничений (2.1)–(2.4), то $\left(a^1_{l(s\ mod\ M+1)}, a^2_{ls}, b^2_{ls}\right) \notin W(x^*)$. Отсюда т.к. $W(x^*) \subseteq P$, то

$$W(x^*) \cap \left\{\left(a^1_{l(s\ mod\ M+1)}, j, k\right) | j \in J, k \in K\right\} = \emptyset.$$

Следовательно, $|W(x^*)| < 2NM$, получаем противоречие, предположение неверно. Отсюда, если $(a^1_{ls}, a^2_{ls}, b^1_{ls}) \in P(x^*)$, то $\left(a^1_{l(s\ mod\ M+1)}, a^2_{l(s\ mod\ M+1)}, b^1_{l(s\ mod\ M+1)}\right) \in W(x^*)$. Таким образом, если для выбранного l выполняется $(a^1_{ls}, a^2_{ls}, b^1_{ls}) \in W(x^*)$, то для l выполняется условие (3.2). Если для выбранного l выполняется $(a^1_{ls}, a^2_{ls}, b^1_{ls}) \notin W(x^*)$, то $\left(a^1_{l(s\ mod\ M+1)}, a^2_{ls}, b^2_{ls}\right) \in W(x^*)$ и аналогичным образом доказывается, что выполняется условие (3.3).

Тогда определим вектор булевых переменных X исходной формулы:

$$X_l = \begin{cases} true, \text{если для } l \text{ выполняется условие (3.2)} \\ false, \text{если для } l \text{ выполняется условие (3.3)} \end{cases}, l = \overline{1,N}.$$

По построению для каждого $s \in \{1,\dots,M\}$ существует $l \in L^-(s)$, что $(d^1_s, d^2_s, b^1_{ls}) \in W(x^*)$ или существует $l \in L^+(s)$, что $(d^1_s, d^2_s, b^2_{ls}) \in W(x^*)$. Отсюда для каждого $s \in \{1,\dots,M\}$ существует $l \in L^+(s)$, что $X_l = true$ или $l \in L^-(s)$, что $X_l = false$. Отсюда каждый дизъюнкт принимает значение $true$ на булевом векторе переменных X. Следовательно, формула принимает значение $true$ на булевом векторе X и является выполнимой.

2. Пусть формула выполнима и X – вектор булевых переменных, при котором формула принимает значение истина. Тогда построим множество разрешенных

назначений, обозначаемое $P(X)$, которое будет определять оптимальное решение задачи комбинирования $Z\big(W(x^1, x^2, x^3, x^4)\big)$. Множество $P(X)$ будем строить следующим алгоритмом.

Шаг 1. Изначально положим $P(X) := \emptyset$.

Шаг 2. Для каждого $l = \overline{1, N}$:

Если $X_l = true$, то

$P(X) := P(X) \cup \{(a_{ls}^1, a_{ls}^2, b_{ls}^1) \big| s = \overline{1, M}\} \cup \{(e_{ls}^1, e_{ls}^2, b_{ls}^2) | l \in \bar{L}(s), s = \overline{1, M}\}$;

Иначе

$$P(X) := P(X) \cup \left\{\left(a_{l(s \bmod M+1)}^1, a_{ls}^2, b_{ls}^2\right) \big| s = \overline{1, M}\right\}$$

$$\cup \{(e_{ls}^1, e_{ls}^2, b_{ls}^1) \big| l \in \bar{L}(s), s = \overline{1, M}\}.$$

Шаг 3. Для каждого $s = \overline{1, M}$:

Если $X_{l_1(s)} = true, l_1(s) \in L^+(s)$ или $X_{l_1(s)} = false, l_1(s) \in L^-(s)$, то $P(X) :=$ $P(X) \cup$

$$\cup \{(d_s^1, d_s^2, b_{l_1(s)s}^2) | X_{l_1(s)} = true\} \cup \{(d_s^1, d_s^2, b_{l_1(s)s}^1) | X_{l_1(s)} = false\} \cup$$

$$\cup \{(q_s^1, q_s^2, b_{l_2(s)s}^2) | X_{l_2(s)} = true\} \cup \{(q_s^1, q_s^2, b_{l_2(s)s}^1) | X_{l_2(s)} = false\} \cup$$

$$\cup \{(w_s^1, w_s^2, b_{l_3(s)s}^2) | X_{l_3(s)} = true\} \cup \{(w_s^1, w_s^2, b_{l_3(s)s}^1) | X_{l_3(s)} = false\};$$

Иначе, если $X_{l_2(s)} = true, l_2(s) \in L^+(s)$ или $X_{l_2(s)} = false, l_2(s) \in L^-(s)$, то $P(X) := P(X) \cup$

$$\cup \{(d_s^1, d_s^2, b_{l_2(s)s}^2) | X_{l_2(s)} = true\} \cup \{(d_s^1, d_s^2, b_{l_2(s)s}^1) | X_{l_2(s)} = false\} \cup$$

$$\cup \{(q_s^1, q_s^2, b_{l_1(s)s}^2) | X_{l_1(s)} = true\} \cup \{(q_s^1, q_s^2, b_{l_1(s)s}^1) | X_{l_1(s)} = false\} \cup$$

$$\cup \{(w_s^1, w_s^2, b_{l_3(s)s}^2) | X_{l_3(s)} = true\} \cup \{\left(w_s^1, w_s^2, b_{l_3(s)s}^1\right) | X_{l_3(s)} = false\};$$

Иначе, если $X_{l_3(s)} = true, l_3(s) \in L^+(s)$ или $X_{l_3(s)} = false, l_3(s) \in L^-(s)$, то $P(X) := P(X) \cup$

$$\cup \{(d_s^1, d_s^2, b_{l_3(s)s}^2) | X_{l_3(s)} = true\} \cup \{(d_s^1, d_s^2, b_{l_3(s)s}^1) | X_{l_3(s)} = false\} \cup$$

$$\cup \{(q_s^1, q_s^2, b_{l_1(s)s}^2) | X_{l_1(s)} = true\} \cup \{(q_s^1, q_s^2, b_{l_1(s)s}^1) | X_{l_1(s)} = false\} \cup$$

$$\cup \{(w_s^1, w_s^2, b_{l_2(s)s}^2) | X_{l_2(s)} = true\} \cup \{\left(w_s^1, w_s^2, b_{l_2(s)s}^1\right) | X_{l_2(s)} = false\}.$$

Далее определим многоиндексную матрицу неизвестных $x^* = \left\|x_{ijk}^*\right\|$:

$$x^*_{ijk} = \begin{cases} 1, \text{если } (i,j,k) \in P(X) \\ 0, \text{иначе} \end{cases}, i \in I, j \in J, k \in K.$$

По построению на шаге 2 в множество $P(X)$ включается $NK + (N-3)K$ элементов. Т.к. формула принимает значение истина на векторе булевых переменных X, то на шаге 3 в множество $P(X)$ включается $3K$ элементов. Отсюда $|P(X)| = 2NK$. Для любой пары $p_1, p_2 \in P(x)$, что $p_1 \neq p_2$, выполняются условия $p_1 = (i_1, j_1, k_1)$, $p_2 = (i_2, j_2, k_2)$, $i_1 \neq i_2$, $j_1 \neq j_2$, $k_1 \neq k_2$. Следовательно, x^* удовлетворяет системе ограничений (2.1)–(2.4).

По построению $P(X) \subseteq P$, т.к. на шаге 2 в $P(X)$ включаются элементы из множеств P_1, P_2, на шаге 3 элементы из множеств P_1, P_2, P_3, P_4. Отсюда x^* является допустимым решением задачи комбинирования $Z\big(W(x^1, x^2, x^3, x^4)\big)$. Далее по построению $P(X) \subseteq R$, т.к. на шаге 2 в $P(X)$ включаются элементы из множеств R_1, R_5, на шаге 3 элементы из множеств R_2, R_3, R_4. Отсюда $C(x^*) = 0$. Следовательно, x^* является оптимальным решением задачи $Z\big(W(x^1, x^2, x^3, x^4)\big)$ и оптимальное значение критерия задачи $Z\big(W(x^1, x^2, x^3, x^4)\big)$ равно 0.

Таким образом оптимальное значение критерия построенной задачи $Z\big(W(x^1, x^2, x^3, x^4)\big)$ равно 0 тогда и только тогда, когда формула выполнима. Приведенная процедура построения задачи $Z\big(W(x^1, x^2, x^3, x^4)\big)$ требует полиномиального от размера исходной формулы времени. Следовательно, класс задач оптимального комбинирования 4 допустимых решений является NP-трудным. Теорема доказана.

3.8. NP-трудность задачи комбинирования с минимаксной сверткой критериев

Лемма 3.4. Оптимизационная задача (3.4), (3.5) NP-трудна:

$$x'_i \in \{0,1\}, i = \overline{1,n}, \tag{3.4}$$

$$\max\left(\sum_{i=1}^{n} a_i x'_i, \sum_{i=1}^{n} b_i(1 - x'_i)\right) \to \min. \tag{3.5}$$

Доказательство. Для доказательства данной леммы полиномиально сведем к задаче (3.4), (3.5) классическую NP-полную задачу разбиения [4]. Рассмотрим задачу разбиения с исходными параметрами $w_i, i = \overline{1,m}$. Определим $n = m, a_i = b_i = w_i, i = \overline{1,n}$. Оптимальное значение критерия задачи (3.4), (3.5) равно $\frac{1}{2}\sum_{i=1}^{m} w_i$ тогда и только тогда, когда задача разбиения имеет решение. Следовательно, задача (3.4), (3.5) является NP-трудной. Лемма доказана.

Теорема 3.10. Класс задач $Z_{\text{minmax}}\big(W(x^1, x^2)\big)$ NP-трудный.

Доказательство. Покажем полиномиальную сводимость NP-трудной задачи (3.4), (3.5) к классу задач $Z_{\text{minmax}}\big(W(x^1, x^2)\big)$.

Пусть $N = 2n$, $I = J = K = \{1, \dots, N\}$, $M = 2$. Трехиндексные матрицы стоимостей $c^1_{ijk}, c^2_{ijk}, i \in I, j \in J, k \in K$ определим следующим образом:

$$c^1_{ijk} = \begin{cases} a_q, \text{если } \exists q \in \{1, \dots, n\}, \text{что } i = j = k = 2q - 1, \\ 0 \text{ иначе}, \end{cases}, i \in I, j \in J, k \in K,$$

$$c^2_{ijk} = \begin{cases} b_q, \text{если } \exists q \in \{1, \dots, n\}, \text{что } i = j = k + 1 = 2q, \\ 0 \text{ иначе}, \end{cases}, i \in I, j \in J, k \in K.$$

Построим два подмножества $P_1, P_2 \subseteq I \times J \times K$, определяющих два допустимых решения системы (2.1)–(2.4):

$$P_1 = \{(i,i,i) | i = \overline{1,N}\},$$

$$P_2 = \{(2i-1, 2i-1, 2i), (2i, 2i, 2i-1) | i = \overline{1,n}\}.$$

Соответствующие два допустимых решения x^1, x^2 системы (2.1)–(2.4) определим как

$$x_{ijk}^{t} = \begin{cases} 1, \text{если } (i,j,k) \in P_t, \\ 0 \text{ иначе,} \end{cases} i \in I, j \in J, k \in K, t \in \{1,2\}.$$

Далее рассмотрим соответствующую задачу комбинирования $Z_{\text{minmax}}(W(x^1, x^2))$. Покажем, что оптимальное значение критерия задачи (3.4), (3.5) совпадает с оптимальным значением критерия задачи $Z_{\text{minmax}}(W(x^1, x^2))$.

1. Пусть x'^* – оптимальное решение задачи (3.5), (3.5), тогда построим $P(x'^*)$ следующим алгоритмом:

Шаг 1. $P(x'^*) = \emptyset$.

Шаг 2. Для каждого $i = \overline{1,n}$:

если $x_i'^* = 1$, тогда $P(x'^*) = P(x'^*) \cup \{(2i-1, 2i-1, 2i-1), (2i, 2i, 2i)\}$,

иначе $P(x'^*) = P(x'^*) \cup \{(2i-1, 2i-1, 2i), (2i, 2i, 2i-1)\}$.

Нетрудно увидеть, что значение критерия задачи $Z_{\text{minmax}}(W(x^1, x^2))$ на решении, соответствующем $P(x'^*)$, совпадёт с оптимальным значением критерия задачи (3.4), (3.5). Докажем это от противного. Предположим, что решение, соответствующее $P(x'^*)$, не является оптимальным в задаче $Z_{\text{minmax}}(W(x^1, x^2))$. Пусть x^*- оптимальное решение задачи $Z_{\text{minmax}}(W(x^1, x^2))$. Тогда значение критерия задачи $Z_{\text{minmax}}(W(x^1, x^2))$ на решении x^* строго меньше значения критерия задачи (3.4), (3.5) на решении x'^*. Тогда построим допустимое решение x' задачи (3.4), (3.5) следующим способом:

$$x'_i = \begin{cases} 1, \text{если } x_{iii}^* = 1, \\ 0 \text{ иначе,} \end{cases} i = \overline{1,n}.$$

По построению значение критерия задачи (3.4), (3.5) на решении x' совпадает со значением критерия задачи $Z_{\text{minmax}}(W(x^1, x^2))$ на решении x^*. Следовательно, x'^* не является оптимальным решением задачи (3.4), (3.5). Получили противоречие, предположение неверно. Отсюда значение критерия задачи $Z_{\text{minmax}}(W(x^1, x^2))$ на решении, соответствующем $P(x'^*)$, совпадёт с оптимальным значением критерия задачи (3.4), (3.5).

2. Пусть x^*- оптимальное решение задачи $Z_{\text{minmax}}(W(x^1, x^2))$, тогда построим оптимальное решение задачи (3.4), (3.5) следующим способом:

$$x'_i = \begin{cases} 1, \text{если } x^*_{iii} = 1, \\ 0 \text{ иначе,} \end{cases}, i = \overline{1,n}.$$

Нетрудно увидеть, что значение критерия задачи (3.4), (3.5) на решении x' совпадает с оптимальным критерием задачи $Z_{\text{minmax}}(W(x^1, x^2))$. Докажем от противного.

Предположим, что решение x' не является оптимумом в задаче (3.4), (3.5). Пусть x'^*- оптимальное решение задачи (3.4), (3.5). Тогда значение критерия задачи (3.4), (3.5) на решении x'^* строго меньше значения критерия задачи $Z_{\text{minmax}}(W(x^1, x^2))$ на решении x^*. Построим решение задачи $Z_{\text{minmax}}(W(x^1, x^2))$ следующим алгоритмом:

Шаг 1. $P(x'^*) = \emptyset$.

Шаг 2. Для каждого $i = \overline{1,n}$:

если $x'^*_i = 1$, тогда $P(x'^*) = P(x'^*) \cup \{(2i-1, 2i-1, 2i-1), (2i, 2i, 2i)\}$,

иначе $P(x'^*) = P(x'^*) \cup \{(2i-1, 2i-1, 2i), (2i, 2i, 2i-1)\}$.

По построению значения критерия задачи $Z_{\text{minmax}}(W(x^1, x^2))$ на решении, соответствующем $P(x'^*)$, совпадает со значением критерия задачи (3.4), (3.5) на решении x'^*. Следовательно, x^* не является оптимальным решением задачи $Z_{\text{minmax}}(W(x^1, x^2))$. Получили противоречие, предположение неверно. Отсюда значение критерия задачи (3.4), (3.5) на решении x' совпадёт с оптимальным значением критерия задачи $Z_{\text{minmax}}\big(W(x^1, x^2)\big)$.

Таким образом, задача (3.4), (3.5) полиномиально сводима к классу задач $Z_{\text{minmax}}(W(x^1, x^2))$. Следовательно, класс задач $Z_{\text{minmax}}(W(x^1, x^2))$ является NP-трудным. Теорема доказана.

3.9. NP-трудность задачи комбинирования многокритериальной задачи в случае линейной свертки критериев

Несложно увидеть, что задача $Z_L\left(W(x^1, x^2, \dots, x^r)\right)$ эквивалентна однокритериальной трехиндексной аксиальной задаче с матрицей стоимостей

$$c_{ijk} = \sum_{u \in \{1,\dots,M\}} a_u \mathrm{c}_{ijk}^u, i \in I, j \in J, k \in K.$$

Таким образом, согласно [54], задача $Z_L\left(W(x^1, x^2)\right)$ полиномиально разрешима. Алгоритм комбинирования, предложенный в [54], требует $O(n)$ вычислительных операций и может быть применен при решении задачи $Z_L\left(W(x^1, x^2)\right)$.

Несложно увидеть, что к классу задач $Z_L\left(W(x^1, x^2, x^3, x^4)\right)$ полиномиально сводим класс трехиндексных аксиальных задач с множеством разрешенных назначений вида $W(x^1, x^2, x^3, x^4)$, который, как было доказано в [56], является NP-трудным. Действительно, рассмотрим трехиндексную аксиальную задачу о назначениях с матрицей стоимостей $\mathrm{c}_{ijk}, i \in I, j \in J, k \in K$. При сведении в соответствующей задаче $Z_L\left(W(x^1, x^2, x^3, x^4)\right)$ определим $\mathrm{c}_{ijk}^1 = \mathrm{c}_{ijk}, \mathrm{c}_{ijk}^u = 0, a_1 = 1, a_u = 0, i \in I, j \in J, k \in K, u = \overline{2, M}$. Отсюда следует утверждение:

Утверждение 3.8. Класс задач $Z_L\left(W(x^1, x^2, \dots, x^r)\right)$ при $r \geq 4$ является NP-трудным.

Сложностной статус класса задач $Z_L\left(W(x^1, x^2, x^3)\right)$ на данный момент неизвестен.

3.10. NP-трудность задачи комбинирования многокритериальной задачи в случае лексикографической свертки критериев

Несложно увидеть, что к классу задач $Z_{\leqslant}\left(W(x^1, x^2, x^3, x^4)\right)$ полиномиально сводим класс трехиндексных аксиальных задач с множеством разрешенных назначений вида $W(x^1, x^2, x^3, x^4)$, который, как было доказано в [56], является NP-трудным. Действительно, рассмотрим трехиндексную аксиальную задачу о назначениях с матрицей стоимостей $\mathrm{c}_{ijk}, i \in I, j \in J, k \in K$. При сведении в соответствующей задаче $Z_{\leqslant}\left(W(x^1, x^2, x^3, x^4)\right)$ определим $\mathrm{c}^1_{ijk} = \mathrm{c}_{ijk}, \mathrm{c}^u_{ijk} = 0$, $i \in I, j \in J, k \in K, u = \overline{2, M}$. Отсюда следует утверждение:

Утверждение 3.9. Класс задач $Z_{\leqslant}\left(W(x^1, x^2, \ldots, x^r)\right)$ при $r \geq 4$ является NP-трудным.

Сложностной статус класса задач $Z_{\leqslant}\left(W(x^1, x^2, x^3)\right)$ на данный момент неизвестен.

3.11. Сложностной статус задач комбинирования

Общая информация о сложностном статусе задач комбинирования приведена в приложении 1 в таблице 1. Согласно результатам, для большинства постановок задач комбинирования в случае комбинирования трех решений сложностной статус задачи неизвестен, задача полиномиально разрешима в случае линейного и минимаксного критерия, линейной и лексикографической сверток, для построения подмножества Парето-оптимальных решений задачи комбинирования, а также для специального подкласса задач комбинирования с квадратичным критерием. Задачи комбинирования хотя бы четырех решений, а также задача комбинирования с минимаксной сверткой критериев являются NP-трудными.

В приложении 1 в таблице 2 представлена информация о разработанных алгоритмах для задач комбинирования. Для полиномиально разрешимых задач комбинирования предложены алгоритмы их точного решения. Для остальных классов предлагается использование разработанных эвристик.

4. Комплекс программ и вычислительный эксперимент

4.1. Описание комплекса программ

В целях проведения вычислительных экспериментов и эксплуатации разработанных алгоритмов решения задач комбинирования был разработан комплекс программ [61]. В рамках разработанного комплекса программ были проведены эксперимент с комбинированием решений аксиальной задачи о назначениях с декомпозиционной матрицей стоимостей, удовлетворяющей неравенству треугольника, полученных алгоритмом из статьи [19] на исходных данных предложенных в этой же статье, эксперимент с комбинированием решений аксиальной задачи о назначениях на данных сгенерированных по схеме из статьи [25], эксперимент с комбинированием решений аксиальной задачи о назначениях с минимаксным критерием, эксперимент с комбинированием решений аксиальной задачи о назначениях с квадратичным критерием, эксперимент с комбинированием решений многокритериальной задачи о назначениях в случае лексикографической свертки критериев, два эксперимента на различных исходных данных с построением аппроксимации Парето-области с использованием комбинирования решений аксиальной двухкритериальной задачи о назначениях. Свидетельство о государственной регистрации программы для ЭВМ представлено в приложении 4. Исходный код основных методов разработанных алгоритмов комбинирования представлен в приложении 5.

Комплекс программ разработан на языке программирования C, C++. В среде разработки Code::Blocks. Для функционирования программного комплекса используются стандартные заголовочные файлы C++: $iostream$, $cstdio$, $cstdlib$, $algorithm$, $cmath$, $vector$, set, map, $unordered\_set$, $unordered\_map$, $queue$, $ctime$, $cassert$, $complex$, $string$, $cstring$, $chrono$, $random$, $bitset$, $iomanip$, $list$, $stack$. Исходные данные задач могут сохраняться на диске в текстовом

формате. Результаты вычислений могут быть напечатаны на экране либо сохранены на диске в текстовом формате.

Функциональные возможности комплекса программ включают в себя:

- возможность сгенерировать входные данные для экспериментов выбрав вариант исходных матриц (трехиндексная матрица, две трехиндексных матрицы, три двухиндексных матрицы), выбрав способ расчета матриц стоимостей (случайный равномерно распределенный, расстояние между случайными равномерно распределенными точками на плоскости) и задав параметры распределения для выбора случайных величин,
- сохранить текущие матрицы стоимостей на диске,
- загрузить матрицы стоимостей с диска,
- сгенерировать случайные решения, задав количество желаемых решений,
- найти решения задачи методом $Spieksma$,
- применить процедуру локальной оптимизации, указав критерий останова,
- построить нижнюю оценку для текущей задачи о назначениях,
- применить выбранную стратегию для комбинации текущего множества решений,
- вывести значение текущего допустимого решения задачи,
- построить аппроксимацию Парето-кривой по текущему набору допустимых решений задачи.

Архитектура и логическая структура.

Комплекс программ состоит из четырех основных модулей: модуля взаимодействия с пользователем, модуля ввода и вывода, основного вычислительного модуля и модуля генерации исходных решений.

Модуль взаимодействия с пользователем обеспечивает принятие команд пользователя, взаимодействие с другими модулями для выполнения задач пользователя: запросы в модуль ввода и вывода для сохранения или загрузки матриц стоимостей или вывода значения критерия текущего допустимого решения задачи, запуск стратегий комбинирования из основного вычислительного модуля,

запрос в модуль генерации исходных решений для получения нового множества исходных решений.

Модуль ввода и вывода обеспечивает сохранение и загрузку матриц стоимостей, вывод результатов эксперимента. Хранение матриц осуществляется в текстовых файлах в заданных форматах для обеспечения возможности однозначной идентификации типа хранимых матриц и возможности визуального просмотра сгенерированных данных в матрицах стоимостей. Вывод результатов эксперимента может осуществляться на экран либо в текстовый файл, если пользователь задаст путь к файлу для сохранения результатов.

Основной вычислительный модуль реализует алгоритмы для проведения вычислительного эксперимента. Разработанные алгоритмы: алгоритм комбинирования двух допустимых решений трехиндексной аксиальной задачи о назначениях, алгоритм комбинирования двух допустимых решений трехиндексной аксиальной задачи о назначениях с минимаксным критерием, алгоритм комбинирования двух допустимых решений трехиндексной аксиальной задачи о назначениях с квадратичным критерием, алгоритм комбинирования двух допустимых решений двухкритериальной трехиндексной аксиальной задачи о назначениях в случае лексикографической свертки, алгоритм нахождения подмножества Парето-оптимальных решений задачи комбинирования двух допустимых решений двухкритериальной трехиндексной аксиальной задачи о назначениях. Разработанные стратегии комбинирования: стратегия комбинирования решений в случайном порядке, стратегия комбинирования решений в порядке неубывания значения критерия, стратегия комбинирования решений со случайными перестановками решений в порядке неубывания значения критерия, построение Парето-аппроксимации через комбинирование решений. Вспомогательные алгоритмы: алгоритм локальной оптимизации, алгоритм решения двухиндексной задачи о назначениях, алгоритм построения нижней оценки.

Модуль генерации исходных решений обеспечивает генерацию исходных решений и генерацию матриц стоимостей согласно запросу пользователя: может

быть сгенерированы одна или две трехиндексных матрицы или три двухиндексных матрицы, значения в матрице стоимостей может быть сгенерировано исходя из случайного равномерного распределения или быть рассчитано исходя из расстояния между случайными равномерно распределенными точками на плоскости, соответствующими индексам ячейки в матрице стоимостей, исходные решения могут быть сгенерированы алгоритмом случайного выбора троек, алгоритмом $Spieksma$ либо жадным алгоритмом.

4.2. Комбинирование решений аксиальной задачи о назначениях

В [19] предложен набор тестовых задач, состоящий из 18 задач, параметр n принимает значения 33 и 66. Предложенный в [19] подход основан на решении трех пар зависимых двухиндексных задач, в результате решения каждой из пары задач строится допустимое решение задачи (1.1)–(1.5). По аналогии с [19] значение критерия (1.5), полученное на данных трех допустимых решениях, будем обозначать через c_{IJ}, c_{IK}, c_{JK}, соответственно. Сами допустимые решения обозначим через x_{IJ}, x_{IK}, x_{JK}, соответственно. Далее последовательно применим Алгоритм 3.1 оптимального комбинирования решений следующим образом. Пусть x_{IJ-JK} является решением задачи $Z\left(W\left(x_{IJ}, x_{JK}\right)\right)$, $x_{IJ-JK-IK}$ является решением задачи $Z\left(W\left(x_{IJ-JK}, x_{IK}\right)\right)$. Значение критерия (1.5) на решении $x_{IJ-JK-IK}$ обозначим через $c_{IJ-JK-IK}$. Оптимальное значение критерия задачи (1.1)–(1.5) обозначим через c^*. Будем сравнивать рекорд среди решений x_{IJ}, x_{IK}, x_{JK} (соответствует $\min\{c_{IJ}, c_{IK}, c_{JK}\}$) с их оптимальной комбинацией $x_{IJ-JK-IK}$ (соответствует $c_{IJ-JK-IK}$). Полученные результаты приведены в приложении 2 в таблице 3.

Среднее отклонение от оптимума для рекорда среди решений x_{IJ}, x_{IK}, x_{JK} составляет 1,675%, среднее отклонение от оптимума для оптимальной комбинации данных решений $x_{IJ-JK-IK}$ составляет 1,527%. Фактическое снижение отклонения от оптимума за счет комбинирования решений произошло на 7 из 18 тестовых задач.

По аналогии с [25] построим тестовый набор с матрицами стоимостей, элементы которых сгенерированы с целочисленными значениями, равномерно распределенными в интервале [0,300]. Будем строить серии экспериментов с задачами размерности $n \in \{10,11,\dots,19\}$, в каждой серии построим $M = 10$ задач. Для каждой из тестовых задач случайным образом сгенерируем $N = n^3$ допустимых решений, к каждому из которых итеративно применим алгоритм локальной оптимизации, предложенный в [20], до тех пока решение не перестанет меняться. Полученные допустимые решения задачи (1.1)–(1.5) и соответствующие значения критерия (1.5) обозначим через x'_t и $c'_t, t = \overline{1,N}$. Тогда рекорд среди полученных решений обозначим $C' = \min\limits_{t=\overline{1,N}} c'_t$. Далее применим Алгоритм 3.1 оптимального комбинирования решений следующим образом. Пусть x''_1 является решением задачи $Z\left(W(x'_1, x'_2)\right)$, x''_t является решением задачи $Z\left(W(x''_{t-1}, x'_{t+1})\right), t = \overline{2,N-1}$. Соответствующие значения критерия обозначим $c''_t, t = \overline{1,N-1}$, и выберем рекорд, достигнутый комбинированием всей последовательности N решений $C'' = c''_{N-1}$. Наконец через C^* обозначим оптимальное значение критерия исходной задачи. Будем сравнивать отклонение от оптимума рекорда среди локально оптимизированных случайных решений и отклонение от оптимума рекорда среди локально оптимальных случайных решений последовательно комбинированных Алгоритмом 3.1. Для серии экспериментов будем оценивать среднее отклонение в серии. Полученные результаты приведены в приложении 2 в таблице 4. Согласно полученным результатам, среднее отклонение для C' по всем сериям составляет 46,671%, для C'' по всем сериям составляет 42,141%.

4.3. Комбинирование решений задачи о назначениях с минимаксным критерием

По аналогии с [25] построим тестовый набор с матрицами стоимостей, элементы которых сгенерированы с целочисленными значениями, равномерно распределенными в интервале [0,300]. Будем строить серии экспериментов с задачами размерности $n \in \{10,11,\dots,19\}$, в каждой серии построим $M = 10$ задач. Для каждой из тестовых задач случайным образом сгенерируем $N = n^4$ допустимых решений. Полученные допустимые решения задачи (1.1)–(1.4), (1.6) и соответствующие значения критерия (1.5) обозначим через x'_t и $c'_t, t = \overline{1,N}$. Тогда рекорд среди полученных решений обозначим $C' = \min\limits_{t=\overline{1,N}} c'_t$. Далее применим Алгоритм 3.1 оптимального комбинирования решений следующим образом. Пусть x''_1 является решением задачи $Z\big(W(x'_1, x'_2)\big)$, x''_t является решением задачи $Z\big(W(x''_{t-1}, x'_{t+1})\big), t = \overline{2,N-1}$. Соответствующие значения критерия обозначим $c''_t, t = \overline{1,N-1}$, и выберем рекорд, достигнутый комбинированием всей последовательности N решений $C'' = c''_{N-1}$. Будем сравнивать отклонение значения критерия решения, полученного при комбинировании, (C'') от значения критерия на решении, полученном при выборе рекорда (C'). Полученные результаты приведены в приложении 2 в таблице 5. Согласно полученным результатам, среднее отклонение от минимума составило $44{,}076\%$.

4.4. Комбинирование решений задачи о назначениях с квадратичным критерием

По аналогии с [25] построим тестовый набор с матрицами стоимостей, элементы которых сгенерированы с целочисленными значениями, равномерно распределенными в интервале [0,300]. Для каждого теста генерировалось $N = n^4$

случайных допустимых решений, каждое из которых проходило через процедуру локальной оптимизации. Для серии экспериментов будем оценивать среднее отклонение от минимума. В каждой серии было 10 задач одинаковой размерности. Полученные результаты приведены в приложении 2 в таблице 6. Согласно полученным результатам, применение алгоритма комбинации позволило улучшить значение критерия в среднем на 4,849% по сравнению с выбором минимума.

4.5. Комбинирование решений в случае лексикографической свертки критериев

Рассмотрим задачу $Z_{\preccurlyeq}$ при $M = 2$. По аналогии с [25] построим тестовый набор с трехиндексными матрицами стоимостей, элементы которых сгенерированы с целочисленными значениями, равномерно распределенными на отрезке [0,300]. Будем проводить серии экспериментов при фиксированных значениях n, количество задач в серии обозначим через K. Построим два эвристических алгоритма для решения задачи $Z_{\preccurlyeq}$. Первый алгоритм основан на построении подмножеств допустимых решений и выборе среди них рекорда, второй – аналогичен первому, но шаг выбора рекорда заменен на последовательное комбинирование решений при помощи Алгоритма 3.5. Отметим, что согласно Утверждению 3.1, Алгоритм 3.5 имеет сложность $O(n)$, что совпадает со сложностью выбора рекорда.

Эвристический алгоритм 1. Построим $N = n^3$ случайных решений, к каждому из которых применим алгоритм локальной оптимизации [20]. Полученные допустимые решения задачи $Z_{\preccurlyeq}$ обозначим через $x'_t, t = \overline{1,N}$. Тогда в качестве решения алгоритма выберем рекорд из полученных решений x^*: $x^* \preccurlyeq x'_t, t = \overline{1,N}$.

Эвристический алгоритм 2. Вместо шага выбора рекорда будем последовательно комбинировать пары решений при помощи Алгоритма 3.5 следующим образом. Пусть x''_1 является решением задачи $Z_{\preccurlyeq}(W(x'_1, x'_2))$. Далее

через $x"_t$ обозначим решением задачи $Z_{\leqslant}\left(W(x"_{t-1}, x'_{t+1})\right), t = \overline{2, N-1}$. В качестве решения алгоритма выберем $x"_{N-1}$.

Через $C^1(x)$, $C^2(x)$ обозначим значения критериев (2.5) на решении x. Будем сравнивать средние отклонение значений критериев при комбинировании (эристический алгоритм 2) от значений критериев при выборе рекорда (эвристический алгоритм 1) в сериях. Полученные результаты приведены в приложении 2 в таблице 7. Согласно полученным результатам, среднее отклонение для первого критерия по всем сериям составляет 3,25%, для второго по всем сериям составляет -0,02%. Что демонстрирует эффективность применения стратегии комбинировании вместо общепринятой стратегии выбора рекорда.

4.6. Комбинирование решений двухкритериальной задачи о назначениях

Предложенный алгоритм 3.6 гарантирует построение подмножества области Парето задачи $Z_2\left(W(x^1, x^2)\right)$. Далее применим данный алгоритм при построении эвристических методов аппроксимации Парето области исходной задачи Z_2. Эффективность работы эвристического алгоритма будем оценивать вычислительным экспериментом.

Построим матрицы c_{ijk}, d_{ijk} следующим способом: для каждого индекса $i \in I \cup J \cup K$ сгенерируем случайную точку p на плоскости XY так, что p_x, p_y целочисленные и равномерно распределенные на отрезке $[0, 2^{32} - 1]$, тогда $c_{ijk} = dist(i, j) + dist(j, k) + dist(i, k)$, где $dist(a, b)$ - Манхэттенское расстояние между точками a и b. Аналогичным образом определим d_{ijk}.

Определим процедуру локальной оптимизации для допустимого решения x задачи Z_2:

Шаг 1. Равновероятно выберем случайное число a $\in [0,1]$.

Шаг 2. Построим трехиндексную матрицу стоимостей $e_{ijk} = ac_{ijk} + (1-a)d_{ijk}$.

Шаг 3. Применим процедуру локальной оптимизации, предложенную в [20] к решению x исходной задачи, но с критерием $\sum_{i\in I}\sum_{j\in J}\sum_{k\in K} e_{ijk}x_{ijk} \to min$.

Построим n случайных допустимых решений $x_1, \dots, x_n$ задачи Z_2, к каждому из которых будем применять предложенную процедуру локальной оптимизации до тех пор, пока решение не перестанет меняться. Полученные решения обозначим $x'_1, \dots, x'_n$. Из полученных решений выберем недоминируемые. Обозначим построенную из этих решений аппроксимацию Парето-кривой как R.

Применим алгоритм 3.6 к парам решений, полученным после локальной оптимизации, для решения следующих задач $Z_2(W(x'_1, x'_2)), \dots, Z_2(W(x'_{n-1}, x'_n))$. Из всех полученных решений выберем недоминируемые. Обозначим построенную из этих решений аппроксимацию Парето-кривой как Q.

Будем сравнивать количество точек, которые недоминируются соответствующей аппроксимацией и не совпадают по значению критерия с какой-либо точкой этой аппроксимации. Введем обозначение количества точек, которые недоминируются аппроксимацией A следующим образом: $B(A) = |\{x | x - \text{допустимое решение задачи } Z_2, \nexists x' \in A, \text{ что } C(x') \le C(x) \text{ и } D(x') \le D(x)\}|$.
Для каждого теста $B(R), B(Q)$ считались путем проверки (полным перебором) всего множества допустимых решений на соответствие условию несуществования доминирующего или равного по значению критериев решения в соответствующей аппроксимации.

Тесты проводились на размерностях $n = 6,7,8$, для каждой размерности было проведено 50 тестов. В приложении 2 в таблице 8 приведены результаты первого вычислительного эксперимента по построению аппроксимации Парето-области. Согласно им, в среднем применение Алгоритма 3.6 позволило уменьшить количество недоминируемых точек на 6,57%.

Построим матрицы c_{ijk}, d_{ijk} следующим способом: для каждого индекса $i \in I \cup J \cup K$ сгенерируем случайную точку p на плоскости XY так, что p_x, p_y

целочисленные и равномерно распределенные на отрезке $[0, 2^{32} - 1]$, тогда $c_{ijk} = dist(i,j) + dist(j,k) + dist(i,k), d_{ijk} = -\max(dist(i,j), dist(j,k), dist(i,k))$, где $dist(a,b)$ – Манхэттенское расстояние между точками a и b.

Построим n^3 случайных решений $x_1, \dots, x_{n^3}$ задачи Z_2, к каждому из которых применим один шаг предложенной выше процедуры локальной оптимизации. Полученные решения обозначим $x'_1, x'_2, \dots, x'_{n^3}$. Из полученных решений выберем недоминируемые. Обозначим построенную из этих решений аппроксимацию Парето-кривой как R.

Применим алгоритм 3.6 к парам решениям, полученным после локальной оптимизации, для решения следующих задач $Z_2(W(x'_1, x'_2)), \dots, Z_2(W(x'_{n^3-1}, x'_{n^3}))$. Из всех полученных решений выберем недоминируемые. Обозначим построенную из этих решений аппроксимацию Парето-кривой как Q.

Будем сравнивать количество точек, которые недоминируются соответствующей аппроксимацией и не совпадают по значению критерия с какой-либо точкой этой аппроксимации.

Тесты проводились на размерностях $n = 6,7,8$, было проведено 50 тестов. В приложении 2 в таблице 9 приведены результаты второго вычислительного эксперимента по построению аппроксимации Парето-области. Согласно им, в среднем применение алгоритма 3.6 позволило уменьшить количество недоминируемых точек на 13,74%.

Заключение

Основным результатом работы являются построенная математическая модель комбинирования допустимых решений трехиндексной аксиальной задачи о назначениях и предложенный подход для постобработки допустимых решений трехиндексной аксиальной задачи о назначениях, основанный на комбинировании решений. Исходные решения могут быть получены любыми эвристическими или приближенными алгоритмами.

Разработаны полиномиальные точные алгоритмы комбинирования двух допустимых решений трехиндексной аксиальной задачи о назначениях, задачи о назначениях с минимаксным критерием, а также многокритериальной задачи о назначениях в случаях линейной и лексикографической сверток критериев.

Разработан эвристический алгоритм комбинирования двух допустимых решений задачи о назначениях с квадратичным критерием и найден подкласс задач, на котором алгоритм находит точное решение.

Разработан полиномиальный алгоритм нахождения подмножества Парето-оптимальных решений задачи комбинирования двух допустимых решений двухкритериальной трехиндексной аксиальной задачи о назначениях.

Показана NP-трудность задачи комбинирования четырех допустимых решений трехиндексной аксиальной задачи о назначениях, многокритериальной задачи о назначениях в случае лексикографической и линейной сверток, а также NP-трудность задачи комбинирования многокритериальной задачи о назначениях с минимаксной сверткой критериев.

Предложенный подход применен при разработке программного модуля для моделирования выполнения производственного плана при различных графиках работы производства, входящий в состав АИС «Ока-план». Акт внедрения представлен в приложении 3.

Разработан программный комплекс, реализующий разработанные алгоритмы, с помощью которого был проведен вычислительный эксперимент, и

который можно использовать для решения рассмотренных задач о назначениях. Приведены результаты вычислительных экспериментов, демонстрирующие повышение качества исходных решений в результате применения разработанных алгоритмов. Свидетельство о государственной регистрации программы для ЭВМ представлено в приложении 4. Исходный код основных методов разработанных алгоритмов комбинирования представлен в приложении 5.

Полученные результаты обобщаются при исследовании многоиндексных задач о назначении.

Дальнейшими возможными направлениями для исследований могут быть исследование вопроса построения эффективного алгоритма комбинирования трех допустимых решений трехиндексной задачи о назначениях, а также исследование возможности применения метода комбинирования допустимых решений в иных задачах.

Список литературы

1. Spieksma F.C.R. Multi Index Assignment Problems. Complexity, Approximation, Applications / P.M. Pardalos, L.S. Pitsoulis (Eds.). Nonlinear Assignment Problems: Algorithms and Applications. Dordrecht: Kluwer Acad. Publishers, 2000. P. 1–11.
2. Афраймович Л.Г. Эвристический метод решения целочисленных декомпозиционных многоиндексных задач // АиТ. 2014. № 8. С. 3–18.

Afraimovich L.G. A Heuristic Method for Solving Integer-Valued Decompositional Multiindex Problems // Autom. Remote Control. 2014. V. 75. No. 8. P. 1357–1368.

3. Афраймович Л.Г., Прилуцкий М. Х. Многоиндексные задачи оптимального планирования производства // АиТ. 2010. № 10. С. 148–155.

Afraimovich L.G., Prilutskii M.Kh. Multiindex Optimal Production Planning Problems // Autom. Remote Control. 2010. V. 71. No. 10. P. 2145–2151.

4. Poore A.B., Multidimensional assignment problems arising in multitarget and multisensor tracking. In: Nonlinear Assignment Problems. Springer, Boston 2000. P. 13–38.
5. Poore A.B., Lu S., Suchomel B.J., Data association using multiple-frame assignments. In: Handbook of Multisensor Data Fusion. CRC Press, Boca Raton. 2017. P. 319–338.
6. S. Vadrevu and R. Nagi, A dual-ascent algorithm for the multi-dimensional assignment problem: application to multiTarget tracking. In: Proceedings of the 2020 IEEE 23rd International Conference on Information Fusion (FUSION). IEEE (2020) 1–8.
7. Y. Zhang, Y. Li, S. Li, J. Zeng, Y. Wang and S. Yan, Multi-target tracking in multi-static networks with autonomous underwater vehicles using a robust multi-sensor labeled Bernoulli filter. J. Mar. Sci. Eng. 11 (2023) 875.

8. A.R. Kammerdiner, A. Semenov and E.L. Pasiliao, Multidimensional assignment problem for multipartite entity resolution. J. Glob. Optim. (2022) 1–33.
9. E. Balas and P.R. Landweer, Traffic assignment in communication satellites. Oper. Res. Lett. 2 (1983) 141–147.
10. C.J. Veenman, M.J.T. Reinders and E. Backer, Establishing motion correspondence using extended temporal scope. Artif. Intell. 145 (2003) 227–243.
11. E.R. Wulan, D. Jamaluddin and W.N. Ramadhan, Determining optimal solutions in learning outcome using one to one fixed method. In: Proceedings of the International Conference on Science and Engineering (ICSE-UIN-SUKA 2021). Atlantis Press (2021) 7–11
12. L.E.U. Rivero, M.R.B. Escarcega and J. Velasco, An integer linear programming model for a case study in classroom assignment problem. Comput. Sist. 24 (2020) 97–104
13. A.M. Frieze and J. Yadegar, An algorithm for solving 3-dimensional assignment problems with application to scheduling a teaching practice. J. Oper. Res. Soc. 32 (1981) 989–995.
14. Zhuang Y., Zhou Y., Hassini E., Yuan Y., Hu X. Rack retrieval and repositioning optimization problem in robotic mobile fulfillment systems // Transportation Research Part E: Logistics and Transportation Review, 2022. V. 167. P. 102920.
15. Gabrovšek B., Novak T., Povh J., Rupnik Poklukar D., Žerovnik J. Multiple Hungarian Method for k-Assignment Problem // Mathematics. 2020. V. 8. 2050.
16. John M., Schmitz C., Park A.Y., Dixon N.E., Huber T., Otting G. Sequence-specific and stereospecific assignment of methyl groups using paramagnetic lanthanides // J. Am. Chem. Soc. 2007. V. 129. P. 13749–13757.
17. Dokka T., Spieksma F.C.R. Facets of the axial three-index assignment polytope // Discrete Appl. Math. 2016. V. 201. P. 86–104.

18. Гэри М., Джонсон Д. Вычислительные машины и труднорешаемые задачи. М.: Мир, 1982.
19. Crama Y., Spieksma F.C.R. Approximation Algorithms for Three-Dimensional Assignment Problems with Triangle Inequalities // Eur. J. Oper. Res. 1992. V. 60. P. 273–279.
20. Huang G., Lim A. A Hybrid Genetic Algorithm for the Three-Index Assignment Problem // Eur. J. Oper. Res. 2006. V. 172. P. 249–257.
21. Karapetyan D., Gutin D. A New Approach to Population Sizing for Memetic Algorithms: A Case Study for the Multidimensional Assignment Problem // Evolutionary Computation. 2011. V. 19. No. 3. P. 345–371.
22. Медведев С.Н., Медведева О.А. Адаптивный алгоритм решения аксиальной трехиндексной задачи о назначениях // АиТ. 2019. № 4. С. 156–172.

Medvedev S.N., Medvedeva O.A. An Adaptive Algorithm for Solving the Axial Three-Index Assignment Problem // Autom. Remote Control. 2019. V. 80. No. 4. P. 718–732.

23. Гимади Э.Х., Коркишко Н.М. Об одном алгоритме решения трехиндексной аксиальной задачи о назначениях на одноциклических подстановках // Дискретный анализ и исследование операций. Сер. 1. 2003. Т. 10. № 2. С. 56–65.
24. Mehbali M. Solution approaches to the Three-Index Assignment Problem // RAIRO - Operations Research. 2025. V. 59. P. 1035–1065.
25. Balas E., Saltzman M.J. An Algorithm for the Three-Index Assignment Problem // Oper. Res. 1991. V. 39. No. 1. P. 150–161.
26. Bandelt H.J., Crama Y., Spieksma F.C.R. Approximation algorithms for multidimensional assignment problems with decomposable costs // Discret. Appl. Math. 1994. V.49. P. 25–50.
27. Burkard R.E., Rudolf R., Woeginger G.J. Three-dimensional axial assignment problems with decomposable cost coefficients // Discrete Appl. Math. 1996. V. 65. P. 123–139.

28. Walteros J., Vogiatzis C., Pasiliao E., Pardalos P. Integer programming models for the multidimensional assignment problem with star costs // Eur. J. Oper. Res. 2014. V. 235. P. 553–568.
29. Spieksma F., Woeginger G. Geometric three-dimensional assignment problems // European Journal of Operational Research. 1996. V. 91. P. 611-618.
30. Афраймович Л.Г. Многоиндексные транспортные задачи с декомпозиционной структурой // АиТ. 2012. № 1. С. 130–147.

Afraimovich L.G. A Multi-index Transport Problems with Decomposition Structure // Autom. Remote Control. 2012. V. 73. No. 1. P. 118–133.

31. Афраймович Л.Г. Многоиндексные транспортные задачи с 2-вложенной структурой // АиТ. 2013. № 1. С. 116–134.

Afraimovich L.G. A Multiindex Transportation Problems with 2-embedded Structure // Autom. Remote Control. 2013. V. 74. No. 1. P. 90–104.

32. Афраймович Л.Г. Трехиндексные задачи линейного программирования с вложенной структурой // АиТ. 2011. № 8. С. 109–120.

Afraimovich L.G. Three-Index Linear Programs with Nested Structure // Autom. Remote Control. 2011. V. 72. No. 8. P. 1679–1689.

33. Kim B.J., Hightower W.L., Hahn P.M., Zhu Y.R., Sun L. Lower bounds for the axial three-index assignment problem // European Journal of Operational Research. 2010. V. 202. P. 654-668.
34. Дичковская С.А., Кравцов М.К. Исследование полиномиальных алгоритмов решения трехиндексной планарной проблемы выбора // Журнал вычислительной математики математической физики. 2006. Т. 46. 2. С. 222–228.
35. Думбадзе Л.Г., Леонов В.Ю., Тизик А.П., Цурков В.И. Декомпозиционный метод решения трехиндексной задачи о назначениях // Известия Российской академии наук. Теория и системы управления, № 5. 2020. С. 56-59.

36. Гимади Э.Х., Глазков Ю.В. Об асимптотически точном алгоритме решения одной модификации трехиндексной планарной задачи о назначениях // Дискретный анализ и исследование операций. Сер. 2. 2006. Т. 13. 1. С. 10–26.
37. Magos D. Tabu search for the planar three-index assignment problem // Journal of Global Optimization. 1996. V. 8. P. 35–48.
38. Прилуцкий М.Х. Программные управления двухстадийными стохастическими производственными системами // АиТ. 2020. № 1. С. 81–92.

Prilutskii M.Kh. Programmed control of two-stage stochastic production systems // Autom. Remote Control. 2020. V. 81. P. 64–73.

39. Прилуцкий М.Х. Оптимальное управление двухстадийными стохастическими производственными системами // АиТ. 2018. № 5. С. 69–82.

Prilutskii M.Kh. Optimal control for two-stage stochastic production systems // Autom. Remote Control. 2018. V. 79. P. 830–840.

40. Прилуцкий М.Х. Оптимальное планирование двухстадийных стохастических производственных систем // АиТ. 2014. № 8. С. 37–47.

Prilutskii M.Kh. Optimal planning for two-stage stochastic industrial systems // Autom. Remote Control. 2014. V. 75. P. 1384–1392.

41. Дичковская С.А., Кравцов М.К. Исследование полиномиальных алгоритмов решения многокритериальной трехиндексной планарной задачи о назначениях // Журн. вычислит. мат. и мат. физики. 2007. Т. 47. 6. С. 1077–1086.
42. Емеличев В.А., Перепелица В.А. Сложность дискретных многокритериальных задач // Дискретная математика. 1994. Т. 6. Вып. 1. С. 3-33.
43. Прилуцкий М.Х. Многокритериальные многоиндексные задачи объемнокалендарного планирования // Известия РАН. Теория и системы управления. 2007. № 1. С. 78–82.

44. He G., Liu J., Zhao C. Approximation algorithms for some graph partitioning problems // Graph Algorithms and Applications 2; World Scientific: Singapore, 2004; pp. 21–31.
45. Kravtsov V.M. Polynomial algorithms for finding the asymptotically optimum plan of the multiindex axial assignment problem // Cybernetics and Systems Analysis. 2005. V. 41. P. 940–944.
46. Zhang L., Sidoti D., Vallabhaneni S., Pattipati K.R., Castnon D.A. Approaches to obtain a large number of ranked solutions to 3-dimensional assignment problems // Journal of Advances in Information Fusion. 2018. V. 13. P. 50–67.
47. Kammerdiner A.R., Vaughan. C. F. Very large-scale neighborhood search for the multidimensional assignment problem // Optimization Methods and Applications. 2017. V. 130. P. 251-262.
48. Kammerdiner A.R., Semenov A., Pasiliao E. Landscape properties of the very large-scale and the variable neighborhood search metaheuristics for the multidimensional assignment problem // Journal of Global Optimization. 2024. V. 88. P. 653–683.
49. Li J., Tharmarasa R., Brown D., Kirubarajan T., Pattipati K.R. A novel convex dual approach to three-dimensional assignment problem: theoretical analysis // Computational Optimization and Applications. 2019. V. 74. P. 481–516.
50. Прилуцкий М.Х. Многокритериальное распределение однородного ресурса в иерархических системах // Автоматика и Телемеханика. 1996. № 2. С. 24–29.
51. Lin C.-J., Ma K. T. Model and algorithms of the fuzzy three-dimensional axial assignment problem with an additional constraint // The South African Journal of Industrial Engineering. 2015. V. 26. N 3. P. 54–70.
52. Spieksma F. C. R. The approximability of three-dimensional assignment problems with bottleneck objective // Optimization Letters. 2009. V. 4. P. 7–16.
53. Garey M.R., Johnson D.S. Computers and Intractability: A Guide to the Theory of NP-Completeness. San Francisco, 1979. 338 p.

Список работ, опубликованных автором по теме диссертации

54. Афраймович Л.Г., Емелин М.Д. Комбинирование решений аксиальной задачи о назначениях // АиТ. 2021. № 8. С. 159-168.

Afraimovich L.G., Emelin M.D. Combining solutions of the axial assignment problem //Autom. Remote Control, 2021. V. 82. N 8. P. 1418-1425.

55. Афраймович Л.Г., Емелин М.Д. Эвристические стратегии комбинирования решений трехиндексной аксиальной задачи о назначениях //АиТ. 2021. № 10. С. 6–12.

Afraimovich L.G., Emelin M.D. Heuristic Strategies for Combining Solutions of the Three-Index Axial Assignment Problem // Autom. Remote Control, 2021. V. 82, N. 10. 1635–1640.

56. Afraimovich L.G., Emelin M.D. Complexity of Solutions Combination for the Three-Index Axial Assignment Problem // Mathematics. 2022. V. 10. 1062.

57. Афраймович Л. Г., Емелин М. Д., Решение задачи оптимального комбинирования двух допустимых решений трёхиндексной аксиальной задачи о назначениях, Тринадцатый международный семинар «Дискретная математика и её приложения», 2019, С. 193-196

58. Афраймович Л.Г., Емелин М. Д., Стратегии комбинирования решений трехиндексной задачи о назначениях, Интеллектуализация обработки информации: Тезисы докладов 13-й Международной конференции, г. Москва 2020 г. — М.: Российская академия наук, 2020. С. 378-380.

59. Афраймович Л.Г., Емелин М. Д., Свертки критериев при комбинировании решений многокритериальной аксиальной задачи о назначениях // АиТ. 2024. № 8. С. 86–98.

60. Афраймович Л.Г., Емелин М. Д., Построение области Парето при комбинировании допустимых решений многокритериальной аксиальной задачи о назначениях // АиТ. 2025. № 5. С. 81–97.

61. Свидетельство о государственной регистрации программы для ЭВМ № 2026661204 Российская Федерация. Решение задачи о назначениях с применением комбинирования решений: № 2026660570 : заявл. 10.04.2026 : опубл. 17.04.2026, Бюл. № 4 / М. Д. Емелин; правообладатель федеральное государственное автономное образовательное учреждение высшего образования «Национальный исследовательский Нижегородский государственный университет им. Н.И. Лобачевского».

Приложения

Приложение 1

	Аксиальная трехиндексная задача о назначениях			Многокритериальная аксиальная трехиндексная задача о назначениях			
	Линейный критерий	Квадратичный критерий	Минимаксный критерий	Линейная свертка	Минимаксная свертка	Лексикографическая свертка	Парето-область
m=2	O(n)	O(n) в случае $A_l^p \geq 0$, $l = \overline{1,q}, p \in \{1,2\}$, $A_l^1 \leq A_l^2$ и $C_l^1 \leq C_l^2$ или $A_l^1 \geq A_l^2$ и $C_l^1 \geq C_l^2$, $l = \overline{1,q}$ В общем случае – открытая проблема	O(n)	O(n)	NP-трудна	O(n)	В двух критериальном случае построение подмножества – O(n) В общем случае – открытая проблема
m=3	Открытая проблема	Открытая проблема	Открытая проблема	Открытая проблема		Открытая проблема	Открытая проблема
m>=4	NP-трудна			NP-трудна		NP-трудна	

Таблица 1 – Сводная таблица сложностного статуса задач комбинирования.

	Аксиальная трехиндексная задача о назначениях			Многокритериальная аксиальная трехиндексная задача о назначениях			
	Линейный критерий	Квадратичный критерий	Минимаксный критерий	Линейная свертка	Минимаксная свертка	Лексикографическая свертка	Парето-область
m=2	Точный алгоритм	Точный алгоритм случае $A_l^p \geq 0$, l $= \overline{1,q}$, $p \in \{1,2\}$, $A_l^1 \leq A_l^2$ и $C_l^1 \leq C_l^2$ или $A_l^1 \geq A_l^2$ и $C_l^1 \geq C_l^2$, $l = \overline{1,q}$. В общем случае – эвристика	Точный алгоритм	Точный алгоритм	Эвристика	Точный алгоритм	Эвристика
m≥3	Эвристика	Эвристика	Эвристика	Эвристика		Эвристика	

Таблица 2 – сводная таблица разработанных алгоритмов комбинирования.

Приложение 2

	n	c^*	c_{IJ}	c_{IK}	c_{JK}	$\min\{c_{IJ}, c_{IK}, c_{JK}\}$	$c_{IJ-JK-IK}$
1	33	1608	1620	1637	1655	1620	1615
2	33	1401	1420	1416	1411	1411	1406
3	33	1604	1613	1635	1632	1613	1604
4	66	2662	2678	2684	2666	2666	2663
5	66	2449	2503	2486	2470	2470	2466
6	66	2758	2811	2788	2792	2788	2777
7	33	4797	4855	4851	4885	4851	4851
8	33	5067	5165	5150	5181	5150	5150
9	33	4287	4341	4364	4371	4341	4341
10	66	9684	9817	9761	9891	9761	9761
11	66	8944	9138	9177	9129	9129	9129
12	66	9745	9921	9869	9975	9869	9869
13	33	133	139	135	136	135	135
14	33	131	137	138	137	137	136
15	33	131	134	136	136	134	134
16	66	286	296	296	295	295	295
17	66	286	295	295	292	292	292
18	66	282	294	297	294	294	294

Таблица 3 – Результаты эксперимента на тестовой базе Spieksma.

n	M	$100\% \cdot \frac{C' - C^*}{C^*}$	$100\% \cdot \frac{C'' - C^*}{C^*}$
10	10	5,366%	5,366%
11	10	8,435%	6,021%
12	10	13,913%	12,704%
13	10	20,686%	15,913%
14	10	33,339%	29,448%
15	10	55,093%	50,347%
16	10	73,054%	69,869%
17	10	83,103%	75,433%
18	10	75,947%	67,858%
19	10	97,771%	88,445%

Таблица 4 – результаты эксперимента по комбинированию решений.

n	Среднее отклонение по группе. $100\% \cdot \frac{C"-C'}{C'}$
10	30,541%
11	36,889%
12	36,675%
13	46,213%
14	46,268%
15	43,111%
16	45,372%
17	53,884%
18	47,176%
19	54,627%

Таблица 5 – результаты эксперимента по комбинированию решений в задаче с минимаксным критерием.

n	Среднее отклонение от минимума
10	5,13%
11	5,223%
12	4,904%
13	5,932%
14	6,339%
15	4,415%
16	4,497%
17	4,267%
18	3,552%
19	4,231%

Таблица 6 – результаты эксперимента по комбинированию решений в задаче с квадратичным критерием.

n	K	$100\% \cdot \frac{C^1(x^*) - C^1(x"_{N-1})}{C^1(x^*)}$	$100\% \cdot \frac{C^2(x^*) - C^2(x"_{N-1})}{C^2(x^*)}$
10	10	3,09%	1,98%
11	10	5,3%	-6,23%
12	10	5,31%	6,58%
13	10	2,05%	0,46%
14	10	0%	0%
15	10	1,26%	-1,85%
16	10	2,5%	2,14%
17	10	2,74%	-4,92%
18	10	3,19%	-0,7%
19	10	7,08%	2,3%

Таблица 7 – результаты эксперимента по комбинированию решений в задаче с лексикографической сверткой критериев.

Размерность задачи	$\frac{B(R)-B(Q)}{B(R)}100\%$
6	2,14%
7	8,03%
8	9,54%

Таблица 8 – первый эксперимент по построению аппроксимации Парето-области.

Размерность задачи	$\frac{B(R)-B(Q)}{B(R)}100\%$
6	11,44%
7	18,08%
8	11,71%

Таблица 9 – второй эксперимент по построению аппроксимации Парето-области.

Приложение 3

УТВЕРЖДАЮ
Первый заместитель директора РФЯЦ-ВНИИЭФ -
директор филиала РФЯЦ-ВНИИЭФ
«НИИИС им. Ю.Е. Седакова», к.т.н.
_____ В.С. Васильев
«26» 11 2025 г.

АКТ

внедрения в филиале РФЯЦ-ВНИИЭФ «НИИИС им. Ю.Е. Седакова» результатов диссертации Емелина М.Д. «Комбинирование решений аксиальной задачи о назначениях», представленной на соискание ученой степени кандидата физико-математических наук по специальности 1.2.2. - Математическое моделирование, численные методы и комплексы программ

Комиссия, созданная приказом первого заместителя директора РФЯЦ-ВНИИЭФ - директора филиала РФЯЦ-ВНИИЭФ «НИИИС им. Ю.Е. Седакова» от 19.11.2025 № 195-95/1815-П в составе:

председатель: Власов В.С., заместитель директора филиала РФЯЦ-ВНИИЭФ «НИИИС им. Ю.Е. Седакова» по информационным технологиям - начальник научно-исследовательского отделения систем жизненного цикла изделий микроэлектроники и РЭА, к.т.н.;

члены комиссии: Сазонов А.А., начальник производственно-технологического отдела, к.х.н.;
Жилин А.В., начальник научно-исследовательского отдела, к.ф.-м.н.;
Балашов В.В., начальник научно-исследовательского отдела, к.т.н.;
Ковтун К.А., начальник группы производственно-технологического сопровождения

рассмотрев диссертацию Емелина М.Д., констатирует, что ее основные результаты использованы в филиале РФЯЦ-ВНИИЭФ «НИИИС им. Ю.Е. Седакова» в ОКР по разработке программных комплексов для цифровой трансформации процессов проектирования-изготовления изделий микроэлектроники и радиоэлектронной аппаратуры, а также в научных исследованиях при создании новых подходов к решению задач объемно-календарного планирования процесса изготовления изделий микроэлектроники. В частности, при разработке автоматизированной информационной системы обеспечения календарного планирования процессов изготовления изделий микроэлектроники (АИС «Ока-план»),

входящей в состав комплекса систем информационной поддержки процессов изготовления изделий микроэлектроники (Комплекс «Кремний»), предложенный автором подход к комбинированию решений трехиндексной задачи транспортного типа - аксиальной задачи о назначениях позволил формализовать решение задачи оптимального объемно-календарного планирования процесса изготовления изделий микроэлектроники и получить (совместно с ННГУ им. Н.И.Лобачевского) следующие результаты:

1. Разработан программный модуль для моделирования выполнения производственного плана при различных графиках работы производства, входящий в состав АИС «Ока-план».
2. Разработан программный модуль для моделирования выполнения производственного плана при условиях ввода и/или вывода оборудования из производственного процесса и выставления приоритетов для отдельных партий, входящий в состав АИС «Ока-план».
3. Разработан программный модуль для моделирования выполнения производственного плана с целью формирования портфеля заказов с учетом уже запущенных партий, а также выполнения ситуационного анализа при долговременном планировании производственных процессов в части определения партий с нарушением срока изготовления изделий микроэлектроники, входящий в состав АИС «Ока-план».

АИС «Ока-план», включающая описанные программные модули, внедрена в постоянную эксплуатацию в филиале РФЯЦ-ВНИИЭФ «НИИИС им. Ю.Е.Седакова» при производстве изделий микроэлектроники.

Экономический эффект от внедрения результатов кандидатской диссертацию Емелина М.Д. заключается в снижении затрат на проведение планирования работ по изготовлению изделий микроэлектроники на величину до 5%.

Председатель комиссии: В.С. Власов

Члены комиссии: А.А. Сазонов

А.В. Жилин

В.В. Балашов

К.А. Ковтун

Приложение 4

РОССИЙСКАЯ ФЕДЕРАЦИЯ

СВИДЕТЕЛЬСТВО

о государственной регистрации программы для ЭВМ

№ 2026661204

«Решение задачи о назначениях с применением комбинирования решений»

Правообладатель: ***федеральное государственное автономное образовательное учреждение высшего образования «Национальный исследовательский Нижегородский государственный университет им. Н.И. Лобачевского» (RU)***

Автор(ы): ***Емелин Максим Денисович (RU)***

Заявка № **2026660570**

Дата поступления **10 апреля 2026 г.**

Дата государственной регистрации в Реестре программ для ЭВМ ***17 апреля 2026 г.***

Руководитель Федеральной службы по интеллектуальной собственности

Ю.С. Зубов

Приложение 5

```
    vector<bool> used;
    vector<vector<ll> > graph;
    vector<ll> component;
    ll q=0;
void dfs(int x)
{
    if (x<n)
    {
        component.push_back(x);
    }
    used[x]=1;
    for(size_t i=0;i<graph[x].size();i++)
    {
        ll nex=graph[x][i];
        if (used[nex]) continue;
        dfs(nex);
    }
}
void fill_graph(vector<pair<pair<ll,ll>,ll> > &best_ans,
                vector<pair<pair<ll,ll>,ll> > &ans)
{
    graph.resize(3*n);
    ll i;
    for(i=0;i<n;i++)
    {
        graph[ans[i].first.first].push_back(ans[i].first.second+n);
        graph[ans[i].first.second+n].push_back(ans[i].first.first);
        graph[ans[i].first.first].push_back(ans[i].second+2*n);
        graph[ans[i].second+2*n].push_back(ans[i].first.first);
        graph[ans[i].second+2*n].push_back(ans[i].first.second+n);
        graph[ans[i].first.second+n].push_back(ans[i].second+2*n);

        graph[best_ans[i].first.first].push_back(best_ans[i].first.second+n);
        graph[best_ans[i].first.second+n].push_back(best_ans[i].first.first);
        graph[best_ans[i].first.first].push_back(best_ans[i].second+2*n);
        graph[best_ans[i].second+2*n].push_back(best_ans[i].first.first);
        graph[best_ans[i].second+2*n].push_back(best_ans[i].first.second+n);
        graph[best_ans[i].first.second+n].push_back(best_ans[i].second+2*n);
    }
    for(i=0;i<3*n;i++)
    {
        sort(graph[i].begin(),graph[i].end());
    }
}
void fill_graph(vector<vector<pair<pair<ll,ll>,ll> > > &best_ans)
{
    graph.resize(3*n);
    ll i;
    for(auto &ans:best_ans)
    for(i=0;i<n;i++)
    {
        graph[ans[i].first.first].push_back(ans[i].first.second+n);
        graph[ans[i].first.second+n].push_back(ans[i].first.first);
        graph[ans[i].first.first].push_back(ans[i].second+2*n);
        graph[ans[i].second+2*n].push_back(ans[i].first.first);
        graph[ans[i].second+2*n].push_back(ans[i].first.second+n);
        graph[ans[i].first.second+n].push_back(ans[i].second+2*n);
    }
    for(i=0;i<3*n;i++)
    {
        sort(graph[i].begin(),graph[i].end());
    }
}
```

```
void combine(vector<pair<pair<ll,ll>,ll> > &best_ans,
             vector<pair<pair<ll,ll>,ll> > &ans)
{
    used.assign(3*n,0);
    ll i;
    fill_graph(best_ans,ans);
    vector<pair<pair<ll,ll>,ll> > new_ans(n);
    for(i=0;i<n;i++)
    {
        if (best_ans[i].first.first!=i||ans[i].first.first!=i) assert(0);
        if (used[i]) continue;
        dfs(i);
        ll ans1=0,ans2=0;
        for(auto j:component)
        {
            ans1+=A[j][best_ans[j].first.second][best_ans[j].second];
            ans2+=A[j][ans[j].first.second][ans[j].second];
        }
        if (ans1<=ans2)
        {
            for(auto j:component)
            {
                new_ans[j]=best_ans[j];
            }
        }
        else
        {
            for(auto j:component)
            {
                new_ans[j]=ans[j];
            }
        }
        component.clear();
    }
    for(i=0;i<3*n;i++) graph[i].clear();
    graph.clear();
    best_ans=new_ans;
}
void combine_mima(vector<pair<pair<ll,ll>,ll> > &best_ans,
             vector<pair<pair<ll,ll>,ll> > &ans)
{
    used.assign(3*n,0);
    ll i;
    fill_graph(best_ans,ans);
    vector<pair<pair<ll,ll>,ll> > new_ans(n);
    for(i=0;i<n;i++)
    {
        if (best_ans[i].first.first!=i||ans[i].first.first!=i) assert(0);
        if (used[i]) continue;
        dfs(i);
        ll ans1=0,ans2=0;
        for(auto j:component)
        {
            ans1=max(ans1,A[j][best_ans[j].first.second][best_ans[j].second]);
            ans2=max(ans2,A[j][ans[j].first.second][ans[j].second]);
        }
        if (ans1<=ans2)
        {
            for(auto j:component)
            {
                new_ans[j]=best_ans[j];
            }
        }
        else
        {
            for(auto j:component)
            {
                new_ans[j]=ans[j];
            }
        }
```

```
        component.clear();
    }
    for(i=0;i<3*n;i++) graph[i].clear();
    graph.clear();
    best_ans=new_ans;
}
vector<pair<pair<ll,ll>,ll> > rand_comb(vector<vector<pair<pair<ll,ll>,ll> > > v,int a=0){
    ll i;
    vector<ll> p(v.size(),0);
    for(i=0;i<p.size();i++) p[i]=i;
    random_shuffle(p.begin()+a,p.end());
    vector<pair<pair<ll,ll>,ll> > ans=v[p[0]];
    for(i=1;i<v.size();i++)
    {
        combine(ans,v[p[i]]);
    }
    return ans;
}
namespace _sort_comb
{
    vector<ll> v;
}
bool sort_comb_down(ll x,ll y)
{
    if (_sort_comb::v[x]==_sort_comb::v[y]) return x<y;
    return _sort_comb::v[x]>_sort_comb::v[y];
}
bool sort_comb_up(ll x,ll y)
{
    if (_sort_comb::v[x]==_sort_comb::v[y]) return x<y;
    return _sort_comb::v[x]<_sort_comb::v[y];
}
vector<pair<pair<ll,ll>,ll> > sort_comb(vector<vector<pair<pair<ll,ll>,ll> > > v,int st=0,int
pr=0){
    ll i,j;
    vector<ll> p(v.size(),0);
    for(i=0;i<p.size();i++) p[i]=i;
    _sort_comb::v.resize(v.size());
    switch(st)
    {
    case 1:
        for(i=0;i<v.size();i++) _sort_comb::v[i]=ll_ans(v[i]);
        sort(p.begin(),p.end(),sort_comb_down);
        break;
    case 2:
        for(i=1;i<v.size();i++)
        {
            vector<pair<pair<ll,ll>,ll> > temp=v[0];
            combine(temp,v[i]);
            _sort_comb::v[i]=ll_ans(temp);
        }
        sort(p.begin()+1,p.end(),sort_comb_up);
        break;
    case 3:
        for(i=0;i<v.size();i++) _sort_comb::v[i]=ll_ans(v[i]);
        sort(p.begin()+1,p.end(),sort_comb_up);
        break;
    case 13:
    {
        random_shuffle(p.begin(),p.end());
        int m=v.size()*pr/100;
        vector<int> ran;
        set<int> sran;
        for(i=0;i<m;i++)
        {
            ran.push_back(p[i]);
            sran.insert(p[i]);
        }
        for(i=0;i<v.size();i++) _sort_comb::v[i]=ll_ans(v[i]);
        sort(p.begin(),p.end(),sort_comb_up);
```

```
        j=0;
        for(i=0;i<n;i++)
        {
            if (sran.count(p[i]))
            {
                p[i]=ran[j];
                j++;
            }
        }
        break;
    }
    case 0:
        for(i=0;i<v.size();i++) _sort_comb::v[i]=ll_ans(v[i]);
        sort(p.begin(),p.end(),sort_comb_up);
        break;
    }
    vector<pair<pair<ll,ll>,ll> > ans=v[p[0]];
    for(i=1;i<v.size();i++)
    {
        combine(ans,v[p[i]]);
    }
    return ans;
}
vector<pair<pair<ll,ll>,ll> > min_comb(vector<vector<pair<pair<ll,ll>,ll> > > v)
{
    ll i;
    vector<pair<pair<ll,ll>,ll> > ans=v[0];
    for(i=1;i<v.size();i++)
    {
        if (ll_ans(ans)>ll_ans(v[i]))
        {
            ans=v[i];
        }
    }
    return ans;
}
vector<pair<pair<ll,ll>,ll> > comb_nn(vector<vector<pair<pair<ll,ll>,ll> > > v)
{
    ll i;
    while(v.size()>1)
    {
        for(i=0;i+1<v.size();i++)
        {
            combine(v[i],v[i+1]);
        }
        v.pop_back();
    }
    return v[0];
}
vector<pair<pair<ll,ll>,ll> > comb_nn2(vector<vector<pair<pair<ll,ll>,ll> > > v)
{
    ll i,j=0;
    vector<pair<pair<ll,ll>,ll> > ans=v[0];
    for(i=1;i<v.size();i++)
    {
        if (ll_ans(ans)>ll_ans(v[i]))
        {
            ans=v[i];
            j=i;
        }
    }
    swap(v[j],v[v.size()-1]);
    v.pop_back();
    while(v.size())
    {
        vector<pair<pair<ll,ll>,ll> > tans=ans;
        combine(tans,v[0]);
        j=0;
        for(i=1;i<v.size();i++)
        {
```

```
            vector<pair<pair<ll,ll>,ll> > b=ans;
            combine(b,v[i]);
            if (ll_ans(tans)>ll_ans(b))
            {
                tans=b;
                j=i;
            }
        }
        swap(v[j],v[v.size()-1]);
        v.pop_back();
        ans=tans;
    }
    return ans;
}
vector<pair<pair<ll,ll>,ll> > comb_all_por(vector<vector<pair<pair<ll,ll>,ll> > > v)
{
    ll i;
    vector<ll> p(v.size(),0);
    for(i=0;i<p.size();i++) p[i]=i;
    vector<pair<pair<ll,ll>,ll> > ans=v[0];
    do
    {
        vector<pair<pair<ll,ll>,ll> > tans=v[p[0]];
        for(ll i=1;i<v.size();i++)
        {
            combine(tans,v[p[i]]);
        }
        if (ll_ans(tans)<ll_ans(ans))
        {
            ans=tans;
        }
    }while(next_permutation(p.begin(),p.end()));
    return ans;
}
void front_add(vector<pair<ll,ll>> &pareto,pair<ll,ll> p)
{
    vector<bool> del(pareto.size(),0);
    bool bad=0;
    for(size_t i=0;i<pareto.size();i++)
    {
        if (p.first<=pareto[i].first&&p.second<=pareto[i].second)
        {
            del[i]=1;
        }
        if (p.first>=pareto[i].first&&p.second>=pareto[i].second)
        {
            bad=1;
        }
    }
    if (bad) return;
    int j=0;
    for(size_t i=0;i<pareto.size();i++)
    {
        if (del[i]) continue;
        pareto[j++]=pareto[i];
    }
    pareto.resize(j+1);
    while(j&&pareto[j-1].first>p.first)
    {
        pareto[j]=pareto[j-1];
        j--;
    }
    pareto[j]=p;
}
void generate_pareto(int t)
{
    string path="tests/pareto_curve_"+to_string(n)+"_"+to_string(t)+".txt";
    freopen(path.c_str(),"w",stdout);
    vector<pair<ll,ll>> pareto;
    vector<int> p(n),q(n);
```

```
    for(int i=0;i<n;i++) p[i]=q[i]=i;
    do
    {
        do
        {
            vector<pair<pair<ll,ll>,ll>> x;
            for(int i=0;i<n;i++)
            {
                x.push_back({{i,p[i]},q[i]});
            }
            front_add(pareto,two_ll_ans(x));
        }while(next_permutation(q.begin(),q.end()));
    }while(next_permutation(p.begin(),p.end()));
    for(auto j:pareto)
    {
        cout<<j.first<<' '<<j.second<<'\n';
    }
    freopen("CON","w",stdout);
    cout.clear();
}
void generate_approximate_pareto(int t)
{
    string path="tests/pareto_curve_aprx_"+to_string(n)+"_"+to_string(t)+".txt";
    freopen(path.c_str(),"w",stdout);
    vector<pair<ll,ll>> pareto;
    vector<int> p(n);
    for(int i=0;i<n;i++) p[i]=i;
    do
    {
        solve2ind(A,A2,x1);
        solve2ind(A2,A,x2);
        comb_two(x1,x2,v);
        for(auto j:v)
        {
            front_add(pareto,two_ll_ans(j));
        }
    }while(next_permutation(p.begin(),p.end()));
    for(auto j:pareto)
    {
        cout<<j.first<<' '<<j.second<<'\n';
    }
    freopen("CON","w",stdout);
    cout.clear();
}
void read_pareto(int t,vector<pair<ll,ll>> &pareto)
{
    string path="tests/pareto_curve_"+to_string(n)+"_"+to_string(t)+".txt";
    freopen(path.c_str(),"r",stdin);
    pair<ll,ll> p;
    while(cin>>p.first>>p.second)
    {
        pareto.push_back(p);
    }
    freopen("CON","r",stdin);
    cin.clear();
}
void    combine_two_criteria(vector<pair<pair<ll,ll>,ll>    >&x,vector<pair<pair<ll,ll>,ll>
>&y,vector<vector<pair<pair<ll,ll>,ll>>>& v)
{
    used.assign(3*n,0);
    ll i;
    fill_graph(x,y);
    vector<pair<pair<ll,ll>,ll>> fixed;
    vector<pair<vector<pair<pair<ll,ll>,ll>>,vector<pair<pair<ll,ll>,ll>>>> choose;
    for(i=0;i<n;i++)
    {
        if (x[i].first.first!=i||y[i].first.first!=i) assert(0);
        if (used[i]) continue;
        dfs(i);
        if (component.size()==1)
```

```
        {
            fixed.push_back(x[i]);
        }
        else
        {

choose.push_back(pair<vector<pair<pair<ll,ll>,ll>>,vector<pair<pair<ll,ll>,ll>>>());
            for(auto j:component)
            {
                choose.back().first.push_back(x[j]);
                choose.back().second.push_back(y[j]);
            }
        }
        component.clear();
    }
    for(i=0;i<3*n;i++) graph[i].clear();
    graph.clear();
    for(i=0;i<(1<<choose.size());i++)
    {
        v.push_back(vector<pair<pair<ll,ll>,ll>>());
        for(int j=0;j<(int)choose.size();j++)
        {
            if ((1<<j)&i) for(auto q:choose[j].first) v.back().push_back(q);
            else for(auto q:choose[j].second) v.back().push_back(q);
        }
        for(auto q:fixed) v.back().push_back(q);
        sort(v.back().begin(),v.back().end());
        check_correct(v.back());
    }
}
void optimal_curve(vector<vector<pair<pair<ll,ll>,ll>>>& v)
{
    if (!v.size()) return;
    vector<pair<ll,ll>> vals(v.size());
    for(int i=0;i<(int)vals.size();i++)
    {
        vals[i]=two_ll_ans(v[i]);
    }
    vector<ll> p(v.size());
    for(int i=0;i<v.size();i++) p[i]=i;
    sort(p.begin(),p.end(),[&](ll x,ll y){return vals[x]==vals[y]?x<y:vals[x]<vals[y];});
    vector<ll> ans;
    vector<vector<pair<pair<ll,ll>,ll>>> ans2;
    ans.push_back(p[0]);
    for(int i=0;i<v.size();i++)
    {
        if (vals[p[i]].second>=vals[ans.back()].second) continue;
        ans.push_back(p[i]);
    }
    for(auto j:ans)
    {
        ans2.push_back(v[j]);
    }
    v=ans2;
}
void pareto_curve(vector<vector<pair<pair<ll,ll>,ll>>>& v)
{
    vector<ll> p(n);
    vector<ll> q(n);
    for(int i=0;i<n;i++) p[i]=q[i]=i;
    do
    {
        do
        {
            v.push_back(vector<pair<pair<ll,ll>,ll>>(n));
            for(int i=0;i<n;i++)
            {
                v.back()[i]={{i,p[i]},q[i]};
            }
        }while(next_permutation(q.begin(),q.end()));
```

```
    }while(next_permutation(p.begin(),p.end()));
    optimal_curve(v);
}
void          comb_two(vector<pair<pair<ll,ll>,ll>>          &x,vector<pair<pair<ll,ll>,ll>>
&y,vector<vector<pair<pair<ll,ll>,ll>>> &v)
{
    used.assign(3*n,0);
    ll i;
    fill_graph(x,y);
    vector<pair<pair<ll,ll>,ll>> fixed;
    vector<pair<vector<pair<pair<ll,ll>,ll>>,vector<pair<pair<ll,ll>,ll>>>> choose;
    vector<pair<ld,ll>> ord;
    auto c=[&](pair<pair<ll,ll>,ll> &x)
    {
        return A[x.first.first][x.first.second][x.second];
    };
    auto d=[&](pair<pair<ll,ll>,ll> &x)
    {
        return A2[x.first.first][x.first.second][x.second];
    };
    for(i=0;i<n;i++)
    {
        if (x[i].first.first!=i||y[i].first.first!=i) assert(0);
        if (used[i]) continue;
        dfs(i);
        if (component.size()==1)
        {
            fixed.push_back(x[i]);
        }
        else
        {
            ll c1=0,c2=0,d1=0,d2=0;
            for(auto j:component)
            {
                c1+=c(x[j]);
                c2+=c(y[j]);
                d1+=d(x[j]);
                d2+=d(y[j]);
            }
            if (c2>c1&&d2<d1)
            {
                ord.push_back({(d1-d2)/(ld)(c2-c1),choose.size()});

choose.push_back(pair<vector<pair<pair<ll,ll>,ll>>,vector<pair<pair<ll,ll>,ll>>>());
                for(auto j:component)
                {
                    choose.back().first.push_back(x[j]);
                    choose.back().second.push_back(y[j]);
                }
            }
            else if (c1>c2&&d1<d2)
            {
                ord.push_back({(d2-d1)/(ld)(c1-c2),choose.size()});

choose.push_back(pair<vector<pair<pair<ll,ll>,ll>>,vector<pair<pair<ll,ll>,ll>>>());
                for(auto j:component)
                {
                    choose.back().first.push_back(y[j]);
                    choose.back().second.push_back(x[j]);
                }
            }
            else if (c1<c2||c1==c2&&d1<=d2)
            {
                for(auto j:component)
                {
                    fixed.push_back(x[j]);
                }
            }
            else
            {
```

```
                for(auto j:component)
                {
                    fixed.push_back(y[j]);
                }
            }
        }
        component.clear();
    }
    for(i=0;i<3*n;i++) graph[i].clear();
    graph.clear();
    sort(ord.begin(),ord.end());
    reverse(ord.begin(),ord.end());
    v.push_back(vector<pair<pair<ll,ll>,ll>>());
    for(int j=0;j<(int)choose.size();j++)
    {
        for(auto q:choose[j].first) v.back().push_back(q);
    }
    for(auto q:fixed) v.back().push_back(q);
    sort(v.back().begin(),v.back().end());
    check_correct(v.back());
    for(int i=0;i<choose.size();i++)
    {
        v.push_back(vector<pair<pair<ll,ll>,ll>>());
        for(int j=0;j<(int)choose.size();j++)
        {
            if (j>i) for(auto q:choose[j].first) v.back().push_back(q);
            else for(auto q:choose[j].second) v.back().push_back(q);
        }
        for(auto q:fixed) v.back().push_back(q);
        sort(v.back().begin(),v.back().end());
        check_correct(v.back());
    }
}
void local_optima(vector<pair<pair<ll,ll>,ll> > &ans,double a=1.0)
{
    ll i,j;
    vector<pair<pair<ll,ll>,ll> > tans(n);
    vector<ll> p;
    vector<vector<ld> > temp_matr(n);
    for(i=0;i<n;i++)
    {
        temp_matr[i].resize(n);
    }
    vector<vector<vector<ld>>> A3(n,vector<vector<ld>>(n,vector<ld>(n,0)));
    for(int i=0;i<n;i++)
    {
        for(int j=0;j<n;j++)
        {
            for(int k=0;k<n;k++)
            {
                A3[i][j][k]=A[i][j][k]*a+A2[i][j][k]*(1-a);
            }
        }
    }
    auto ll_ans=[&A3](vector<pair<pair<ll,ll>,ll>> x)
    {
        ld ans=0;
        for(auto j:x)
        {
            ans+=A3[j.first.first][j.first.second][j.second];
        }
        return ans;
    };
    bool d=1;
    while(d)
    {
        d=0;
        for(i=0;i<n;i++)
        {
            for(j=0;j<n;j++)
```

```
    {
        temp_matr[i][j]=A3[ans[i].first.first][ans[i].first.second][j];
    }
}
parsoch(temp_matr,p);
for(i=1;i<n+1;i++)
{
    j=i-1;
    tans[p[i]-1]={ans[p[i]-1].first,j};
}
for(i=0;i<n;i++)
{
    if (tans[i].first.first!=i) assert(0);
}
if (ll_ans(ans)>ll_ans(tans))
{
    d=1;
}
if (ll_ans(ans)<ll_ans(tans))
{
    assert(0);
}
ans=tans;
for(i=0;i<n;i++)
{
    for(j=0;j<n;j++)
    {
        temp_matr[i][j]=A3[ans[i].first.first][j][ans[i].second];
    }
}
parsoch(temp_matr,p);
for(i=1;i<n+1;i++)
{
    j=i-1;
    tans[p[i]-1]={{ans[p[i]-1].first.first,j},ans[p[i]-1].second};
}
for(i=0;i<n;i++)
{
    if (tans[i].first.first!=i) assert(0);
}
if (ll_ans(ans)>ll_ans(tans))
{
    d=1;
}
if (ll_ans(ans)<ll_ans(tans))
{
    assert(0);
}
ans=tans;
for(i=0;i<n;i++)
{
    for(j=0;j<n;j++)
    {
        temp_matr[i][j]=A3[i][ans[j].first.second][ans[j].second];
    }
}
parsoch(temp_matr,p);
for(i=1;i<n+1;i++)
{
    j=i-1;
    tans[p[i]-1]={{p[i]-1,ans[j].first.second},ans[j].second};
}
for(i=0;i<n;i++)
{
    if (tans[i].first.first!=i) assert(0);
}
if (ll_ans(ans)>ll_ans(tans))
{
    d=1;
}
```

```
        if (ll_ans(ans)<ll_ans(tans))
        {
            assert(0);
        }
        ans=tans;
        break;
    }
}
void local_optima2(vector<pair<pair<ll,ll>,ll> > &ans)
{
    bool d=1;
    while(d)
    {
        d=0;
        for(int i=0;i<n;i++)
        {
            for(int j=i+1;j<n;j++)
            {
                vector<pair<pair<ll,ll>,ll> > tans=ans;
                swap(tans[i].second,tans[j].second);
                if (ll_ans2(tans)<ll_ans2(ans))
                {
                    d=1;
                    ans=tans;
                }
                swap(tans[i].first.second,tans[j].first.second);
                if (ll_ans2(tans)<ll_ans2(ans))
                {
                    d=1;
                    ans=tans;
                }
                swap(tans[i].second,tans[j].second);
                if (ll_ans2(tans)<ll_ans2(ans))
                {
                    d=1;
                    ans=tans;
                }
            }
        }
    }
}
void    local_optima_leks(vector<pair<pair<ll,ll>,ll>    >    &ans,vector<vector<vector<ll>>>
&A,vector<vector<vector<ll>>> &A2)
{
    bool d=1;
    auto ll_ans_leks=[&](vector<pair<pair<ll,ll>,ll> > &x)
    {
        ll tans=0;
        for(ll i=0;i<n;i++)
        {
            tans+=A[ans[i].first.first][ans[i].first.second][ans[i].second];
        }
        ll d=0;
        for(ll i=0;i<n;i++)
        {
            d+=A2[ans[i].first.first][ans[i].first.second][ans[i].second];
        }
        return make_pair(tans,d);
    };
    while(d)
    {
        d=0;
        for(int i=0;i<n;i++)
        {
            for(int j=i+1;j<n;j++)
            {
                vector<pair<pair<ll,ll>,ll> > tans=ans;
                swap(tans[i].second,tans[j].second);
                if (ll_ans_leks(tans)<ll_ans_leks(ans))
                {
```

```
                    d=1;
                    ans=tans;
                }
                swap(tans[i].first.second,tans[j].first.second);
                if (ll_ans_leks(tans)<ll_ans_leks(ans))
                {
                    d=1;
                    ans=tans;
                }
                swap(tans[i].second,tans[j].second);
                if (ll_ans_leks(tans)<ll_ans_leks(ans))
                {
                    d=1;
                    ans=tans;
                }
            }
        }
    }
}
void local_optima_mima(vector<pair<pair<ll,ll>,ll> > &ans,double time=0)
{
    bool d=1;
    while(d)
    {
        d=0;
        for(int i=0;i<n;i++)
        {
            for(int j=i+1;j<n;j++)
            {
                vector<pair<pair<ll,ll>,ll> > tans=ans;
                swap(tans[i].second,tans[j].second);
                if (ll_ans_mima(tans)<ll_ans_mima(ans))
                {
                    d=1;
                    ans=tans;
                }
                swap(tans[i].first.second,tans[j].first.second);
                if (ll_ans_mima(tans)<ll_ans_mima(ans))
                {
                    d=1;
                    ans=tans;
                }
                swap(tans[i].second,tans[j].second);
                if (ll_ans_mima(tans)<ll_ans_mima(ans))
                {
                    d=1;
                    ans=tans;
                }
            }
        }
    }
}
void lower_bound(ld up,vector<pair<pair<ll,ll>,ll> > &ans)
{
    ans.resize(n);
    ll i,j,k;
    C.resize(n+1);
    for(i=0;i<n+1;i++)
    {
        C[i].resize(n+1);
    }
    vector<ld> u(n,0);
    vector<vector<ld> > temp_matr;
    vector<vector<ll> > dir;
    vector<ll> check(2*n,(ll)1e18);
    do
    {
        temp_matr.resize(n);
        dir.resize(n);
        for(i=0;i<n;i++)
```

```
        {
            temp_matr[i].resize(n);
            dir[i].resize(n);
            for(j=0;j<n;j++)
            {
                temp_matr[i][j]=200;
                for(k=0;k<n;k++)
                {
                    if (temp_matr[i][j]>A[i][j][k]-u[i])
                    {
                        temp_matr[i][j]=A[i][j][k]-u[i];
                        dir[i][j]=k;
                    }
                }
            }
        }
        copy_matr(temp_matr);
        parsoch(n,n,C,p);
        ll down=0;
        vector<ll> m(n,1);
        ll norm=0;
        for(i=1;i<n+1;i++)
        {
            down+=temp_matr[p[i]-1][i-1]+u[i-1];
            m[dir[p[i]-1][i-1]]--;
        }
        for(i=0;i<n;i++)
        {
            norm+=m[i]*m[i];
        }
        if (norm==0)
        {
            break;
        }
        ld lam,temp=down/up;
        if (temp>=0.95)
        {
            lam=0.5;
        }
        else
        {
            if (temp<0.9)
            {
                lam=1;
            }
            else
            {
                lam=0.75;
            }
        }
        ld t=(up-down)/norm*lam;
        for(i=0;i<n;i++)
        {
            u[i]+=t*m[i];
        }
        if (t<1e-8)
        {
            break;
        }
        for(i=0;i<2*n-1;i++)
        {
            check[i]=check[i+1];
        }
        check[2*n-1]=down;
    }while((check[2*n-1]-check[0])>1e-8);
    for(i=1;i<n+1;i++)
    {
        ans[p[i]-1]={{p[i]-1,i-1},dir[p[i]-1][i-1]};
    }
}
```