Изображение выглядит как текст, письмо, чернила, документ

Автоматически созданное описание

Изображение выглядит как текст, письмо, Шрифт, чернила

Автоматически созданное описание

Изображение выглядит как текст, рукописный текст, диаграмма, черно-белый

Автоматически созданное описание

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 5](#_Toc167068760)

[1 МЕТОД ПАРЕТО 7](#_Toc167068761)

[1.1 Введение 7](#_Toc167068762)

[1.2 Выбор Парето-оптимального множества 7](#_Toc167068763)

[1.3 Указание верхних/нижних границ критериев 10](#_Toc167068764)

[1.4 Субоптимизация 11](#_Toc167068765)

[1.5 Лексикографическая оптимизация 11](#_Toc167068766)

[1.6 Результаты работы программы 12](#_Toc167068767)

[1.7 Заключение 13](#_Toc167068768)

[2 МЕТОД ЭЛЕКТРА II 14](#_Toc167068769)

[2.1 Введение 14](#_Toc167068770)

[2.2 Выбор лучшего варианта 14](#_Toc167068771)

[2.3 Веса предпочтений 16](#_Toc167068772)

[2.4 Вывод 28](#_Toc167068773)

[2.5 Результат работы программы 29](#_Toc167068774)

[2.6 Заключение 29](#_Toc167068775)

[3 МЕТОД АНАЛИЗА ИЕРАРХИЙ 30](#_Toc167068776)

[3.1 Введение 30](#_Toc167068777)

[3.2 Постановка задачи 30](#_Toc167068778)

[3.3 Представление проблемы в виде иерархии 30](#_Toc167068779)

[3.4 Установка приоритетов критериев 32](#_Toc167068780)

[3.5 Синтез приоритетов 32](#_Toc167068781)

[3.6 Согласованность локальных приоритетов 40](#_Toc167068782)

[3.7 Синтез альтернатив 47](#_Toc167068783)

[3.8 Вывод 48](#_Toc167068784)

[3.9 Результаты работы программы 48](#_Toc167068785)

[3.10 Заключение 50](#_Toc167068786)

[4 ГРАФИЧЕСКИЙ МЕТОД 51](#_Toc167068787)

[4.1 Введение 51](#_Toc167068788)

[4.2 Постановка задачи 51](#_Toc167068789)

[4.3 Данные индивидуального варианта 51](#_Toc167068790)

[4.4 Подготовка данных 52](#_Toc167068791)

[4.5 Построение графика 52](#_Toc167068792)

[4.6 Выделение области допустимых решений 53](#_Toc167068793)

[4.7 Максимум функции 54](#_Toc167068794)

[4.8 Минимум функции 55](#_Toc167068795)

[4.9 Заключение 57](#_Toc167068796)

[5 СИМПЛЕКСНЫЙ МЕТОД 58](#_Toc167068797)

[5.1 Введение 58](#_Toc167068798)

[5.2 Постановка задачи 58](#_Toc167068799)

[5.3 Математическая модель задачи 59](#_Toc167068800)

[5.4 Решение задачи 59](#_Toc167068801)

[5.5 Пример работы программы 66](#_Toc167068802)

[5.6 Заключение 66](#_Toc167068803)

[6 ДВОЙСТВЕННАЯ ЗАДАЧА 68](#_Toc167068804)

[6.1 Введение 68](#_Toc167068805)

[6.2 Постановка задачи 68](#_Toc167068806)

[6.3 Математическая модель 69](#_Toc167068807)

[6.4 Соответствующая исходной двойственная задача 69](#_Toc167068808)

[6.5 Первая теорема двойственности 70](#_Toc167068809)

[6.6 Вторая теорема двойственности 73](#_Toc167068810)

[6.7 Третья теорема двойственности 74](#_Toc167068811)

[6.8 Результаты работы программы 78](#_Toc167068812)

[6.9 Заключение 78](#_Toc167068813)

[7 ТРАНСПОРТНАЯ ЗАДАЧА 79](#_Toc167068814)

[7.1 Введение 79](#_Toc167068815)

[7.2 Постановка задачи 79](#_Toc167068816)

[7.3 Математическая модель транспортной задачи 80](#_Toc167068817)

[7.4 Метод северо-западного угла 80](#_Toc167068818)

[7.5 Метод минимальной стоимости 81](#_Toc167068819)

[7.6 Метод потенциалов 82](#_Toc167068820)

[7.7 Результаты выполнения программы 89](#_Toc167068821)

[7.8 Заключение 93](#_Toc167068822)

[ЗАКЛЮЧЕНИЕ 94](#_Toc167068823)

[СПИСОК ИНФОРМАЦИОННЫХ ИСТОЧНИКОВ 95](#_Toc167068824)

[ПРИЛОЖЕНИЯ 96](#_Toc167068825)

ВВЕДЕНИЕ

Управление основывается на определенных решениях, которые необходимы для достижения цели. Важнейшим признаком таких решений является его непосредственная направленность их на организацию коллективной деятельности. Такие решения принято называть управленческими решениями.

Управленческое решение – важнейший вид управленческого труда, а также совокупность взаимосвязанных, целенаправленных и логически последовательных управленческих действий, которые обеспечивают реализацию управленческих задач;

Субъектом управленческого решения является лицо, принимающее решение (ЛПР), которое наделено полномочиями и несет ответственность за реализацию управленческого решения. ЛПР может быть представлено как одним человеком, так и группой людей (коллективом). Соответственно относительно признака численности ЛПР выделяют следующие виды управленческих решений – индивидуальные и коллективные.

Решения могут делиться на два типа:

* бинарное - определено двумя диаметрально противоположными альтернативами, которые вынуждают к выбору типа «да/нет»;
* многокритериальное - имеется выбор из некоторого конечного числа возможных альтернатив.

Математическое программирование является одним из разделов исследования операций – прикладного направления кибернетики, используемого для решения практических организационных задач. Задачи математического программирования находят применение в различных областях человеческой деятельности, где необходим выбор одного из возможных образов действий (программ действий).

Традиционно в математическом программировании выделяют следующие основные разделы.

Линейное программирование – целевая функция линейна, а множество, на котором ищется экстремум целевой функции, задается системой линейных равенств и неравенств. В свою очередь в линейном программировании существуют классы задач, структура которых позволяет создать специальные методы их решения, выгодно отличающиеся от методов решения задач общего характера. Так, в линейном программировании появился раздел транспортных задач.

Нелинейное программирование – целевая функция и ограничения нелинейны.

Сущность линейного программирования состоит в нахождении точек наибольшего или наименьшего значения некоторой функции при определенном наборе ограничений, налагаемых на аргументы и образующих систему ограничений, которая имеет, как правило, бесконечное множество решений.

Математическая модель любой задачи линейного программирования включает в себя:

* максимум или минимум целевой функции (критерий оптимальности);
* систему ограничений в форме линейных уравнений и неравенств;
* требование неотрицательности переменных.

1 метод парето

1.1 Введение

Цель работы: познакомиться с методом Парето и применить его для нахождения оптимальной альтернативы в заданной предметной области.

Предметная область: выбор оптимального высшего учебного заведения.

Метод Парето применяется в задачах многокритериальной оптимизации, то есть если альтернативы нужно сравнивать по двум и более критериям.

Суть метода Парето заключается в прямом сравнении альтернатив между собой. Сравнение производится по критериям, причём по каким-то критериям требуется максимизация, а по каким-то – минимизация. Если одна альтернатива лучше другой по всем критериям, то она называется доминирующей, а другая альтернатива называется доминируемой. Если есть критерии, по которым одна альтернатива лучше другой, и есть критерии, по которой она хуже, то эти альтернативы являются несравнимыми.

В Парето-оптимальное множество входят только те альтернативы, которые не хуже всех других альтернатив, т.е. которые не является доминируемыми по сравнению с любой другой альтернативой.

Для сужения получаемого множества оптимальных альтернатив существуют метод указания верхних/нижних границ; субоптимизация и лексикографическая оптимизация).

1.2 Выбор Парето-оптимального множества

Задача: выбрать оптимальное техническое высшее учебное заведение для поступления на направление «программная инженерия». Рассмотренные критерии выбора:

* Проходной балл на бюджет, взятый за 2023 год;
* Количество бюджетных мест, взятое за 2023 год;
* Минимальная стоимость обучения за год на 2023 год;
* Размер государственной академической стипендии для студентов, сдавших сессию на «хорошо» и «отлично» на 2023 год;
* Национальный рейтинг университета на основе информации с сайта academia.interfax.ru;
* Расстояние до общежития (километры).

В таблице 1.1.1 приведены десять альтернатив и шесть критериев для выбора оптимального технического высшего учебного заведения.

Таблица 1.1.1 – Альтернативы и критерии

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| № | Варианты  решений | Критерии | | | | | |
| Проходной балл (+) | Количество бюджетных мест (-) | Стоимость обучения (руб.) (+) | Размер стипендии (руб.) (+) | Рейтинг университета (баллы) (+) | Рассто-  яние до обще-жития (км) (-) |
| 1 | РТУ МИРЭА | 276 | 285 | 320000 | 1800 | 843 | 13,7 |
| 2 | МГТУ имени Н.Э. Баумана | 291 | 84 | 363040 | 1854 | 964 | 4,2 |
| 3 | ВШЭ | 295 | 135 | 700000 | 1994 | 878 | 7,9 |
| 4 | МАИ | 247 | 25 | 277770 | 1765 | 761 | 12,4 |
| 5 | ИТМО | 304 | 45 | 349000 | 2252 | 838 | 3,4 |
| 6 | СПбГУ | 276 | 35 | 335300 | 1843 | 915 | 7,3 |
| 7 | МТУСИ | 263 | 60 | 290000 | 1799 | 705 | 8,4 |
| 8 | СГУ им. Чернышев-ского | 192 | 30 | 42960 | 1454 | 847 | 12,3 |
| 9 | НИЯУ МИФИ | 300 | 37 | 287000 | 1875 | 975 | 15 |
| 10 | МФТИ | 290 | 30 | 432000 | 1900 | 965 | 4,8 |

Знаком (-) указывается отрицательное стремление критерия (чем меньше, тем лучше), а знаком (+) – положительное (чем больше, тем лучше). В таблице 1.1.2 приведен результат попарного сравнения альтернатив.

Таблица 1.1.2 – Сравнения альтернатив

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | A1 | A2 | A3 | A4 | A5 | A6 | A7 | A8 | A9 | A10 |
| A1 | x | x | x | x | x | x | x | x | x | x |
| A2 | А2 | x | x | x | x | x | x | x | x | x |
| A3 | А3 | н | x | x | x | x | x | x | x | x |
| A4 | н | н | н | x | x | x | x | x | x | x |
| A5 | н | н | н | н | x | x | x | x | x | x |
| A6 | А6 | н | н | н | н | x | x | x | x | x |
| A7 | н | н | н | н | А5 | А6 | x | x | x | x |
| A8 | н | н | н | н | н | н | н | x | x | x |
| A9 | н | н | н | н | н | н | н | н | x | x |
| A10 | А10 | н | н | н | н | А10 | А10 | А10 | н | x |

Парето-оптимальное множество определено альтернативами {2, 3, 4, 5, 9, 10} (МГТУ имени Н.Э. Баумана, ВШЭ, МАИ, ИТМО, НИЯУ МИФИ, МФТИ). Парето-оптимальное множество представлено в Таблице 1.1.3.

Таблица 1.1.3 – Парето-оптимальное множество

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| № | Варианты  решений | Критерии | | | | | |
| Проходной балл (+) | Количество бюджетных мест (-) | Стоимость обучения (руб.) (+) | Размер стипендии (руб.) (+) | Рейтинг университета (баллы) (+) | Рассто-  яние до обще-жития (км) (-) |
| 2 | МГТУ имени Н.Э. Баумана | 291 | 84 | 363040 | 1854 | 964 | 4,2 |
| 3 | ВШЭ | 295 | 135 | 700000 | 1994 | 878 | 7,9 |
| 4 | МАИ | 247 | 25 | 277770 | 1765 | 761 | 12,4 |
| 5 | ИТМО | 304 | 45 | 349000 | 2252 | 838 | 3,4 |
| 9 | НИЯУ МИФИ | 300 | 37 | 287000 | 1875 | 975 | 15 |
| 10 | МФТИ | 290 | 30 | 432000 | 1900 | 965 | 4,8 |

Очевидно, что выделение множества Парето часто не является удовлетворительным решением. Это связано с тем, что при достаточно большом исходном множестве вариантов множество Парето оказывается недопустимо большим для того, чтобы ЛПР было бы в состоянии осуществить выбор самостоятельно. Таким образом, выделение множества Парето можно рассматривать лишь как предварительный этап оптимизации, и налицо проблема дальнейшего сокращения этого множества.

1.3 Указание верхних/нижних границ критериев

Установим верхнюю и нижнюю границы. Нижняя граница: проходной балл должен быть не ниже 270 и рейтинг университета должен быть не ниже 840 баллов. Верхняя граница: расстояние до общежития должно быть менее 14 км. В таблице 1.3.1 приведено множество альтернатив после отсечения по верхним и нижним границам.

Таблица 1.3.1 – Результат указания границ критериев

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| № | Варианты  решений | Критерии | | | | | |
| Проходной балл (+) | Количество бюджетных мест (-) | Стоимость обучения (руб.) (+) | Размер стипендии (руб.) (+) | Рейтинг университета (баллы) (+) | Рассто-  яние до обще-жития (км) (-) |
| 1 | РТУ МИРЭА | 276 | 285 | 320000 | 1800 | 843 | 13,7 |
| 2 | МГТУ имени Н.Э. Баумана | 291 | 84 | 363040 | 1854 | 964 | 4,2 |
| 3 | ВШЭ | 295 | 135 | 700000 | 1994 | 878 | 7,9 |
| 6 | СПбГУ | 276 | 35 | 335300 | 1843 | 915 | 7,3 |
| 10 | МФТИ | 290 | 30 | 432000 | 1900 | 965 | 4,8 |

Заметим, что альтернативы A2, A3, А6, A10 доминируют над A1 в то время, как А10 доминирует над А6. Следовательно, Парето-оптимальное множество состоит из альтернатив {2, 3, 10}. Единственную альтернативу выбрать не удалось в силу не слишком «жестких границ» для критериев, однако указание границ помогло уменьшить размер Парето-множества.

1.4 Субоптимизация

Главный критерий: рейтинг университета. Нижняя граница: проходной балл должен быть не ниже 290 и рейтинг университета должен быть не ниже 840 баллов. Верхняя граница: расстояние до общежития должно быть менее 14 км. Отбросим варианты, которые не удовлетворяют данным ограничениям и составим таблицу 1.4.1 результата субоптимизации.

Таблица 1.4.1 – Результат субоптимизации

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| № | Варианты  решений | Критерии | | | | | |
| Проходной балл (+) | Количество бюджетных мест (-) | Стоимость обучения (руб.) (+) | Размер стипендии (руб.) (+) | Рейтинг университета (баллы) (+) | Рассто-  яние до обще-жития (км) (-) |
| 2 | МГТУ имени Н.Э. Баумана | 291 | 84 | 363040 | 1854 | 964 | 4,2 |
| 3 | ВШЭ | 295 | 135 | 700000 | 1994 | 878 | 7,9 |
| 10 | МФТИ | 290 | 30 | 432000 | 1900 | 965 | 4,8 |

Учитывая главный критерий, оптимальным решением является МФТИ, потому что у этого университета наибольший рейтинг. Заметим, что окончательное решение имеет субъективный характер.

1.5 Лексикографическая оптимизация

Лексикографическая оптимизация основана на упорядочении критериев по их относительной важности. На первом шаге отбирают исходы, которые имеют максимальную оценку по важнейшему критерию. Если такой исход единственный, то его и считают оптимальным. Если же таких исходов несколько, то среди них отбирают те, которые имеют максимальную оценку по следующему за важнейшим критерию.

Установим следующий приоритет критериев: рейтинг университета, проходной балл, стоимость обучения, количество бюджетных мест, расстояние до общежития, размер стипендии.

При сравнении альтернатив лишь по первому критерию остается всего одно решение, которое является наилучшим. Вынесем оптимальное решение в Таблицу 1.5.1.

Таблица 1.5.1 – Результат лексикографической оптимизации

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| № | Варианты  решений | Критерии | | | | | |
| Проходной балл (+) | Количество бюджетных мест (-) | Стоимость обучения (руб.) (+) | Размер стипендии (руб.) (+) | Рейтинг университета (баллы) (+) | Рассто-  яние до обще-жития (км) (-) |
| 9 | НИЯУ МИФИ | 300 | 37 | 287000 | 1875 | 975 | 15 |

Лексикографическая оптимизация позволяет выделить единственное наилучшее решение.

## 1.6 Результаты работы программы

Метод Парето и методы его оптимизации были реализованы в программе, результаты работы которой представлены на Рисунках 1.6.1 – 1.6.5.

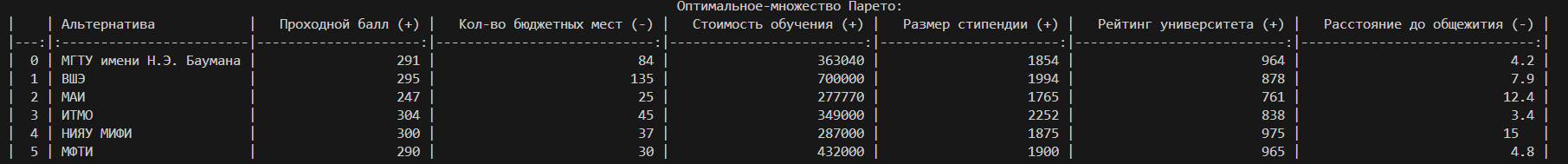


Рисунок 1.6.1 – Парето-оптимальное множество

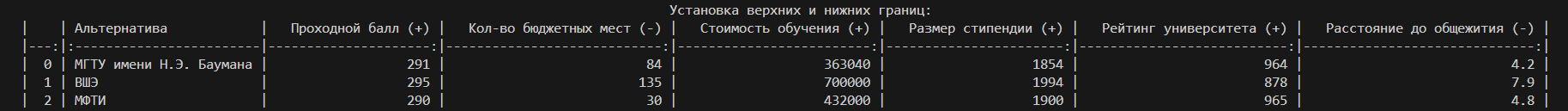


Рисунок 1.6.2 – Установки верхних/нижних границ для критериев

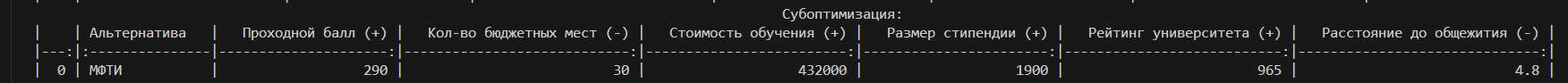


Рисунок 1.6.3 – Субоптимизация

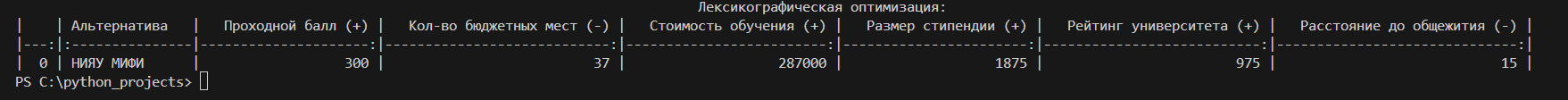


Рисунок 1.6.4 – Лексикографическая оптимизация

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 1.6.5 – Границы для оптимизации и приоритеты критериев

## 1.7 Заключение

В ходе выполнения данной практической работы мной был изучен метод Парето, применён на практике для определения оптимального решения в задаче нахождения оптимального высшего учебного заведения, а также сделана программная реализация. Это было проделано также и для методов сужения.

Основным плюсом метода Парето является простота реализации, однако в результате может получиться несколько оптимальных альтернатив, из-за чего ЛПР придётся самостоятельно выбирать одно из них. Методы сужения помогают решить эту проблему, однако выбор характера сужения носит субъективный характер.

2 метод электра II

2.1 Введение

Цель работы: изучить метод Электра II и научиться применять его в нахождении оптимального решения в выбранной предметной области.

Предметная область: выбор оптимального высшего учебного заведения.

Метод Электра II состоит из нескольких этапов. На первом этапе определяется множество решений и для каждого из N критериев определяется вес – число, характеризующее важность соответствующего критерия. На втором этапе для каждой пары альтернатив вычисляется P+ - сумма весов критериев, по которым одна альтернатива предпочтительнее другой, и P- - сумма весов критериев, по которым эта же альтернатива менее предпочтительна по сравнению с другой. На третьем этапе вычисляются отношения P+/ P-, и если полученное отношение больше 1, то оно сохраняется в матрицу, а если меньше или равно, то не сохраняется.

На основе полученной матрицы строится граф предпочтений, и если в нём обнаруживаются петли, то назначается порог, который отбрасывает слабые связи, то есть те пары альтернатив, которые не сильно отличаются друг от друга. Если в графе не осталось петель и граф остался целостным, то выбираются та альтернативы, к которым не идёт ни одно ребро на графе. Они являются оптимальными.

2.2 Выбор лучшего варианта

Составлена таблица критериев, по которым оцениваются университеты (Таблица 2.2.1).

Таблица 2.2.1 – Таблица критериев для оценки альтернатив

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Критерии | Вес критерия | Шкала | Код | Стремление |
| Проходной балл (+) | 4 | Более 270  Более 250  Не более 250 | 15  10  5 | max |
| Количество бюджетных мест (-) | 5 | Более 100  Более 50  Не более 50 | 15  10  5 | min |
| Стоимость обучения (руб.) (+) | 2 | Более 350 тыс. рублей  Более 250 тыс. рублей  Не более 250 тыс. | 15  10  5 | max |
| Размер стипендии (руб.) (+) | 5 | Больше 2 тыс. рублей  Не более 2 тыс. рублей | 10  5 | max |
| Рейтинг университета (баллы) (+) | 2 | Больше 900 баллов  Больше 800 баллов  Не более 800 | 15  10  5 | max |
| Расстояние до общежития (км) (-) | 4 | Больше 10 км  Больше 5 км  Не более 5 км | 15  10  5 | min |

Составлена таблица оценок выбора оптимального технического университета. Для 10-ти альтернатив заполнена Таблица 2.2.2.

Таблица 2.2.2 – Таблица оценок по критериям

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| № | Варианты решений | Критерии | | | | | |
| Проходной балл (+) | Количество бюджетных мест (-) | Стоимость обучения (руб.) (+) | Размер стипендии (руб.) (+) | Рейтинг университета (баллы) (+) | Рассто-  яние до обще-жития (км) (-) |
| 1 | РТУ МИРЭА | 15 | 15 | 10 | 10 | 10 | 5 |
| 2 | МГТУ имени Н.Э. Баумана | 15 | 10 | 15 | 10 | 15 | 15 |
| 3 | ВШЭ | 15 | 15 | 15 | 10 | 10 | 5 |
| 4 | МАИ | 5 | 5 | 10 | 5 | 5 | 5 |
| 5 | ИТМО | 15 | 5 | 10 | 10 | 10 | 10 |
| 6 | СПбГУ | 15 | 5 | 10 | 10 | 15 | 15 |
| 7 | МТУСИ | 10 | 10 | 10 | 5 | 5 | 5 |

Продолжение Таблицы 2.2.2

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 8 | СГУ им. Чернышев-ского | 5 | 5 | 5 | 5 | 10 | 5 |
| 9 | НИЯУ МИФИ | 15 | 5 | 10 | 10 | 15 | 15 |
| 10 | МФТИ | 15 | 5 | 15 | 5 | 15 | 15 |
| Вес | | 4 | 5 | 2 | 5 | 2 | 4 |
| Стремление | | max | min | max | max | max | min |

## 2.3 Веса предпочтений

Рассмотрим альтернативы 1 и 2 (i = 1, j = 2):

P12 = 0 + 0 + 0 + 0 + 0 + 4 = 4;

N12 = 0 + 5 + 2 + 0 + 2 + 0 = 9;

D12 = P12 / N12 = 4/9 = 0.44 <= 1 - отбрасываем.

P21 = 0 + 5 + 2 + 0 + 2 + 0 = 9;

N21 = 0 + 0 + 0 + 0 + 0 + 4 = 4;

D21 = P21 / N21 = 9/4 = 2.25 > 1 - принимаем.

Рассмотрим альтернативы 1 и 3 (i = 1, j = 3):

P13 = 0 + 0 + 0 + 0 + 0 + 0 = 0;

N13 = 0 + 0 + 2 + 0 + 0 + 0 = 2;

D13 = P13 / N13 = 0/2 = 0 <= 1 - отбрасываем.

P31 = 0 + 0 + 2 + 0 + 0 + 0 = 2;

N31 = 0 + 0 + 0 + 0 + 0 + 0 = 0;

D31 = P31 / N31 = 2/0 = ∞ > 1 - принимаем.

Рассмотрим альтернативы 1 и 4 (i = 1, j = 4):

P14 = 4 + 0 + 0 + 5 + 2 + 0 = 11;

N14 = 0 + 5 + 0 + 0 + 0 + 0 = 5;

D14 = P14 / N14 = 11/5 = 2.2 > 1 - принимаем.

P41 = 0 + 5 + 0 + 0 + 0 + 0 = 5;

N41 = 4 + 0 + 0 + 5 + 2 + 0 = 11;

D41 = P41 / N41 = 5/11 = 0.45 <= 1 - отбрасываем.

Рассмотрим альтернативы 1 и 5 (i = 1, j = 5):

P15 = 0 + 0 + 0 + 0 + 0 + 4 = 4;

N15 = 0 + 5 + 0 + 0 + 0 + 0 = 5;

D15 = P15 / N15 = 4/5 = 0.8 <= 1 - отбрасываем.

P51 = 0 + 5 + 0 + 0 + 0 + 0 = 5;

N51 = 0 + 0 + 0 + 0 + 0 + 4 = 4;

D51 = P51 / N51 = 5/4 = 1.25 > 1 - принимаем.

Рассмотрим альтернативы 1 и 6 (i = 1, j = 6):

P16 = 0 + 0 + 0 + 0 + 0 + 4 = 4;

N16 = 0 + 5 + 0 + 0 + 2 + 0 = 7;

D16 = P16 / N16 = 4/7 = 0.57 <= 1 - отбрасываем.

P61 = 0 + 5 + 0 + 0 + 2 + 0 = 7;

N61 = 0 + 0 + 0 + 0 + 0 + 4 = 4;

D61 = P61 / N61 = 7/4 = 1.75 > 1 - принимаем.

Рассмотрим альтернативы 1 и 7 (i = 1, j = 7):

P17 = 4 + 0 + 0 + 5 + 2 + 0 = 11;

N17 = 0 + 5 + 0 + 0 + 0 + 0 = 5;

D17 = P17 / N17 = 11/5 = 2.2 > 1 - принимаем.

P71 = 0 + 5 + 0 + 0 + 0 + 0 = 5;

N71 = 4 + 0 + 0 + 5 + 2 + 0 = 11;

D71 = P71 / N71 = 5/11 = 0.45 <= 1 - отбрасываем.

Рассмотрим альтернативы 1 и 8 (i = 1, j = 8):

P18 = 4 + 0 + 2 + 5 + 0 + 0 = 11;

N18 = 0 + 5 + 0 + 0 + 0 + 0 = 5;

D18 = P18 / N18 = 11/5 = 2.2 > 1 - принимаем.

P81 = 0 + 5 + 0 + 0 + 0 + 0 = 5;

N81 = 4 + 0 + 2 + 5 + 0 + 0 = 11;

D81 = P81 / N81 = 5/11 = 0.45 <= 1 - отбрасываем.

Рассмотрим альтернативы 1 и 9 (i = 1, j = 9):

P19 = 0 + 0 + 0 + 0 + 0 + 4 = 4;

N19 = 0 + 5 + 0 + 0 + 2 + 0 = 7;

D19 = P19 / N19 = 4/7 = 0.57 <= 1 - отбрасываем.

P91 = 0 + 5 + 0 + 0 + 2 + 0 = 7;

N91 = 0 + 0 + 0 + 0 + 0 + 4 = 4;

D91 = P91 / N91 = 7/4 = 1.75 > 1 - принимаем.

Рассмотрим альтернативы 1 и 10 (i = 1, j = 10):

P110 = 0 + 0 + 0 + 5 + 0 + 4 = 9;

N110 = 0 + 5 + 2 + 0 + 2 + 0 = 9;

D110 = P110 / N110 = 9/9 = 1 <= 1 - отбрасываем.

P1010 = 0 + 5 + 2 + 0 + 2 + 0 = 9;

N1010 = 0 + 0 + 0 + 5 + 0 + 4 = 9;

D101 = P101 / N101 = 9/9 = 1 <= 1 - отбрасываем.

Рассмотрим альтернативы 2 и 3 (i = 2, j = 3):

P23 = 0 + 5 + 0 + 0 + 2 + 0 = 7;

N23 = 0 + 0 + 0 + 0 + 0 + 4 = 4;

D23 = P23 / N23 = 7/4 = 1.75 > 1 - принимаем.

P32 = 0 + 0 + 0 + 0 + 0 + 4 = 4;

N32 = 0 + 5 + 0 + 0 + 2 + 0 = 7;

D32 = P32 / N32 = 4/7 = 0.57 <= 1 - отбрасываем.

Рассмотрим альтернативы 2 и 4 (i = 2, j = 4):

P24 = 4 + 0 + 2 + 5 + 2 + 0 = 13;

N24 = 0 + 5 + 0 + 0 + 0 + 4 = 9;

D24 = P24 / N24 = 13/9 = 1.44 > 1 - принимаем.

P42 = 0 + 5 + 0 + 0 + 0 + 4 = 9;

N42 = 4 + 0 + 2 + 5 + 2 + 0 = 13;

D42 = P42 / N42 = 9/13 = 0.69 <= 1 - отбрасываем.

Рассмотрим альтернативы 2 и 5 (i = 2, j = 5):

P25 = 0 + 0 + 2 + 0 + 2 + 0 = 4;

N25 = 0 + 5 + 0 + 0 + 0 + 4 = 9;

D25 = P25 / N25 = 4/9 = 0.44 <= 1 - отбрасываем.

P52 = 0 + 5 + 0 + 0 + 0 + 4 = 9;

N52 = 0 + 0 + 2 + 0 + 2 + 0 = 4;

D52 = P52 / N52 = 9/4 = 2.25 > 1 - принимаем.

Рассмотрим альтернативы 2 и 6 (i = 2, j = 6):

P26 = 0 + 0 + 2 + 0 + 0 + 0 = 2;

N26 = 0 + 5 + 0 + 0 + 0 + 0 = 5;

D26 = P26 / N26 = 2/5 = 0.4 <= 1 - отбрасываем.

P62 = 0 + 5 + 0 + 0 + 0 + 0 = 5;

N62 = 0 + 0 + 2 + 0 + 0 + 0 = 2;

D62 = P62 / N62 = 5/2 = 2.5 > 1 - принимаем.

Рассмотрим альтернативы 2 и 7 (i = 2, j = 7):

P27 = 4 + 0 + 2 + 5 + 2 + 0 = 13;

N27 = 0 + 0 + 0 + 0 + 0 + 4 = 4;

D27 = P27 / N27 = 13/4 = 3.25 > 1 - принимаем.

P72 = 0 + 0 + 0 + 0 + 0 + 4 = 4;

N72 = 4 + 0 + 2 + 5 + 2 + 0 = 13;

D72 = P72 / N72 = 4/13 = 0.31 <= 1 - отбрасываем.

Рассмотрим альтернативы 2 и 8 (i = 2, j = 8):

P28 = 4 + 0 + 2 + 5 + 2 + 0 = 13;

N28 = 0 + 5 + 0 + 0 + 0 + 4 = 9;

D28 = P28 / N28 = 13/9 = 1.44 > 1 - принимаем.

P82 = 0 + 5 + 0 + 0 + 0 + 4 = 9;

N82 = 4 + 0 + 2 + 5 + 2 + 0 = 13;

D82 = P82 / N82 = 9/13 = 0.69 <= 1 - отбрасываем.

Рассмотрим альтернативы 2 и 9 (i = 2, j = 9):

P29 = 0 + 0 + 2 + 0 + 0 + 0 = 2;

N29 = 0 + 5 + 0 + 0 + 0 + 0 = 5;

D29 = P29 / N29 = 2/5 = 0.4 <= 1 - отбрасываем.

P92 = 0 + 5 + 0 + 0 + 0 + 0 = 5;

N92 = 0 + 0 + 2 + 0 + 0 + 0 = 2;

D92 = P92 / N92 = 5/2 = 2.5 > 1 - принимаем.

Рассмотрим альтернативы 2 и 10 (i = 2, j = 10):

P210 = 0 + 0 + 0 + 5 + 0 + 0 = 5;

N210 = 0 + 5 + 0 + 0 + 0 + 0 = 5;

D210 = P210 / N210 = 5/5 = 1 <= 1 - отбрасываем.

P1020 = 0 + 5 + 0 + 0 + 0 + 0 = 5;

N1020 = 0 + 0 + 0 + 5 + 0 + 0 = 5;

D102 = P102 / N102 = 5/5 = 1 <= 1 - отбрасываем.

Рассмотрим альтернативы 3 и 4 (i = 3, j = 4):

P34 = 4 + 0 + 2 + 5 + 2 + 0 = 13;

N34 = 0 + 5 + 0 + 0 + 0 + 0 = 5;

D34 = P34 / N34 = 13/5 = 2.6 > 1 - принимаем.

P43 = 0 + 5 + 0 + 0 + 0 + 0 = 5;

N43 = 4 + 0 + 2 + 5 + 2 + 0 = 13;

D43 = P43 / N43 = 5/13 = 0.38 <= 1 - отбрасываем.

Рассмотрим альтернативы 3 и 5 (i = 3, j = 5):

P35 = 0 + 0 + 2 + 0 + 0 + 4 = 6;

N35 = 0 + 5 + 0 + 0 + 0 + 0 = 5;

D35 = P35 / N35 = 6/5 = 1.2 > 1 - принимаем.

P53 = 0 + 5 + 0 + 0 + 0 + 0 = 5;

N53 = 0 + 0 + 2 + 0 + 0 + 4 = 6;

D53 = P53 / N53 = 5/6 = 0.83 <= 1 - отбрасываем.

Рассмотрим альтернативы 3 и 6 (i = 3, j = 6):

P36 = 0 + 0 + 2 + 0 + 0 + 4 = 6;

N36 = 0 + 5 + 0 + 0 + 2 + 0 = 7;

D36 = P36 / N36 = 6/7 = 0.86 <= 1 - отбрасываем.

P63 = 0 + 5 + 0 + 0 + 2 + 0 = 7;

N63 = 0 + 0 + 2 + 0 + 0 + 4 = 6;

D63 = P63 / N63 = 7/6 = 1.17 > 1 - принимаем.

Рассмотрим альтернативы 3 и 7 (i = 3, j = 7):

P37 = 4 + 0 + 2 + 5 + 2 + 0 = 13;

N37 = 0 + 5 + 0 + 0 + 0 + 0 = 5;

D37 = P37 / N37 = 13/5 = 2.6 > 1 - принимаем.

P73 = 0 + 5 + 0 + 0 + 0 + 0 = 5;

N73 = 4 + 0 + 2 + 5 + 2 + 0 = 13;

D73 = P73 / N73 = 5/13 = 0.38 <= 1 - отбрасываем.

Рассмотрим альтернативы 3 и 8 (i = 3, j = 8):

P38 = 4 + 0 + 2 + 5 + 0 + 0 = 11;

N38 = 0 + 5 + 0 + 0 + 0 + 0 = 5;

D38 = P38 / N38 = 11/5 = 2.2 > 1 - принимаем.

P83 = 0 + 5 + 0 + 0 + 0 + 0 = 5;

N83 = 4 + 0 + 2 + 5 + 0 + 0 = 11;

D83 = P83 / N83 = 5/11 = 0.45 <= 1 - отбрасываем.

Рассмотрим альтернативы 3 и 9 (i = 3, j = 9):

P39 = 0 + 0 + 2 + 0 + 0 + 4 = 6;

N39 = 0 + 5 + 0 + 0 + 2 + 0 = 7;

D39 = P39 / N39 = 6/7 = 0.86 <= 1 - отбрасываем.

P93 = 0 + 5 + 0 + 0 + 2 + 0 = 7;

N93 = 0 + 0 + 2 + 0 + 0 + 4 = 6;

D93 = P93 / N93 = 7/6 = 1.17 > 1 - принимаем.

Рассмотрим альтернативы 3 и 10 (i = 3, j = 10):

P310 = 0 + 0 + 0 + 5 + 0 + 4 = 9;

N310 = 0 + 5 + 0 + 0 + 2 + 0 = 7;

D310 = P310 / N310 = 9/7 = 1.29 > 1 - принимаем.

P1030 = 0 + 5 + 0 + 0 + 2 + 0 = 7;

N1030 = 0 + 0 + 0 + 5 + 0 + 4 = 9;

D103 = P103 / N103 = 7/9 = 0.78 <= 1 - отбрасываем.

Рассмотрим альтернативы 4 и 5 (i = 4, j = 5):

P45 = 0 + 0 + 0 + 0 + 0 + 4 = 4;

N45 = 4 + 0 + 0 + 5 + 2 + 0 = 11;

D45 = P45 / N45 = 4/11 = 0.36 <= 1 - отбрасываем.

P54 = 4 + 0 + 0 + 5 + 2 + 0 = 11;

N54 = 0 + 0 + 0 + 0 + 0 + 4 = 4;

D54 = P54 / N54 = 11/4 = 2.75 > 1 - принимаем.

Рассмотрим альтернативы 4 и 6 (i = 4, j = 6):

P46 = 0 + 0 + 0 + 0 + 0 + 4 = 4;

N46 = 4 + 0 + 0 + 5 + 2 + 0 = 11;

D46 = P46 / N46 = 4/11 = 0.36 <= 1 - отбрасываем.

P64 = 4 + 0 + 0 + 5 + 2 + 0 = 11;

N64 = 0 + 0 + 0 + 0 + 0 + 4 = 4;

D64 = P64 / N64 = 11/4 = 2.75 > 1 - принимаем.

Рассмотрим альтернативы 4 и 7 (i = 4, j = 7):

P47 = 0 + 5 + 0 + 0 + 0 + 0 = 5;

N47 = 4 + 0 + 0 + 0 + 0 + 0 = 4;

D47 = P47 / N47 = 5/4 = 1.25 > 1 - принимаем.

P74 = 4 + 0 + 0 + 0 + 0 + 0 = 4;

N74 = 0 + 5 + 0 + 0 + 0 + 0 = 5;

D74 = P74 / N74 = 4/5 = 0.8 <= 1 - отбрасываем.

Рассмотрим альтернативы 4 и 8 (i = 4, j = 8):

P48 = 0 + 0 + 2 + 0 + 0 + 0 = 2;

N48 = 0 + 0 + 0 + 0 + 2 + 0 = 2;

D48 = P48 / N48 = 2/2 = 1 <= 1 - отбрасываем.

P84 = 0 + 0 + 0 + 0 + 2 + 0 = 2;

N84 = 0 + 0 + 2 + 0 + 0 + 0 = 2;

D84 = P84 / N84 = 2/2 = 1 <= 1 - отбрасываем.

Рассмотрим альтернативы 4 и 9 (i = 4, j = 9):

P49 = 0 + 0 + 0 + 0 + 0 + 4 = 4;

N49 = 4 + 0 + 0 + 5 + 2 + 0 = 11;

D49 = P49 / N49 = 4/11 = 0.36 <= 1 - отбрасываем.

P94 = 4 + 0 + 0 + 5 + 2 + 0 = 11;

N94 = 0 + 0 + 0 + 0 + 0 + 4 = 4;

D94 = P94 / N94 = 11/4 = 2.75 > 1 - принимаем.

Рассмотрим альтернативы 4 и 10 (i = 4, j = 10):

P410 = 0 + 0 + 0 + 0 + 0 + 4 = 4;

N410 = 4 + 0 + 2 + 0 + 2 + 0 = 8;

D410 = P410 / N410 = 4/8 = 0.5 <= 1 - отбрасываем.

P1040 = 4 + 0 + 2 + 0 + 2 + 0 = 8;

N1040 = 0 + 0 + 0 + 0 + 0 + 4 = 4;

D104 = P104 / N104 = 8/4 = 2 > 1 - принимаем.

Рассмотрим альтернативы 5 и 6 (i = 5, j = 6):

P56 = 0 + 0 + 0 + 0 + 0 + 4 = 4;

N56 = 0 + 0 + 0 + 0 + 2 + 0 = 2;

D56 = P56 / N56 = 4/2 = 2 > 1 - принимаем.

P65 = 0 + 0 + 0 + 0 + 2 + 0 = 2;

N65 = 0 + 0 + 0 + 0 + 0 + 4 = 4;

D65 = P65 / N65 = 2/4 = 0.5 <= 1 - отбрасываем.

Рассмотрим альтернативы 5 и 7 (i = 5, j = 7):

P57 = 4 + 5 + 0 + 5 + 2 + 0 = 16;

N57 = 0 + 0 + 0 + 0 + 0 + 4 = 4;

D57 = P57 / N57 = 16/4 = 4 > 1 - принимаем.

P75 = 0 + 0 + 0 + 0 + 0 + 4 = 4;

N75 = 4 + 5 + 0 + 5 + 2 + 0 = 16;

D75 = P75 / N75 = 4/16 = 0.25 <= 1 - отбрасываем.

Рассмотрим альтернативы 5 и 8 (i = 5, j = 8):

P58 = 4 + 0 + 2 + 5 + 0 + 0 = 11;

N58 = 0 + 0 + 0 + 0 + 0 + 4 = 4;

D58 = P58 / N58 = 11/4 = 2.75 > 1 - принимаем.

P85 = 0 + 0 + 0 + 0 + 0 + 4 = 4;

N85 = 4 + 0 + 2 + 5 + 0 + 0 = 11;

D85 = P85 / N85 = 4/11 = 0.36 <= 1 - отбрасываем.

Рассмотрим альтернативы 5 и 9 (i = 5, j = 9):

P59 = 0 + 0 + 0 + 0 + 0 + 4 = 4;

N59 = 0 + 0 + 0 + 0 + 2 + 0 = 2;

D59 = P59 / N59 = 4/2 = 2 > 1 - принимаем.

P95 = 0 + 0 + 0 + 0 + 2 + 0 = 2;

N95 = 0 + 0 + 0 + 0 + 0 + 4 = 4;

D95 = P95 / N95 = 2/4 = 0.5 <= 1 - отбрасываем.

Рассмотрим альтернативы 5 и 10 (i = 5, j = 10):

P510 = 0 + 0 + 0 + 5 + 0 + 4 = 9;

N510 = 0 + 0 + 2 + 0 + 2 + 0 = 4;

D510 = P510 / N510 = 9/4 = 2.25 > 1 - принимаем.

P1050 = 0 + 0 + 2 + 0 + 2 + 0 = 4;

N1050 = 0 + 0 + 0 + 5 + 0 + 4 = 9;

D105 = P105 / N105 = 4/9 = 0.44 <= 1 - отбрасываем.

Рассмотрим альтернативы 6 и 7 (i = 6, j = 7):

P67 = 4 + 5 + 0 + 5 + 2 + 0 = 16;

N67 = 0 + 0 + 0 + 0 + 0 + 4 = 4;

D67 = P67 / N67 = 16/4 = 4 > 1 - принимаем.

P76 = 0 + 0 + 0 + 0 + 0 + 4 = 4;

N76 = 4 + 5 + 0 + 5 + 2 + 0 = 16;

D76 = P76 / N76 = 4/16 = 0.25 <= 1 - отбрасываем.

Рассмотрим альтернативы 6 и 8 (i = 6, j = 8):

P68 = 4 + 0 + 2 + 5 + 2 + 0 = 13;

N68 = 0 + 0 + 0 + 0 + 0 + 4 = 4;

D68 = P68 / N68 = 13/4 = 3.25 > 1 - принимаем.

P86 = 0 + 0 + 0 + 0 + 0 + 4 = 4;

N86 = 4 + 0 + 2 + 5 + 2 + 0 = 13;

D86 = P86 / N86 = 4/13 = 0.31 <= 1 - отбрасываем.

Рассмотрим альтернативы 6 и 9 (i = 6, j = 9):

P69 = 0 + 0 + 0 + 0 + 0 + 0 = 0;

N69 = 0 + 0 + 0 + 0 + 0 + 0 = 0;

D69 = P69 / N69 = 0/0 = 1 <= 1 - отбрасываем.

P96 = 0 + 0 + 0 + 0 + 0 + 0 = 0;

N96 = 0 + 0 + 0 + 0 + 0 + 0 = 0;

D96 = P96 / N96 = 0/0 = 1 <= 1 - отбрасываем.

Рассмотрим альтернативы 6 и 10 (i = 6, j = 10):

P610 = 0 + 0 + 0 + 5 + 0 + 0 = 5;

N610 = 0 + 0 + 2 + 0 + 0 + 0 = 2;

D610 = P610 / N610 = 5/2 = 2.5 > 1 - принимаем.

P1060 = 0 + 0 + 2 + 0 + 0 + 0 = 2;

N1060 = 0 + 0 + 0 + 5 + 0 + 0 = 5;

D106 = P106 / N106 = 2/5 = 0.4 <= 1 - отбрасываем.

Рассмотрим альтернативы 7 и 8 (i = 7, j = 8):

P78 = 4 + 0 + 2 + 0 + 0 + 0 = 6;

N78 = 0 + 5 + 0 + 0 + 2 + 0 = 7;

D78 = P78 / N78 = 6/7 = 0.86 <= 1 - отбрасываем.

P87 = 0 + 5 + 0 + 0 + 2 + 0 = 7;

N87 = 4 + 0 + 2 + 0 + 0 + 0 = 6;

D87 = P87 / N87 = 7/6 = 1.17 > 1 - принимаем.

Рассмотрим альтернативы 7 и 9 (i = 7, j = 9):

P79 = 0 + 0 + 0 + 0 + 0 + 4 = 4;

N79 = 4 + 5 + 0 + 5 + 2 + 0 = 16;

D79 = P79 / N79 = 4/16 = 0.25 <= 1 - отбрасываем.

P97 = 4 + 5 + 0 + 5 + 2 + 0 = 16;

N97 = 0 + 0 + 0 + 0 + 0 + 4 = 4;

D97 = P97 / N97 = 16/4 = 4 > 1 - принимаем.

Рассмотрим альтернативы 7 и 10 (i = 7, j = 10):

P710 = 0 + 0 + 0 + 0 + 0 + 4 = 4;

N710 = 4 + 5 + 2 + 0 + 2 + 0 = 13;

D710 = P710 / N710 = 4/13 = 0.31 <= 1 - отбрасываем.

P1070 = 4 + 5 + 2 + 0 + 2 + 0 = 13;

N1070 = 0 + 0 + 0 + 0 + 0 + 4 = 4;

D107 = P107 / N107 = 13/4 = 3.25 > 1 - принимаем.

Рассмотрим альтернативы 8 и 9 (i = 8, j = 9):

P89 = 0 + 0 + 0 + 0 + 0 + 4 = 4;

N89 = 4 + 0 + 2 + 5 + 2 + 0 = 13;

D89 = P89 / N89 = 4/13 = 0.31 <= 1 - отбрасываем.

P98 = 4 + 0 + 2 + 5 + 2 + 0 = 13;

N98 = 0 + 0 + 0 + 0 + 0 + 4 = 4;

D98 = P98 / N98 = 13/4 = 3.25 > 1 - принимаем.

Рассмотрим альтернативы 8 и 10 (i = 8, j = 10):

P810 = 0 + 0 + 0 + 0 + 0 + 4 = 4;

N810 = 4 + 0 + 2 + 0 + 2 + 0 = 8;

D810 = P810 / N810 = 4/8 = 0.5 <= 1 - отбрасываем.

P1080 = 4 + 0 + 2 + 0 + 2 + 0 = 8;

N1080 = 0 + 0 + 0 + 0 + 0 + 4 = 4;

D108 = P108 / N108 = 8/4 = 2 > 1 - принимаем.

Рассмотрим альтернативы 9 и 10 (i = 9, j = 10):

P910 = 0 + 0 + 0 + 5 + 0 + 0 = 5;

N910 = 0 + 0 + 2 + 0 + 0 + 0 = 2;

D910 = P910 / N910 = 5/2 = 2.5 > 1 - принимаем.

P1090 = 0 + 0 + 2 + 0 + 0 + 0 = 2;

N1090 = 0 + 0 + 0 + 5 + 0 + 0 = 5;

D109 = P109 / N109 = 2/5 = 0.4 <= 1 - отбрасываем.

Составлена матрица предпочтений с внесенными и принятыми значениями D (Таблица 2.3.1).

Таблица 2.3.1 – Полная матрица предпочтений альтернатив

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 1 | - | - | - | 2.2 | - | - | 2.2 | 2.2 | - | - |
| 2 | 2.25 | - | 1.75 | 1.44 | - | - | 3.25 | 1.44 | - | - |
| 3 | ∞ | - | - | 2.6 | 1.2 | - | 2.6 | 2.2 | - | 1.29 |
| 4 | - | - | - | - | - | - | 1.25 | - | - | - |

Продолжение Таблицы 2.3.1

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 5 | 1.25 | 2.25 | - | 2.75 | - | 2 | 4 | 2.75 | 2 | 2.25 |
| 6 | 1.75 | 2.5 | 1.17 | 2.75 | - | - | 4 | 3.25 | - | 2.5 |
| 7 | - | - | - | - | - | - | - | - | - | - |
| 8 | - | - | - | - | - | - | 1.17 | - | - | - |
| 9 | 1.75 | 2.5 | 1.17 | 2.75 | - | - | 4 | 3.25 | - | 2.5 |
| 10 | - | - | - | 2 | - | - | 3.25 | 2 | - | - |

По матрице построен граф предпочтений (Рисунок 2.3.1).

Изображение выглядит как зарисовка, рисунок, Штриховая графика, диаграмма

Автоматически созданное описание

Рисунок 2.3.1 – Вид графа предпочтений

Назначен порог отбора предпочтений C = 1.76 (это соответствует тому, что учитываются только более сильные связи в графе).

Таким образом, матрица разрежается. В ней остаются только самые сильные связи (Таблица 2.3.2).

Таблица 2.3.2 **–** Матрица предпочтений проектов, при пороге С=1.76

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 1 | - | - | - | 2.2 | - | - | 2.2 | 2.2 | - | - |
| 2 | 2.25 | - | - | - | - | - | 3.25 | - | - | - |
| 3 | ∞ | - | - | 2.6 | - | - | 2.6 | 2.2 | - | - |
| 4 | - | - | - | - | - | - | - | - | - | - |
| 5 | - | 2.25 | - | 2.75 | - | 2 | 4 | 2.75 | 2 | 2.25 |
| 6 | - | 2.5 | - | 2.75 | - | - | 4 | 3.25 | - | 2.5 |
| 7 | - | - | - | - | - | - | - | - | - | - |
| 8 | - | - | - | - | - | - | - | - | - | - |
| 9 | - | 2.5 | - | 2.75 | - | - | 4 | 3.25 | - | 2.5 |
| 10 | - | - | - | 2 | - | - | 3.25 | 2 | - | - |

По этой матрице построен граф предпочтений (Рисунок 2.3.2).

Изображение выглядит как зарисовка, рисунок, Штриховая графика, диаграмма

Автоматически созданное описание

Рисунок 2.3.2 – Вид графа предпочтений для случая порога принятия решений C = 1.76

Циклов в графе нет, при этом граф остался целостным. Оптимальным решением является альтернатива А5 и A3.

2.4 Вывод

Метод Электра II позволяет определить оптимальное решение, уменьшив субъективный фактор, который был у метода Парето и у методов сужения, однако если ставить порог равным 1, то в графе могут появляться циклы, из-за которых невозможно определить оптимальное решение. Поэтому нужно экспериментально определять подходящее значение порога.

2.5 Результат работы программы

Результаты работы программы, реализующей метод Электра II, представлены на Рисунках 2.5.1 – 2.5.2.

Изображение выглядит как снимок экрана, текст

Автоматически созданное описание

Рисунок 2.5.1 – Вывод матрицы предпочтений

Изображение выглядит как снимок экрана

Автоматически созданное описание

Рисунок 2.5.2 – Вывод матрицы предпочтений с порогом = 1.76

2.6 Заключение

В ходе данной практической работы мной был изучен метод Электра II из семейства Электра и применён для нахождения оптимального высшего технического заведения. Преимуществами метода является большая объективность по сравнению с методом Парето и его методами сужения, однако, чтобы получить единственное решение, необходимо дополнительно устанавливать порог стремления, чтобы на графе предпочтений не образовывалось циклов и чтобы он оставался целостным.

3 метод анализа иерархий

3.1 Введение

Метод анализа иерархии заключается в иерархическом представлении задачи. Метод имеет три этапа:

1. Представление задачи в виде иерархической структуры.
2. Оценка приоритетов (весов) критериев с учётом их места в иерархии относительной важности.
3. Выбор лучшей альтернативы по значениям её характеристик и важности критериев.

3.2 Постановка задачи

Задача практической работы: выбрать лучшее техническое высшее учебное заведение.

## 3.3 Представление проблемы в виде иерархии

Первый этап – представление проблемы в виде иерархии или сети. В простейшем случае, иерархия строится, начиная с цели, которая помещается в вершину иерархии. Через промежуточные уровни, на которых располагаются критерии и от которых зависят последующие уровни, к самому низкому уровню, который содержит перечень альтернатив.

Иерархия считается полной, если каждый элемент заданного уровня является критерием для всех элементов нижнего уровня. На Рисунке 3.3.1 изображена полная доминантная иерархия для поставленной задачи.

Изображение выглядит как диаграмма, линия, Технический чертеж, План

Автоматически созданное описание

Рисунок 3.3.1 – Полная доминантная иерархия

Критерии:

К 1 – проходной балл (+);

К 2 – количество бюджетных мест (-);

К 3 – стоимость обучения (+);

К 4 – рейтинг университета (баллы) (+);

К 5 – расстояние до общежития (км) (-).

Альтернативы:

А 1 – МГТУ имени Н.Э. Баумана;

А 2 – ВШЭ;

А 3 – МАИ;

А 4 – НИЯУ МИФИ;

А 5 – МФТИ.

3.4 Установка приоритетов критериев

После иерархического представления задачи установлены приоритеты критериев и оценена каждая из альтернатив по критериям, определена наиболее важная их них. В методе анализа иерархий элементы сравниваются попарно по отношению к их влиянию на общую для них характеристику. Парные сравнения приводят к записи характеристик сравнений в виде квадратной таблицы чисел, которая называется матрицей. Для облегчения работы введена шкала относительной важности (Таблица 3.4.1).

Таблица 3.4.1 – Шкала относительной важности.

|  |  |  |
| --- | --- | --- |
| Интенсивность  относительной  важности | Определение | Объяснение |
| 1 | Равная важность | Равный вклад двух критериев в цель. |
| 3 | Слабое превосходство | Дают легкое превосходство одной альтернативы над другой |
| 5 | Умеренное превосходство | Опыт и суждения дают умеренное превосходство |
| 7 | Сильное превосходство | Одному из критериев дается настолько сильное предпочтение. |
| 9 | Абсолютное превосходство | Очевидность превосходства одного критерия над другим |
| 2,4,6,8 | Промежуточные решения между двумя соседними суждениями | Применяется в компромиссных случаях |

Шкала содержит соответствующие обратные значения.

3.5 Синтез приоритетов

После построения иерархии и определения величин парных субъективных суждений следует этап, на котором иерархическая декомпозиция и относительные суждения объединяются для получения осмысленного решения многокритериальной задачи принятия решений. Из групп парных сравнений формируется набор локальных критериев, которые выражают относительное влияние элементов на элемент, расположенный на уровне выше. Составлена обратно симметричная матрица для парного сравнения критериев (Таблица 3.5.1).

Таблица 3.5.1 – Матрица парного сравнения критериев

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Цель | К 1 | К 2 | К 3 | К 4 | К 5 | Vi | W2i |
| К 1 | 1 | 2 | 2 | 1/2 | 4 | 1.516 | 0.245 |
| К 2 | 1/2 | 1 | 2 | 1/5 | 2 | 0.833 | 0.135 |
| К 3 | 1/2 | 1/2 | 1 | 1/8 | 2 | 0.574 | 0.093 |
| К 4 | 2 | 5 | 8 | 1 | 2 | 2.759 | 0.446 |
| К 5 | 1/4 | 1/2 | 1/2 | 1/2 | 1 | 0.5 | 0.081 |
| ∑Vi | | | | | | 6.182 |

Для определения относительной ценности каждого элемента необходимо найти геометрическое среднее и с этой целью перемножить n элементов каждой строки и из полученного результата извлечь корни n-й степени (размерность матрицы n = 5).

Строка № 1

V1=(1\*2\*2\*1/2\*4)1/5 =1.516;

Строка № 2

V2=(1/2\*1\*2\*1/5\*2)1/5 = 0.833;

Строка № 3

V3=(1/2\*1/2\*1\*1/8\*2)1/5 = 0.574;

Строка № 4

V4=(2\*5\*8\*1\*2)1/5 =2.759;

Строка № 5

V5=(1/4\*1/2\*1/2\*1/2\*1)1/5 =0.5.

Проведена нормализация полученных чисел. Для этого определен нормирующий коэффициент ∑Vi.

∑Vi = V1 + V2 + V3 + V4 + V5 = 1.516 + 0.833 + 0.574 + 2.759 + 0.5 = 6.182.

Найдена важность приоритетов W2i, для этого каждое из чисел Vi разделено на ∑Vi.

***Строка № 1***

W21= 1.516/ ∑Vi = 0.245 = Y21;

***Строка № 2***

W22= 0.833 / ∑Vi = 0.135 = Y22;

***Строка № 3***

W23= 0.574 / ∑Vi = 0.093 = Y23;

***Строка № 4***

W24= 2.759 / ∑Vi = 0.446 = Y24;

***Строка № 5***

W25= 0.5 / ∑Vi = 0.081 = Y25.

В результате получен вектор приоритетов:

W2i = (Y21=0.245; Y22=0.135; Y23=0.093; Y24=0.446; Y25=0.081), где индекс 2 означает, что вектор приоритетов относится ко второму уровню иерархии.

К 1 – проходной балл (Таблица 3.5.2);

Таблица 3.5.2 – Матрица сравнения по критерию 1

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| К1 | А1 | А2 | А3 | А4 | А5 | VК1Y | W3К1Y |
| А1 | 1 | 1/2 | 8 | 1/4 | 2 | 1.149 | 0.188 |
| А2 | 2 | 1 | 6 | 1/2 | 2 | 1.644 | 0.269 |
| А3 | 1/8 | 1/6 | 1 | 1/5 | 1/6 | 0.213 | 0.035 |
| А4 | 4 | 2 | 5 | 1 | 2 | 2.639 | 0.432 |
| А5 | 1/2 | 1/2 | 1/6 | 1/2 | 1 | 0.461 | 0.075 |
| ∑VК1Y | | | | | | 6.053 |

Определена относительная ценность каждого элемента.

Строка № 1

VК11=(1\*1/2\*8\*1/4\*2)1/5=1.149;

Строка № 2

VК12=(2\*1\*6\*1/2\*2)1/5=1.644;

Строка № 3

VК13=(1/8\*1/6\*1\*1/5\*1/6)1/5=0.213;

Строка № 4

VК14=(4\*2\*8\*1\*2)1/5=2.639;

Строка № 5

VК15=(1/2\*1/2\*1/6\*1/2\*1)1/5=0.461.

Проведена нормализация полученных чисел. Для этого определен нормирующий коэффициент ∑VK1Y.

∑VК1Y = VК11 + VК12 + VК13 + VК14 + VК15 = 1.149 + 1.644 + 0.213 + 2.639 + 0.461 = 6.106.

Найдена важность приоритетов W3К1Y, для этого каждое из чисел VK1Y разделено на ∑VK1Y.

Строка № 1

W3К11= 1.149 / ∑Vi = 1.149 / 6.106 = 0.188;

Строка № 2

W3К12= 1.644 / ∑Vi = 1.644 / 6.106 = 0.269;

Строка № 3

W3К13= 0.213 / ∑Vi = 0.213 / 6.106 = 0.035;

Строка № 4

W3К14= 2.639 / ∑Vi = 2.639 / 6.106 = 0.432;

Строка № 5

W3К15= 0.461 / ∑Vi = 0.461 / 6.106 = 0.075;

В результате получаем вектор приоритетов:

W3К1Y = (Y311=0.188; Y312=0.269; Y313=0.035; Y314=0.432; Y315=0.075),

где индекс 3 означает, что вектор приоритетов относится к третьему уровню иерархии критерия К1.

К 2 – количество бюджетных мест (Таблица 3.5.3):

Таблица 3.5.3 – Матрица сравнения по критерию 2

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| К2 | А1 | А2 | А3 | А4 | А5 | VК2Y | W3К2Y |
| А1 | 1 | 5 | 1/6 | 1/5 | 1/5 | 0.507 | 0.074 |
| А2 | 1/5 | 1 | 1/8 | 1/5 | 1/4 | 0.263 | 0.038 |
| А3 | 6 | 8 | 1 | 2 | 2 | 2.862 | 0.415 |
| А4 | 5 | 5 | 1/2 | 1 | 1/2 | 1.443 | 0.209 |
| А5 | 5 | 4 | 1/2 | 2 | 1 | 1.821 | 0.264 |
| ∑VК2Y | | | | | | 6.896 |  |

Определена относительная ценность каждого элемента.

Строка № 1

VК21=(1\*5\*1/6\*1/5\*1/5)1/5=0.507;

Строка № 2

VК22=(1/5\*1\*1/8\*1/5\*1/4)1/5=0.263;

Строка № 3

VК23=(6\*8\*1\*2\*2)1/5=2.862;

Строка № 4

VК24=(5\*5\*1/2\*1\*1/2)1/5=1.443;

Строка № 5

VК25=(5\*4\*1/2\*2\*1)1/5=1.821.

Проведена нормализация полученных чисел. Для этого определен нормирующий коэффициент ∑VK2Y.

∑VК2Y = VК21 + VК22 + VК23 + VК24 + VК25 = 0.507 + 0.263 + 2.862 + 1.443 + 1.821 = 6.896.

Найдена важность приоритетов W3К2Y, для этого каждое из чисел VK2Y разделено на ∑VK2Y.

Строка № 1

W3К21= 0.507 / ∑Vi = 0.507 / 6.896 = 0.074;

Строка № 2

W3К22= 0.263 / ∑Vi = 0.263 / 6.896 = 0.038;

Строка № 3

W3К23= 2.862 / ∑Vi = 2.862 / 6.896 = 0.415;

Строка № 4

W3К24= 1.443 / ∑Vi = 1.443 / 6.896 = 0.209;

Строка № 5

W3К25= 1.821 / ∑Vi = 1.821 / 6.896 = 0.264.

В результате получаем вектор приоритетов:

W3К2Y = (Y321=0.074; Y322=0.038; Y323=0.415; Y324=0.209; Y325=0.264),

где индекс 3 означает, что вектор приоритетов относится к третьему уровню иерархии критерия К2.

К 3 – стоимость обучения (Таблица 3.5.4):

Таблица 3.5.4 – Матрица сравнения по критерию 3

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| К3 | А1 | А2 | А3 | А4 | А5 | VК3Y | W3К3Y |
| А1 | 1 | 1/8 | 4 | 4 | 1/4 | 0.871 | 0.107 |
| А2 | 8 | 1 | 9 | 8 | 4 | 4.704 | 0.578 |
| А3 | 1/4 | 1/9 | 1 | 1/2 | 1/5 | 0.308 | 0.038 |
| А4 | 1/4 | 1/8 | 2 | 1 | 1/4 | 0.435 | 0.053 |
| А5 | 4 | 1/4 | 5 | 4 | 1 | 1.821 | 0.224 |
| VК35 | | | | | | 8.139 |

Определена относительная ценность каждого элемента.

Строка № 1

VК31=(1\*1/8\*4\*4\*1/4)1/5= 0.871;

Строка № 2

VК32=(8\*1\*9\*8\*4)1/5= 4.704;

Строка № 3

VК33=(1/4\*1/9\*1\*1/2\*1/5)1/5= 0.308;

Строка № 4

VК34=(1/4\*1/8\*2\*1\*1/4)1/5= 0.435;

Строка № 5

VК35=(4\*1/4\*5\*4\*1)1/5=1.821.

Проведена нормализация полученных чисел. Для этого определен нормирующий коэффициент ∑VK3Y.

∑VК3Y = VК31 + VК32 + VК33 + VК34 + VК35 = 0.871 + 4.704 + 0.308 + 0.435 + 1.821 = 8.139.

Найдена важность приоритетов W3К2Y, для этого каждое из чисел VK2Y разделено на ∑VK2Y.

Строка № 1

W3К31= 0.871 / ∑Vi = 0.871 / 8.139 = 0.107;

Строка № 2

W3К32= 4.704/ ∑Vi = 4.704 / 8.139 = 0.578;

Строка № 3

W3К33= 0.308 / ∑Vi = 0.308 / 8.139 = 0.038;

Строка № 4

W3К34= 0.435 / ∑Vi = 0.435 / 8.139 = 0.053;

Строка № 5

W3К35= 1.821 / ∑Vi = 1.821 / 8.139 = 0.224.

В результате получаем вектор приоритетов:

W3К3Y = (Y331=0.107; Y332=0.578; Y333=0.038; Y334=0.053; Y335=0.224),

где индекс 3 означает, что вектор приоритетов относится к третьему уровню иерархии критерия К3.

К 4 – рейтинг университета (Таблица 3.5.5);

Таблица 3.5.5 – Матрица сравнения по критерию 4

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| К4 | А1 | А2 | А3 | А4 | А5 | VК4Y | W3К4Y |
| А1 | 1 | 4 | 8 | 1/2 | 1 | 1.741 | 0.261 |
| А2 | 1/4 | 1 | 4 | 1/3 | 1/3 | 0.644 | 0.097 |
| А3 | 1/8 | 1/4 | 1 | 1/8 | 1/8 | 0.218 | 0.033 |
| А4 | 2 | 3 | 8 | 1 | 1 | 2.169 | 0.326 |
| А5 | 1 | 3 | 8 | 1 | 1 | 1.888 | 0.283 |
| ∑VК4Y | | | | | | 6.66 |

Определена относительная ценность каждого элемента.

Строка № 1

VК41=(1\*4\*8\*1/2\*1)1/5 = 1.741;

Строка № 2

VК42=(1/4\*1\*4\*1/3\*1/3)1/5 = 0.644;

Строка № 3

VК43=(1/8\*1/4\*1\*1/8\*1/8)1/5 = 0.218;

Строка № 4

VК44=(2\*3\*8\*1\*1)1/5 = 2.169;

Строка № 5

VК45=(1\*3\*8\*1\*1)1/5 = 1.888.

Проведена нормализация полученных чисел. Для этого определен нормирующий коэффициент ∑VK4Y.

∑VК4Y = VК41 + VК42 + VК43 + VК44 + VК45 = 1.741 + 0.644 + 0.218 + 2.169 + 1.888 = 6.66.

Найдена важность приоритетов W3К4Y, для этого каждое из чисел VK4Y разделено на ∑VK4Y.

Строка № 1

W3К41= 1.741/ ∑Vi = 1.741 / 6.66 = 0.261;

Строка № 2

W3К42= 0.644/ ∑Vi = 0.644 / 6.66 = 0.097;

Строка № 3

W3К43= 0.218/ ∑Vi = 0.218 / 6.66 = 0.033;

Строка № 4

W3К44= 2.169/ ∑Vi = 2.169 / 6.66 = 0.326;

Строка № 5

W3К45= 1.888/ ∑Vi = 1.888 / 6.66 = 0.283.

В результате получаем вектор приоритетов:

W3К4Y = (Y341=0.261; Y342=0.097; Y343=0.033; Y344=0.326; Y345=0.283),

где индекс 3 означает, что вектор приоритетов относится к третьему уровню иерархии критерия К4.

К 5 – Расстояние до общежития (Таблица 3.5.6).

Таблица 3.5.6 – Матрица сравнения по критерию 5.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| К5 | А1 | А2 | А3 | А4 | А5 | VК5Y | W3К5Y |
| А1 | 1 | 5 | 8 | 9 | 2 | 3.728 | 0.47 |
| А2 | 1/5 | 1 | 4 | 5 | 1/3 | 1.059 | 0.134 |
| А3 | 1/8 | 1/4 | 1 | 2 | 1/8 | 0.379 | 0.048 |
| А4 | 1/9 | 1/5 | 1/2 | 1 | 1/8 | 0.268 | 0.034 |
| А5 | 1/2 | 3 | 8 | 8 | 1 | 2.491 | 0.314 |
| ∑VК5Y | | | | | | 7.925 |

Определена относительная ценность каждого элемента.

Строка № 1

VК51=(1\*5\*8\*9\*2)1/5 =3.728;

Строка № 2

VК52=(1/5\*1\*4\*5\*1/3)1/5 =1.059;

Строка № 3

VК53=(1/8\*1/4\*1\*2\*1/8)1/5 =0.379;

Строка № 4

VК54=(1/9\*1/5\*1/2\*1\*1/8)1/5 =0.268;

Строка № 5

VК55=(1/2\*3\*8\*8\*1)1/5 =2.491.

Проведена нормализация полученных чисел. Для этого определен нормирующий коэффициент ∑VK5Y.

∑VК5Y = VК51 + VК52 + VК53 + VК54 + VК55 = 3.728 + 1.059 + 0.379 + 0.268 + 2.491 = 7.925.

Найдена важность приоритетов W3К5Y, для этого каждое из чисел VK5Y разделено на ∑VK5Y.

Строка № 1

W3К51= 3.728/ ∑Vi = 0.47;

Строка № 2

W3К52= 1.059/ ∑Vi = 0.134;

Строка № 3

W3К53= 0.379/ ∑Vi = 0.048;

Строка № 4

W3К54= 0.268/ ∑Vi = 0.034;

Строка № 5

W3К55= 2.491/ ∑Vi = 0.314.

В результате получаем вектор приоритетов:

W3К5Y = (Y351=0.47; Y352=0.134; Y353=0.048; Y354=0.034; Y355=0.314),

где индекс 3 означает, что вектор приоритетов относится к третьему уровню иерархии критерия К5.

## 3.6 Согласованность локальных приоритетов

Любая матрица суждений в общем случае не согласована, так как суждения отражают субъективные мнения ЛПР, а сравнение элементов, которые имеют количественные эквиваленты, может быть несогласованным из-за присутствия погрешности при проведении измерений. Совершенной согласованности парных сравнений даже в идеальном случае на практике достичь трудно. Нужен способ оценки степени согласованности при решении конкретной задачи.

Метод анализа иерархий дает возможность провести такую оценку.

Вместе с матрицей парных сравнений есть мера оценки степени отклонения от согласованности. Когда такие отклонения превышают установленные пределы тем, кто проводит решение задачи, необходимо их пересмотреть.

В таблице приведены средние значения индекса случайной согласованности (СИ) для случайных матриц суждений разного порядка.

В нашей задаче размерность матрицы n=5, тогда среднее значение индекса случайной согласованности СИ = 1,12.

Определены индекс согласованности и отношение согласованности для матрицы «Выбор лучшего технического вуза» (Таблица 3.6.1).

Таблица 3.6.1 – Матрица «Выбор лучшего технического вуза»

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Цель | К 1 | К 2 | К 3 | К 4 | К 5 | W2i |
| К 1 | 1 | 2 | 2 | 1/2 | 4 | 0.245 |
| К 2 | 1/2 | 1 | 2 | 1/5 | 2 | 0.135 |
| К 3 | 1/2 | 1/2 | 1 | 1/8 | 2 | 0.093 |
| К 4 | 2 | 5 | 8 | 1 | 2 | 0.446 |
| К 5 | 1/4 | 1/2 | 1/2 | 1/2 | 1 | 0.081 |

Определена сумма каждого столбца матрицы суждений.

S1 = 1 + 1/2 + 1/2 + 2 + 1/4 = 4.25;

S2 = 2 + 1 + 1/2 + 5 + 1/2 = 9;

S3 = 2 + 2 + 1 + 8 + 1/2 = 13.5;

S4 = 1/2 + 1/5 + 1/8 + 1 + 1/2 = 2.325;

S5 = 4 + 2 + 2 + 2 +1 = 11.

Полученный результат умножен на компоненту нормализованного вектора приоритетов, т.е. сумму суждений первого столбца на первую компоненту, сумму суждений второго столбца - на вторую и т.д.

Р1 = S1 х W21 = 1.041;

Р2 = S2 х W22 = 1.215;

Р3 = S3 х W23 = 1.256;

Р4 = S4 х W24 = 1.037;

Р5 = S5 х W25 = 0.891.

Сумма чисел Рj отражает пропорциональность предпочтений, чем ближе эта величина к n (числу объектов и видов действия в матрице парных сравнений), тем более согласованны суждения.

λmax = Р1 + Р2 + Р3 + Р4 + Р5 = 5.44.

Отклонение от согласованности выражается индексом согласованности.

ИС = (λmax - n)/(n - 1) = (5.222-5)/(5-1) = 0.11.

Отношение индекса согласованности ИС к среднему значению случайного индекса согласованности СИ называется отношением согласованности ОС.

ОС = ИС/СИ = 0.098.

Значение ОС меньше или равное 0.10 считается приемлемым, значит матрица «Выбор лучшего технического вуза» согласована.

Определены индекс согласованности и отношение согласованности для матрицы К 1 – проходной балл (Таблица 3.6.2).

Таблица 3.6.2 – Матрица сравнения по критерию 1.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| К1 | А1 | А2 | А3 | А4 | А5 | W3К1Y |
| А1 | 1 | 1/2 | 8 | 1/4 | 2 | 0.188 |
| А2 | 2 | 1 | 6 | 1/2 | 2 | 0.269 |
| А3 | 1/8 | 1/6 | 1 | 1/5 | 1/6 | 0.035 |
| А4 | 4 | 2 | 5 | 1 | 2 | 0.432 |
| А5 | 1/2 | 1/2 | 1/6 | 1/2 | 1 | 0.075 |

Определяется сумма каждого столбца матрицы суждений.

S1К1 = 1 + 2 + 1/8 + 4 + 1/2 = 7.625;

S2 К1 = 1/2 + 1 + 1/6 + 2 + 1/2 = 4.167;

S3 К1 = 8 + 6 + 1 + 5 + 1/6 = 20.167;

S4 К1 = 1/4 + 1/2 + 1/5 + 1 + 1/2 = 2.45;

S5 К1 = 2 + 2 + 1/6 + 2 + 1 = 7.167.

Затем полученный результат умножен на компоненту нормализованного вектора приоритетов.

Р1 К1 = S1 х W3К11 = 1.487;

Р2 К1 = S2 х W3К12 = 1.163;

Р3 К1 = S3 х W3К13 = 0.807;

Р4 К1 = S1 х W3К14 = 1;

Р5 К1 = S1 х W3К15 = 0.559.

Найдена пропорциональность предпочтений.

λmax К1 = Р1К1 + Р2К1 + Р3К1 + Р4К1 + Р5К1 = 5.016.

Отклонение от согласованности выражается индексом согласованности.

ИС К1 = (λmax К1 - n)/(n - 1) = 0.004.

Найдено отношение согласованности ОС.

ОС К1 = ИС/СИ = 0.004.

Значение ОС меньше или равное 0.10 считается приемлемым, значит матрица К 1 (проходной балл) согласована.

Определены индекс согласованности и отношение согласованности для матрицы К 2 – количество бюджетных мест (Таблица 3.6.3).

Таблица 3.6.3 – Матрица сравнения по критерию 2.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| К2 | А1 | А2 | А3 | А4 | А5 | W3К2Y |
| А1 | 1 | 5 | 1/6 | 1/5 | 1/5 | 0.074 |
| А2 | 1/5 | 1 | 1/8 | 1/5 | 1/4 | 0.038 |
| А3 | 6 | 8 | 1 | 2 | 2 | 0.415 |
| А4 | 5 | 5 | 1/2 | 1 | 1/2 | 0.209 |
| А5 | 5 | 4 | 1/2 | 2 | 1 | 0.264 |

Определена сумма каждого столбца матрицы суждений.

S1К2 = 1 + 1/5 + 6 + 5 + 5 = 17.2;

S2 К2 = 5 + 1 + 8 + 5 + 4 = 23;

S3 К2 = 1/6 + 1/8 + 1 + 1/2 + 1/2 = 2.292;

S4 К2 = 1/5 + 1/5 + 2 + 1 + 2 = 5.4;

S5 К2 = 1/5 + 1/4 + 2 + 1/2 + 1 = 3.95.

Затем полученный результат умножен на компоненту нормализованного вектора приоритетов.

Р1 К2 = S1 х W3 К21 = 1.273;

Р2 К2 = S2 х W3 К22 = 0.874;

Р3 К2 = S3 х W3 К23 = 0.951;

Р4 К2 = S4 х W3 К24 = 1.129;

Р5 К2 = S5 х W3 К25 = 1.043.

Найдена пропорциональность предпочтений.

λmax К2 = Р1К2 + Р2К2 + Р3К2 + Р4К2 + Р5К2 = 5.27.

Отклонение от согласованности выражается индексом согласованности.

ИС К2 = (λmax К2 - n)/(n - 1) = (5.27 – 5) / (5 – 1) = 0.067.

Найдено отношение согласованности ОС.

ОС К2 = ИС/СИ = 0.06.

Значение ОС меньше или равное 0.10 считается приемлемым, значит матрица К 2 (количество бюджетных мест) согласована.

Определены индекс согласованности и отношение согласованности для матрицы К 3 – стоимость обучения (Таблица 3.6.4).

Таблица 3.6.4 – Матрица сравнения по критерию 3.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| К3 | А1 | А2 | А3 | А4 | А5 | W3К3Y |
| А1 | 1 | 1/8 | 4 | 4 | 1/4 | 0.107 |
| А2 | 8 | 1 | 9 | 8 | 4 | 0.578 |
| А3 | 1/4 | 1/9 | 1 | 1/2 | 1/5 | 0.038 |
| А4 | 1/4 | 1/8 | 2 | 1 | 1/4 | 0.053 |
| А5 | 4 | 1/4 | 5 | 4 | 1 | 0.224 |

Определена сумма каждого столбца матрицы суждений.

S1К3 = 1 + 8 + 1/4 + 1/4 + 4 = 13.5;

S2 К3 = 1/8 + 1 + 1/9 + 1/8 + 1/4 = 1.611;

S3 К3 = 4 + 9 + 1 + 2 + 5 = 21;

S4 К3 = 4 + 8 + 1/2 + 1 + 4 = 17.5;

S5 К3 = 1/4 + 4 + 1/5 + 1/4 + 1 = 5.7.

Затем полученный результат умножен на компоненту нормализованного вектора приоритетов.

Р1 К3 = S1 х W3 К31 = 1.444;

Р2 К3 = S2 х W3 К32 = 0.931;

Р3 К3 = S3 х W3 К33 = 0.798;

Р4 К3 = S4 х W3 К34 = 0.927;

Р5 К3 = S5 х W3 К35 = 1.277.

Найдем пропорциональность предпочтений.

λmax К3 = Р1К3 + Р2К3 + Р3К3 + Р4К3 + Р5К3 = 5.377.

Отклонение от согласованности выражается индексом согласованности.

ИС К3 = (λmax К3 - n)/(n - 1) = 0.094.

Найдено отношение согласованности ОС.

ОС К3 = ИС/СИ = 0.084.

Значение ОС меньше или равное 0.10 считается приемлемым, значит матрица К 3 (стоимость обучения) согласована.

Определены индекс согласованности и отношение согласованности для матрицы К 4 – рейтинг университета (Таблица 3.6.5).

Таблица 3.6.5 – Матрица сравнения по критерию 4.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| К4 | А1 | А2 | А3 | А4 | А5 | W3К4Y |
| А1 | 1 | 4 | 8 | 1/2 | 1 | 0.261 |
| А2 | 1/4 | 1 | 4 | 1/3 | 1/3 | 0.097 |
| А3 | 1/8 | 1/4 | 1 | 1/8 | 1/8 | 0.033 |
| А4 | 2 | 3 | 8 | 1 | 1 | 0.326 |
| А5 | 1 | 3 | 8 | 1 | 1 | 0.283 |

Определена сумма каждого столбца матрицы суждений.

S1К4 = 1 + 1/4 + 1/8 + 2 + 1 = 4.375;

S2К4 = 4 + 1 + 1/4 + 3 + 3 = 11.25;

S3К4 = 8 + 4 + 1 + 8 + 8 = 29;

S4К4 = 1/2 + 1/3 + 1/8 + 1 + 1 = 2.958;

S5К4 = 1 + 1/3 + 1/8 + 1 + 1 = 3.458.

Затем полученный результат умножен на компоненту нормализованного вектора приоритетов.

Р1К4 = S1 х W3 К41 = 1.142;

Р2К4 = S2 х W3 К42 = 1.091;

Р3К4 = S3 х W3 К43 = 0.957;

Р4К4 = S4 х W3 К44 = 0.964;

Р5К4 = S5 х W3 К45 = 0.979.

Найдена пропорциональность предпочтений.

λmax К4 = Р1К4 + Р2К4 + Р3К4 + Р4К4 + Р5К4 = 5.133.

Отклонение от согласованности выражается индексом согласованности.

ИС К4 = (λmax К4 - n)/(n - 1) = 0.033.

Найдено отношение согласованности ОС.

ОС К4 = ИС/СИ = 0.029.

Значение ОС меньше или равное 0,10 считается приемлемым, значит матрица К 4 (рейтинг университета) согласована.

Определены индекс согласованности и отношение согласованности для матрицы К 5 – расстояние до общежития (Таблица 3.6.6).

Таблица 3.6.6 – Матрица сравнения по критерию 5.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| К5 | А1 | А2 | А3 | А4 | А5 | W3К5Y |
| А1 | 1 | 5 | 8 | 9 | 2 | 0.47 |
| А2 | 1/5 | 1 | 4 | 5 | 1/3 | 0.134 |
| А3 | 1/8 | 1/4 | 1 | 2 | 1/8 | 0.048 |
| А4 | 1/9 | 1/5 | 1/2 | 1 | 1/8 | 0.034 |
| А5 | 1/2 | 3 | 8 | 8 | 1 | 0.314 |

Определена сумма каждого столбца матрицы суждений.

S1К5 = 1 + 1/5 + 1/8 + 1/9 + 1/2 = 1.936;

S2К5 = 5 + 1 + 1/4 + 1/5 + 3 = 9.45;

S3К5 = 8 + 4 + 1 + 1/2 + 8 = 21.5;

S4К5 = 9 + 5 + 2 + 1 + 8 = 25;

S5К5 = 2 + 1/3 + 1/8 + 1/8 + 1 = 3.583.

Затем полученный результат умножен на компоненту нормализованного вектора приоритетов.

Р1К5 = S1 х W3 К41 = 0.91;

Р2К5 = S2 х W3 К42 = 1.266;

Р3К5 = S3 х W3 К43 = 1.032;

Р4К5 = S1 х W3 К44 = 0.85;

Р5К5 = S1 х W3 К45 = 1.125.

Найдена пропорциональность предпочтений.

λmax К5 = Р1К5 + Р2К5 + Р3К5 + Р4К5 + Р5К5 = 5.183.

Отклонение от согласованности выражается индексом согласованности.

ИС К5 = (λmax К5 - n)/(n - 1) = 0.046.

Найдено отношение согласованности ОС.

ОС К5 = ИС/СИ = 0.041.

Значение ОС меньше или равное 0,10 считается приемлемым, значит матрица К 5 (расстояние до общежития) согласована.

3.7 Синтез альтернатив

Векторы приоритетов и отношения согласованности определяются для всех матриц суждений, начиная со второго уровня.

Для определения приоритетов альтернатив локальные приоритеты умножены на приоритет соответствующего критерия на высшем уровне и найдены суммы по каждому элементу в соответствии с критериями, на которые воздействует этот элемент.

W2i = (Y21=0.245; Y22=0.135; Y23=0.093; Y24=0.446; Y25=0.081);

W3К1Y = (Y311=0.195; Y312=0.279; Y313=0.04; Y314=0.408; Y315=0.078);

W3К2Y = (Y321=0.074; Y322=0.038; Y323=0.415; Y324=0.209; Y325=0.264);

W3К3Y = (Y331=0.107; Y332=0.578; Y333=0.038; Y334=0.053; Y335=0.224);

W3К4Y = (Y341=0.261; Y342=0.097; Y343=0.033; Y344=0.326; Y345=0.283);

W3К5Y = (Y351=0.47; Y352=0.134; Y353=0.048; Y354=0.034; Y355=0.314).

Приоритеты альтернатив получены следующим образом:

W1 = W21 х W3К11 + W22 х W3К21 + W23 х W3К31 + W24 х W3К41 + W25 х W3К51 = 0.222.

W2 = W21 х W3К12 + W22 х W3К22 + W23 х W3К32 + W24 х W3К42 + W25 х W3К52 = 0.181.

W3 = W21 х W3К13 + W22 х W3К23 + W23 х W3К33 + W24 х W3К43 + W25 х W3К53 = 0.088.

W4 = W21 х W3К14 + W22 х W3К24 + W23 х W3К34 + W24 х W3К44 + W25 х W3К54 = 0.281.

W5 = W21 х W3К15 + W22 х W3К25 + W23 х W3К35 + W24 х W3К45 + W25 х W3К55 = 0.227.

Таким образом, приоритеты альтернатив равны:

альтернатива А1 (МГТУ имени Н.Э. Баумана) - W1 приоритет равен = 0.222;

альтернатива А2 (ВШЭ)- W2 приоритет равен = 0.181;

альтернатива А3 (МАИ) - W3 приоритет равен = 0.088;

альтернатива А4 (НИЯУ МИФИ) – W4 приоритет равен = 0.281;

альтернатива А5 (МФТИ) - W5 приоритет равен = 0.227.

3.8 Вывод

Самой оптимальной является та альтернатива, приоритет которой максимален. Такой альтернативой является А4.

3.9 Результаты работы программы

Результаты работы программы, реализующей метод анализа иерархий, приведены на Рисунках 3.9.1 – 3.9.7.

Изображение выглядит как снимок экрана, черный

Автоматически созданное описание

Рисунок 3.9.1 – Матрица парного сравнения критериев

Изображение выглядит как снимок экрана, текст, Шрифт

Автоматически созданное описание

Рисунок 3.9.2 – Сгенерированная матрица для первого критерия

Изображение выглядит как снимок экрана, черный

Автоматически созданное описание

Рисунок 3.9.3 – Сгенерированная матрица для второго критерия

Изображение выглядит как снимок экрана, текст, Шрифт

Автоматически созданное описание

Рисунок 3.9.4 – Сгенерированная матрица для третьего критерия

Изображение выглядит как снимок экрана, текст, черный

Автоматически созданное описание

Рисунок 3.9.5 – Сгенерированная матрица для четвёртого критерия

Изображение выглядит как снимок экрана, текст

Автоматически созданное описание

Рисунок 3.9.6 – Сгенерированная матрица для пятого критерия

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 3.9.7 – Результат работы программы

3.10 Заключение

В ходе данной работы мной был изучен метод анализа иерархий, проведён его ручной расчёт для 5 критериев и 5 альтернатив. Преимуществом метода является гарантированное получение единственного оптимального решения, а недостатком является требование соблюдать согласованность матриц приоритетов, из-за чего необходимо проводить повторные расчёты в случае, если матрица не согласована.

4 ГРАФИЧЕСКИЙ МЕТОД

4.1 Введение

Линейное программирование – это метод решения оптимизационных задач, основанный на линейной модели. Графический метод - один из способов решения таких задач, который позволяет визуально представить ограничения и целевую функцию на графике.

Сначала на графике отображаются все уравнения модели. Затем накладывают ограничения модели, чтобы найди область допустимых решений. Областью допустимых решений является пересечение всех ограничений модели. После этого находится вектор градиента функции модели, направление которого означает рост функции. Для нахождения максимума функции необходимо параллельно перемещать ее линию графика в направлении вектора градиента до тех пор, пока линия пересекает ОДР. Последняя точка пересечения будет обозначать максимум функции. Подставив ее координаты в качестве аргументов функции модели можно найти максимальное значение.

4.2 Постановка задачи

Решить задачу линейного программирования с двумя переменными графическим методом.

4.3 Данные индивидуального варианта

## 4.4 Подготовка данных

В среде Microsoft Excel добавим 4 столбца:

1. – значения от 0 до 10 с шагом 0,5;
2. – значения ограничения ;
3. – значения ограничения ;
4. – значения ограничения ;
5. – значения целевой функции при условии .

Таблица 4.4.1 – Данные для графика

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| x1 |  |  |  |  |
| 0 | -3,50 | 2,50 | 6,00 | 0,00 |
| 0,5 | -2,25 | 2,75 | 5,50 | 0,00 |
| 1 | -1,00 | 3,00 | 5,00 | 0,00 |
| 1,5 | 0,25 | 3,25 | 4,50 | 0,00 |
| 2 | 1,50 | 3,50 | 4,00 | 0,00 |
| 2,5 | 2,75 | 3,75 | 3,50 | 0,00 |
| 3 | 4,00 | 4,00 | 3,00 | 0,00 |
| 3,5 | 5,25 | 4,25 | 2,50 | 0,00 |
| 4 | 6,50 | 4,50 | 2,00 | 0,00 |
| 4,5 | 7,75 | 4,75 | 1,50 | 0,00 |
| 5 | 9,00 | 5,00 | 1,00 | 0,00 |
| 5,50 | 10,25 | 5,25 | 0,50 | 0,00 |
| 6,00 | 11,50 | 5,50 | 0,00 | 0,00 |
| 6,50 | 12,75 | 5,75 | -0,50 | 0,00 |
| 7,00 | 14,00 | 6,00 | -1,00 | 0,00 |
| 7,50 | 15,25 | 6,25 | -1,50 | 0,00 |
| 8,00 | 16,50 | 6,50 | -2,00 | 0,00 |
| 8,50 | 17,75 | 6,75 | -2,50 | 0,00 |
| 9,00 | 19,00 | 7,00 | -3,00 | 0,00 |
| 9,50 | 20,25 | 7,25 | -3,50 | 0,00 |
| 10,00 | 21,50 | 7,50 | -4,00 | 0,00 |

4.5 Построение графика

Выделим таблицу подготовленных данных и построим гладкий график. Произведем настройку шага координатной оси x1 и получим следующий график (Рисунок 4.5.1).

Изображение выглядит как диаграмма, линия, График, текст

Автоматически созданное описание

Рисунок 4.5.1 – Построение графиков по данным

4.6 Выделение области допустимых решений

Чтобы определить форму ОДР надо рассмотреть каждую из построенных прямых по отдельности и, заменив мысленно в соответствующем уравнении знак равенства на исходное неравенство, определить, с какой стороны от рассматриваемой прямой лежит ОДР. Для этого необходимо решить соответствующее неравенство относительно точки (0,0). Если неравенство истинно, то ОДР лежит в полуплоскости, которой принадлежит точка (0,0), если ложно – то в полуплоскости, которая не содержит точку (0,0). ОДР будет являться областью пересечения всех полуплоскостей, задаваемых неравенствами-ограничителями.

В результате получим область допустимых решений, представленную на Рисунке 4.6.1.

Изображение выглядит как линия, График, диаграмма

Автоматически созданное описаниеРисунок 4.6.1 – Выделение области допустимых решений

4.7 Максимум функции

Для нахождения максимума функции найдем её градиент по формуле 4.1:

(4.1)

Для нахождения минимума функции найдем её градиент по формуле 4.1:

(4.2)

Градиент функции будет равен {-1,2}, а антиградиент функции будет равен {1,-2}. Изобразим эти вектора на графике (Рисунок 4.7.1).

Теперь начинаем мысленно сдвигать прямую целевой функции в направлении градиента, и определяем последнюю точку ОДР, которая лежит на пути прямой. На пути прямой лежит отрезок с началом в точке () и конце в точке (3;4).

Изображение выглядит как линия, График, диаграмма, текст

Автоматически созданное описаниеРисунок 4.7.1 – Точка максимума функции

Найдем значение функции в точке максимума.

Подставив координаты найденных точек (максимума) в систему уравнения и убедимся, что точки принадлежать к области ОДР:

Получим значение равное F(x)max = -3 + 2\*4 = 5. Убедимся, что для любой точки, принадлежащей отрезку, максимальное значение функции остаётся неизменным. Подставим координаты начала отрезка: F(x)max = .

## 4.8 Минимум функции

Для нахождения минимума функции будем перемещать прямую в сторону антиградиента. Отметим на графике найденную точку (Рисунок 4.8.1).

Изображение выглядит как линия, диаграмма, График, Параллельный

Автоматически созданное описание

Рисунок 4.8.1 – Точка минимума функции

Найдем координаты точки минимума:

В результате получим точку с координатами (). Найдем значение функции в этой точке.

Подставив координаты найденных точек (минимума) в систему уравнения и убедимся, что точки принадлежать к области ОДР:

Получим результат F(x)min =

Ответ:

F(x)max = 5.

F(x)min = .

4.9 Заключение

В данной работе был подробно рассмотрен графический метод решения задачи линейного программирования. Он очень прост в реализации, а также показывает наглядное решение. Тем не менее, при количестве параметров больше двух графическая реализация метода значительно усложняется.

5 СИМПЛЕКСНЫЙ МЕТОД

5.1 Введение

Симплексный метод – это из методов решения задач линейного программирования. Его суть заключается в сокращении перебора вершин многогранника условий. Если исследуемая вершина не отвечает условию максимума/минимума, то происходит переход к вершине, повышая/понижая значимость целевой функции. Это делается с помощью введения оценок, которые должны быть положительными, чтобы был достигнут максимум функции, и отрицательными, чтобы был достигнут минимум. С помощью этого количество перебираемых вершин существенно снижается.

5.2 Постановка задачи

**Вариант №13**

Задание. Решить прямую ЗЛП с помощью симплексного метода и обратную с помощью теорем двойственности. Определить интервалы устойчивости.

Задача. В кондитерском цехе выпускают печенье двух сортов. В таблице 5.2.1 указан расход продуктов для каждого сорта и количество имеющихся продуктов.

Таблица 5.2.1. Исходные данные задачи

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Сорт | Масло | Яйца | Сахар | Молоко | Цена за 1 кг, ден. ед. |
| 1-й сорт | 0,2 | 0,75 | 0,15 | 0,15 | 1,4 |
| 2-й сорт | 0,1 | 0,20 | 0,20 | 0,25 | 0,9 |
| Запасы продуктов | 100 | 150 | 100 | 150 |

Определить, какое общее количество печенья каждого сорта надо выпекать, чтобы общая стоимость была наибольшей.

5.3 Математическая модель задачи

Пусть х1 – количество печенья первого сорта, х2 – количество печенья второго сорта. Прибыль от продажи печенья составит 1.4x1 + 0.9x2, прибыль требуется максимизировать.

Ограничения задачи:

Таким образом, переходим к задаче линейного программирования:

Приведем задачу к канонической форме. Для этого в левые части ограничений вводим дополнительные переменные: х3 ≥ 0. Эти переменные выбираются так, чтобы они обращали неравенства в равенства.

5.4 Решение задачи

Построим начальную симплекс-таблицу. Запишем систему в векторной форме:

Векторы 𝐴3, 𝐴4, 𝐴5, A6 являются линейно независимыми единичными векторами 4-мерного пространства и образуют базис этого пространства.

Поэтому за базисные переменные выбираем переменные 𝑥3, 𝑥4, 𝑥5, x6. Небазисными переменными являются 𝑥1, 𝑥2. Разложение позволяет найти первое базисное допустимое решение.

Для этого свободные переменные 𝑥1, 𝑥2 приравниваем нулю. В результате получим разложение

Которому соответствует первоначальный опорный план

Для проверки плана 𝑥(0) на оптимальность построим первую симплекс-таблицу. Введем в рассмотрение вектор коэффициентов целевой функции при базисных переменных.

В левый столбец Таблицы 5.4.1 запишем переменные 𝑥3, 𝑥4, x5, x6, образующие базис, в верхней строке – небазисные переменные 𝑥1, 𝑥2. В строке 𝑐j запишем коэффициенты целевой функции, соответствующие небазисным переменным с1 = 1.4, с2 = 0.9. В столбце запишем коэффициенты целевой функции, соответствующие базисным переменным Столбец, определяемый переменной 𝑥1, состоит из коэффициентов вектора . Аналогично, столбец, определяемый переменной 𝑥2, состоит из коэффициентов вектора . Крайний правый столбец заполняется элементами столбца , в нем же в результате вычислений получаем оптимальный план.

Заполнение f-строки (Таблица 5.4.2). Найдем относительные оценки ∆1, ∆2 и значение целевой функции 𝑄.

Таблица 5.4.1 – Начальная симплекс-таблица задачи о максимальном доходе

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | сj | 1.4 | 0.9 |  |
|  |  | X1 | X2 |  |
| 0 | X3 | 0.2 | 0.1 | 100 |
| 0 | X4 | 0.75 | 0.2 | 150 |
| 0 | X5 | 0.15 | 0.2 | 100 |
| 0 | X6 | 0.15 | 0.25 | 150 |
|  | f |  |  |  |
|  |  | Δ1 | Δ2 | Q |

Таблица 5.4.2 – Заполнение f-строки

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | сj | 1.4 | 0.9 |  |  |
|  |  | X1 | X2 |  |  |
| 0 | X3 | 0.2 | 0.1 | 100 | 100 / 0.2 = 500 |
| 0 | X4 | 0.75 | 0.2 | 150 | 150 / 0.75 = 200 min |
| 0 | X5 | 0.15 | 0.2 | 100 | 100 / 0.15 = 666.66 |
| 0 | X6 | 0.15 | 0.25 | 150 | 150 / 0.15 = 1000 |
|  | f | -1.4 | -0.9 | 0 |  |
|  |  | Δ1 | Δ2 | Q |  |

Для оптимальности опорного решения в задаче на максимум требуется выполнение неотрицательности всех относительных оценок ∆i ≥ 0. Так как оценки ∆1= −1.4, ∆2= −0.9 в f-строке отрицательны, то это свидетельствуют о возможности улучшения полученного решения.

Наибольшая по модулю отрицательная оценка ∆1 = −1.4. В базис будет включена соответствующая ей небазисная переменная 𝑥1. Составим отношения свободных членов к положительным элементам разрешающего столбца. Данные отношения приведены справа от таблицы. Наименьшему частному соответствует строка с переменной 𝑥4. Эта переменная исключается из базиса. В Таблице 5.4.2 разрешающий столбец и разрешающая строка выделены. Разрешающим элементом является число 𝑎21 = 0.75.

Далее построим новую симплекс-таблицу. Ниже поэтапно демонстрируется процесс заполнения новой симплекс-таблицы (Таблицы 5.4.3).

Таблица 5.4.3 – Новая симплекс-таблица

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | сj | 0 | 0.9 |  |
|  |  | X4 | X2 |  |
| 0 | X3 |  |  |  |
| 1.4 | X1 | 4/3 |  |  |
| 0 | X5 |  |  |  |
| 0 | X6 |  |  |  |
|  | f |  |  |  |
|  |  | Δ1 | Δ2 | Q |

В Таблице 5.4.3 переменные x1 и 𝑥4 меняются местами вместе с коэффициентами 𝑐𝑗. Разрешающий элемент заменяется на обратный. В Таблице 5.4.4 элементы разрешающей строки делятся на разрешающий элемент. Элементы разрешающего столбца делятся на разрешающий элемент и меняют знак.

Таблица 5.4.4 – Симплекс преобразования

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | сj | 0 | 0.9 |  |
|  |  | X4 | X2 |  |
| 0 | X3 | -4/15 |  |  |
| 1.4 | X1 | 4/3 | 4/15 | 200 |
| 0 | X5 | -0.2 |  |  |
| 0 | X6 | -0.2 |  |  |
|  | f | 28/15 |  |  |
|  |  | Δ1 | Δ2 | Q |

Таблица 5.4.5 – Итерация 0

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | сj | 0 | 0.9 |  |  |
|  |  | X4 | X2 |  |  |
| 0 | X3 | -4/15 | 7/150 | 60 | 60 / (7/150) = 1285.7 |
| 1.4 | X1 | 4/3 | 4/15 | 200 | 200 / (4/15) = 750 |
| 0 | X5 | -0.2 | 4/25 | 70 | 70 / (4/25) = 437.5 min |
| 0 | X6 | -0.2 | 21/100 | 120 | 120 / (21/100) = 571.0 = 571.4 |
|  | f | 28/15 | -79/150 | 280 |  |

Остальные элементы (Таблица 5.4.5) рассчитываются по «правилу прямоугольника».

Базисное решение, которое дает последняя таблица

Это решение не является оптимальным, так как в f-строке имеется отрицательная оценка ∆2. Наибольшая по модулю отрицательная оценка ∆2 = . В базис будет включена соответствующая ей небазисная переменная 𝑥2. Составим отношения свободных членов к положительным элементам разрешающего столбца. Наименьшему частному соответствует строка с переменной 𝑥5, которая исключается из базиса. В Таблице 5.4.5 разрешающий столбец и разрешающая строка выделены. Разрешающим элементом является число 𝑎32 = .

Продемонстрируем процесс заполнения новой симплекс-таблицы (Таблицы 5.4.6).

Таблица 5.4.6 – Новая симплекс-таблица

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | сj | 0 | 0 |  |
|  |  | X4 | X5 |  |
| 0 | X3 |  |  |  |
| 1.4 | X1 |  |  |  |
| 0.9 | X2 |  | 25/4 |  |
| 0 | X6 |  |  |  |
|  | f |  |  |  |
|  |  | Δ1 | Δ2 | Q |

В Таблице 5.4.6 переменные 𝑥2 и 𝑥5 меняются местами вместе с коэффициентами 𝑐𝑗. Разрешающий элемент заменяется на обратный. В Таблице 5.4.7 элементы разрешающей строки делятся на разрешающий элемент. Элементы разрешающего столбца делятся на разрешающий элемент и меняют знак.

Таблица 5.4.7 – Симплекс преобразования

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | сj | 0 | 0 |  |
|  |  | X4 | X5 |  |
| 0 | X3 |  | -7/24 |  |
| 1.4 | X1 |  | -5/3 |  |
| 0.9 | X2 | -5/4 | 25/4 | 437.5 |
| 0 | X6 |  | -21/16 |  |
|  | f |  | 79/24 |  |
|  |  | Δ1 | Δ2 | Q |

Таблица 5.4.8 – Итерация 1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | сj | 0 | 0 |  |
|  |  | X4 | X5 |  |
| 0 | X3 | -5/24 | -7/24 | 475/12 |
| 1.4 | X1 | 5/3 | -5/3 | 250/3 |
| 0.9 | X2 | -5/4 | 25/4 | 437.5 |
| 0 | X6 | 1/16 | -21/16 | 225/8 |
|  | f | 29/24 | 79/24 | 6125/12 |
|  |  | Δ1 | Δ2 | Q |

Остальные элементы (Таблица 5.4.8) рассчитываются по «правилу прямоугольника».

Если в последней таблице f-строке не содержит отрицательных оценок, то это свидетельствует об оптимальности полученного решения:

Базисное решение, которое дает последняя таблица

Проверим решение по «правилу прямоугольника».

Таким образом, кондитерский цех должен выпекать 𝑥1 = кг печенья первого сорта и кг печенья второго сорта. Тогда общая стоимость будет наибольшей и кондитерская получит прибыль от продажи 510.417 [ден.ед].

5.5 Пример работы программы

Результаты выполнения программы, реализующей симплексный метод, представлены на Рисунках 5.5.1 – 5.5.3.

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Рисунок 5.5.1 – Условия задачи в csv файле

Изображение выглядит как текст, Шрифт, снимок экрана

Автоматически созданное описание

Рисунок 5.5.2 – Обработка входных данных

Изображение выглядит как текст, снимок экрана, Мультимедийное программное обеспечение, Графическое программное обеспечение

Автоматически созданное описание

Рисунок 5.5.3 – Выполнение двух итераций и найденное решение

5.6 Заключение

В ходе данной работы мной был изучен симплекс-метод, произведён его ручной расчёт для решения поставленной задачи линейного программирования, а также была разработана программа на языке Python для решения задач симплекс-методом.

Плюсом метода является его универсальность, т.к. можно решать задачи линейного программирования для любого числа переменных и ограничений, однако в определённых условиях метод может уйти в полный перебор вершин области допустимых решений, что приведёт к очень долгому времени поиска решения.

6 ДВОЙСТВЕННАЯ ЗАДАЧА

6.1 Введение

Обычно с задачей линейного программирования (ЗЛП) связана другая линейная задача, называемая двойственной. Тогда первоначальная задача называется исходной или прямой. Математические модели двойственных задач могут быть симметричными или несимметричными. В симметричных задачах система ограничений как исходной, так и двойственной задачи задается неравенствами, причем на двойственные переменные налагается условие не отрицательности. В несимметричных двойственных задачах система ограничений исходной задачи задается в виде равенств, а в двойственной – в виде неравенств, причем в последней переменные могут быть и отрицательными.

6.2 Постановка задачи

**Вариант №13**

Задание. Решить прямую ЗЛП с помощью симплексного метода и обратную с помощью теорем двойственности. Определить интервалы устойчивости.

Задача. В кондитерском цехе выпускают печенье двух сортов. В таблице 5.2.1 указан расход продуктов для каждого сорта и количество имеющихся продуктов.

Таблица 5.2.1. Исходные данные задачи

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Сорт | Масло | Яйца | Сахар | Молоко | Цена за 1 кг, ден. ед. |
| 1-й сорт | 0,2 | 0,75 | 0,15 | 0,15 | 1,4 |
| 2-й сорт | 0,1 | 0,20 | 0,20 | 0,25 | 0,9 |
| Запасы продуктов | 100 | 150 | 100 | 150 |

Определить, какое общее количество печенья каждого сорта надо выпекать, чтобы общая стоимость была наибольшей.

6.3 Математическая модель

Пусть х1 – количество печенья первого сорта, х2 – количество печенья второго сорта. Прибыль от продажи печенья составит 1.4x1 + 0.9x2, прибыль требуется максимизировать.

Ограничения задачи:

Таким образом, переходим к задаче линейного программирования:

6.4 Соответствующая исходной двойственная задача

Найдем соответствующую двойственную задачу. Введем вектор двойственных переменных размерности два . Соответствующие векторы и матрица ограничений имеет вид:

.

Запишем двойственную задачу. Найти минимум функции.

При ограничениях:

6.5 Первая теорема двойственности

Если одна из пары двойственных задач имеет оптимальный план, то и другая имеет оптимальный план, причем экстремальные значения целевых функций равны. В ходе решения прямой задачи было определено, что максимальный доход от продажи составляет тыс. ден. ед., оптимальный план

Оптимальное решение двойственной задачи может быть получено из оптимального решения прямой задачи. Так как прямая задача имеет решение, то на основании первой теоремы о двойственности задача также разрешима. Ее решение может быть найдено из формулы:

Где D – матрица, составленная из компонентов векторов входящих в последний базис, при котором получен оптимальный план исходной задачи.

В последней симплекс-таблице базисными переменными являются 𝑥3, 𝑥1, 𝑥2, 𝑥6. Соответствующие этим переменным векторы , в разложении используются для формирования столбцов матрицы D.

Тогда,

Для вычисления обратной матрицы 𝐷-1 запишем матрицу 𝐷 дописав к ней справа единичную матрицу.

Для нахождения обратной матрицы 𝐷-1 используем элементарные преобразования над строками матрицы. Таким образом, преобразуются левая часть полученной матрицы в единичную.

Разделим вторую строку на 0.75;

от первой строки отнимем вторую строку, умноженную на 0.2; от третьей строки отнимем вторую строку, умноженную на 0.15, от четвертой строки отнимем вторую строку, умноженную на 0.15;

разделим третью строку на 0.16;

от первой строки отнимем третью строку, умноженную на ; от второй строки отнимем третью, умноженную на ; от четвертой строки отнимем третью строку, умноженную на 0.21;

Запишем обратную матрицу.

Базисными переменными в симплекс-таблице являются , тогда

При этом минимальное значение целевой функции двойственной задачи

совпадает с максимальным значением 𝑓𝑚𝑎𝑥 = [тыс. ден.ед.] прямой задачи, что является результатом взаимодвойственности. Таким образом,

## 6.6 Вторая теорема двойственности

Для того, чтобы планы и ЗЛП двойственной пары были оптимальными, необходимо и достаточно, чтобы эти планы удовлетворяли условиям дополняющей нежесткости.

Итак, имеем оптимальное решение прямой задачи: объем производства печенья первого сорта – 𝑥1 = ; объем производства печенья второго сорта – 𝑥2 = 437.5; максимальный доход от продажи 𝑓𝑚𝑎𝑥 = [тыс. ден.ед.]. Рассмотрим выполнение неравенств прямой задачи при подстановке 𝑥1, 𝑥2 в систему ограничений (Таблица 6.6.1).

Согласно Таблице 6.6.1 имеем следующую систему уравнений:

Решим данную систему уравнений

Решение, найденное из первой теоремы двойственности равнозначно решению из второй теоремы.

Таким образом, вторая теорема дает нахождение оптимального решения двойственной задачи, пользуясь условием обращения в равенство сопряженных неравенств в системах ограничения.

Таблица 6.6.1 – Выполнение неравенств прямой задачи

|  |  |  |
| --- | --- | --- |
| Ограничение | Расчет | Вывод |
| 0.2х1 + 0.1х2 ≤ 100 | 0.2\* + 0.1\*437.5 <100  60 < 100 | Первое ограничение прямой задачи выполняется как строгое неравенство, остается спрос на печенье первого сорта. Значит, этот ресурс не является дефицитным и его оценка в оптимальном плане равна нулю (𝑦1 = 0). |
| 0.75х1 + 0.2х2 ≤ 150 | 0.75\*+0.2\*437.5 = 150  150=150 | Второе ограничение прямой задачи выполняется как равенство. Это означает, что печенье первого сорта полностью используется в оптимальном плане, является дефицитным и его оценка согласно второй теоремы двойственности отлична от нуля (𝑦2 ≠ 0). |
| 0.15х1 + 0.2х2 ≤ 100 | 0.15\*+0.2\*437.5 = 100  100=100 | Третье ограничение прямой задачи выполняется как равенство. Это означает что печенье второго сорта полностью используется в оптимальном плане, является дефицитным и его оценка согласно второй теоремы двойственности отлична от нуля (𝑦3 ≠ 0). |
| 0.15х1 + 0.25х2 ≤ 150 | 0.15\*+0.25\*437.5<150  121 < 150 | Четвёртое ограничение прямой задачи выполняется как строгое неравенство, остается спрос на печенье второго сорта. Значит, этот ресурс не является дефицитным и его оценка в оптимальном плане равна нулю (𝑦4 = 0). |
| х1 ≥ 0 | > 0 | Первое ограничение в двойственной задаче будет равенством 0.2y1 + 0.75y2 + 0.15y3 + 0.15y4 = 1.4 |
| х2 ≥ 0 | 437.5 > 0 | Второе ограничение в двойственной задаче будет равенством 0.1y1 + 0.2y2 + 0.2y3 + 0.25y4 = 0.9 |

## 6.7 Третья теорема двойственности

Третью теорему двойственности иногда называют теоремой об оценках. Рассматривая ограничения ЗЛП, можно констатировать: изменение правых частей ограничений исходной задачи приводит к изменению максимального значения целевой функции 𝑍𝑚𝑎𝑥.

Выпишем необходимые элементы из прямой задачи о максимальном доходе. Обратная матрица базиса оптимального плана:

Индексы базисных переменных оптимального плана:

Свободные члены неравенств (ограничений) прямой задачи:

Теперь воспользуемся формулами для нахождения нижней и верхней границ интервалов устойчивости оценок по видам ресурсов.

*Ресурс 1 (Масло)*. Найдем нижнюю границу. В четвёртом столбце обратной матрицы один положительный элемент (1), ему соответствует индекс базисной переменной оптимального плана ().

Найдем верхнюю границу. Среди элементов четвёртого столбца отсутствуют отрицательные элементы.

Таким образом, получаем

Тогда первый ресурс может изменяться в интервале:

При таком значении оптимальный план двойственной задачи остается неизменным. Аналогичные рассуждения позволяют найти интервалы устойчивости оценок для остальных ресурсов.

*Ресурс 2 (Яйца).* Рассматриваем третий столбец обратной матрицы, в котором один положительный элемент () и три отрицательных (, , ). Данным элементам соответствуют следующие индексы базисных переменных оптимального плана: положительного элемента – ; для отрицательных – ,150.

Тогда находим нижнюю границу.

Найдем верхнюю границу.

Выбираем наибольшее значение, равное .

Получаем

Тогда второй ресурс может изменяться в интервале:

*Ресурс 3 (Сахар).* Рассматриваем второй столбец обратной матрицы, в котором два положительных элемента (, ). Данным элементам соответствуют индексы соответствующего базисного переменного оптимального плана – 150, 150.

Находим нижнюю границу.

Выбираем наименьшее значение, равное 90.

Найдем верхнюю границу.

Выбираем наибольшее значение, равное .

Тогда, получаем что

Получаем интервал устойчивости оценок по отношению к третьему ограничению:

*Ресурс 4 (Молоко)*. Найдем нижнюю границу. В первом столбце обратной матрицы один положительный элемент (1), ему соответствует индекс базисной переменной оптимального плана ().

Найдем верхнюю границу. Среди элементов четвёртого столбца отсутствуют отрицательные элементы.

Таким образом, получаем

Тогда первый ресурс может изменяться в интервале:

Далее оценим влияние изменения объема ресурсов на величину максимальной стоимости продукции. Как известно, это дефицитные ресурсы  *).* Введем верхние границы в формулу:

Совместное влияние изменений этих ресурсов приводит к изменению максимальной стоимости продукции 𝐺𝑚𝑎𝑥 на величину:

Следовательно, оптимальное значение целевой функции при максимальном изменении ресурсов:

Таким образом, двойственные оценки позволяют судить о чувствительности решения к изменениям.

6.8 Результаты работы программы

Результаты выполнения программы, реализующей три теоремы двойственности, представлены на Рисунке 6.8.1.

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 6.8.1 –Теоремы двойственности

6.9 Заключение

Каждая из задач двойственной пары фактически является самостоятельной задачей линейного программирования и может быть решена не зависимо от другой. Связь задач заключается в том, что решение одной из них может быть получено непосредственно из решения другой. Взаимная симметрия прямой и двойственной задач определяет существование определенного соответствия между их оптимальными решениями. Эти соответствия устанавливают теоремы двойственности.

7 ТРАНСПОРТНАЯ ЗАДАЧА

7.1 Введение

Транспортная задача – это частный случай задачи линейного программирования, где ограничениями являются запасы поставщиков и потребности потребителей, а целевая функция – это стоимость перевозок. Функцию нужно минимизировать.

Транспортную задачу можно решать более простыми методами, чем симплекс-метод. Для этого нужно составить опорный план, используя метод северо-западного угла или метод минимальной стоимости. После построения опорного плана применяется метод потенциалов, который позволяет построить оптимальный план проще, чем симплекс-метод.

Транспортные задачи бывают закрытые и открытые. Закрытой задача является, если сумма количества ресурсов у поставщиков равна сумме потребностей у поставщиков, и открытой в противном случае. В данной работе рассматривается решение закрытой транспортной задачи.

7.2 Постановка задачи

**Вариант 13**.

Задача. Имеются поставщики и потребители, у которых известны запасы и потребности, соответственно, а также известны стоимости перевозки от каждого поставщика к каждому потребителю. Данные занесены в Таблицу 7.2.1.

Таблица 7.2.1. Исходные данные задачи.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Потребители  Поставщики | 40 | 30 | 30 | 50 |
| 60 | 2 | 3 | 5 | 1 |
| 70 | 3 | 4 | 9 | 4 |
| 20 | 2 | 5 | 2 | 5 |

Определить оптимальный план перевозок.

7.3 Математическая модель транспортной задачи

Математически задачу можно сформулировать следующим образом. Определить переменные , которые минимизируют суммарную стоимость перевозок.

и удовлетворяют системе ограничений

1. – с каждого пункта отправления груз должен быть вывезен полностью;
2. – потребитель должен получить ровно столько, сколько ему требуется;

Транспортная задача является закрытой, т.к. и .

7.4 Метод северо-западного угла

Заполним ячейку . Т.к. потребности меньше запасов (40 < 60), то . Запасы первого поставщика не исчерпаны, поэтому происходит переход к ячейке . Т.к. потребности больше запасов (30 > 20), то Запасы первого поставщика исчерпаны, происходит переход к ячейки . В составе заявки пункта остались неудовлетворёнными 10 единиц. Эти 10 единиц покроем за счёт пункта . Далее перейдём к ячейки . Запасов второго поставщика достаточно, значит . Запасы третьего потребителя удовлетворены, значит переходим к ячейки . Так как потребность больше запасов , (50 > 30) . Из запасов пункта выделим все доступные 20 единиц, чтобы удовлетворить запрос пункта .

Полученное решение является опорным решением транспортной задачи. Общая стоимость перевозок составляет:

(единиц).

Все поставки распределены, количество базисных клеток равно 6 = m + n – 1, значит, план невырожденный (Таблица 7.4.1).

Таблица 7.4.1 – Метод северо-западного угла

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Пункты | B1 | B2 | B3 | B4 | Запасы |
| A1 | |  |  | | --- | --- | |  | 2 | | 40 |  | | |  |  | | --- | --- | |  | 3 | | 20 |  | | |  |  | | --- | --- | |  | 5 | |  |  | | |  |  | | --- | --- | |  | 1 | |  |  | | 60 |
| A2 | |  |  | | --- | --- | |  | 3 | |  |  | | |  |  | | --- | --- | |  | 4 | | 10 |  | | |  |  | | --- | --- | |  | 9 | | 30 |  | | |  |  | | --- | --- | |  | 4 | | 30 |  | | 70 |
| A3 | |  |  | | --- | --- | |  | 2 | |  |  | | |  |  | | --- | --- | |  | 5 | |  |  | | |  |  | | --- | --- | |  | 2 | |  |  | | |  |  | | --- | --- | |  | 5 | | 20 |  | | 20 |
| Потребности | 40 | 30 | 30 | 50 | 150 |

7.5 Метод минимальной стоимости

Выбираем ячейку с минимальной стоимостью . Значение определяется как минимальное из остатков запасов и потребностей потребителя . Тогда . Мысленно вычёркиваем из таблицы столбец , так как его запрос удовлетворён. Перейдём к ячейки со стоимостью перевозки, равной 2. В рассматриваемую клетку запишем минимальное из остатков запасов и потребностей потребителя . В данном случае запишем 10 – остаток запасов в пункте , после чего мысленно вычеркнем строку . Далее в ячейку запишем минимум из потребностей (30) и запасов (20) - 20 и вычеркнем строку . Следующая минимальная стоимость находится в ячейке . Сюда запишем 10 и вычеркнем столбец . В ячейку запишем 30, так как потребность (30) меньше, чем запас (60). В последнюю оставшуюся ячейку запишем 30 и зачеркнём соответствующий столбец и строку . Все ресурсы израсходованы, а потребности удовлетворены.

Общая стоимость перевозок груза составляет:

(единиц).

Опорный план, составленный способом минимальной стоимости, более близок к оптимальному решению (Таблица 7.5.1), однако решение все ещё не является самым оптимальным.

Таблица 7.5.1 – Метод минимальной стоимости

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Пункты | B1 | B2 | B3 | B4 | Запасы |
| A1 | |  |  | | --- | --- | |  | 2 | | 10 |  | | |  |  | | --- | --- | |  | 3 | |  |  | | |  |  | | --- | --- | |  | 5 | |  |  | | |  |  | | --- | --- | |  | 1 | | 50 |  | | 60 |
| A2 | |  |  | | --- | --- | |  | 3 | | 10 |  | | |  |  | | --- | --- | |  | 4 | | 30 |  | | |  |  | | --- | --- | |  | 9 | | 30 |  | | |  |  | | --- | --- | |  | 4 | |  |  | | 70 |
| A3 | |  |  | | --- | --- | |  | 2 | | 20 |  | | |  |  | | --- | --- | |  | 5 | |  |  | | |  |  | | --- | --- | |  | 2 | |  |  | | |  |  | | --- | --- | |  | 5 | |  |  | | 20 |
| Потребности | 40 | 30 | 30 | 50 | 150 |

7.6 Метод потенциалов

Для определения исходного плана перевозок воспользуемся методом северо-западного угла. Согласно уже проведённым расчётам, исходный план представлен в Таблице 7.6.1. Общее число базисных клеток: m + n – 1 = 3 + 4 – 1 = 6.

Таблица 7.6.1 – Метод северо-западного угла

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Пункты | B1 | B2 | B3 | B4 | Запасы |
| A1 | |  |  | | --- | --- | |  | 2 | | 40 |  | | |  |  | | --- | --- | |  | 3 | | 20 |  | | |  |  | | --- | --- | |  | 5 | |  |  | | |  |  | | --- | --- | |  | 1 | |  |  | | 60 |
| A2 | |  |  | | --- | --- | |  | 3 | |  |  | | |  |  | | --- | --- | |  | 4 | | 10 |  | | |  |  | | --- | --- | |  | 9 | | 30 |  | | |  |  | | --- | --- | |  | 4 | | 30 |  | | 70 |
| A3 | |  |  | | --- | --- | |  | 2 | |  |  | | |  |  | | --- | --- | |  | 5 | |  |  | | |  |  | | --- | --- | |  | 2 | |  |  | | |  |  | | --- | --- | |  | 5 | | 20 |  | | 20 |
| Потребности | 40 | 30 | 30 | 50 | 150 |

Стоимость перевозок по этому плану:

(единиц).

Вычислим потенциалы и исходя из базисных переменных. Для их нахождения используем условие .

Считая , имеем .

Для каждой свободной клетки вычислим относительные оценки:

Условие оптимальности плана перевозок не выполняется, поэтому построим замкнутый цикл пересчета и определим величины перераспределения груза.

Минимальной оценкой является для клетки (3,3).

Для определения количества груза подлежащего распределению, построим замкнутый цикл (указан стрелками) (Таблица 7.6.2). Одна из вершин цикла находится в незанятой клетке , которую отмечаем знаком . Все остальные вершины цикла находятся в базисных клетках, с чередующимися знаками и . Найдем , равное наименьшему из чисел, стоящих в отрицательных вершинах цикла. Значение записываем в незанятую клетку. Двигаясь далее по означенному циклу, вычитаем из объемов перевозок, расположенных в клетках, которые обозначены знаком , и прибавляем к объемам перевозок, находящихся в клетках, отмеченных знаком . Элементы таблицы, не входящие в цикл, остаются без изменений. Таблица нового плана представлена в Таблице 7.6.3.

Таблица 7.6.2 – Таблица перерасчёта

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  | v1=2 | v2=3 | v3=8 | v4=3 |  |
|  | Пункты | B1 | B2 | B3 | B4 | Запасы |
| u1=0 | A1 | |  |  | | --- | --- | |  | 2 | | 40 |  | | |  |  | | --- | --- | |  | 3 | | 20 |  | | |  |  | | --- | --- | |  | 5 | |  |  | | |  |  | | --- | --- | |  | 1 | |  |  | | 60 |
| u2=1 | A2 | |  |  | | --- | --- | |  | 3 | |  |  | | |  |  | | --- | --- | |  | 4 | | 10 |  | | |  |  | | --- | --- | | - | 9 | | 30 |  | | |  |  | | --- | --- | | + | 4 | | 30 |  | | 70 |
| u3=2 | A3 | |  |  | | --- | --- | |  | 2 | |  |  | | |  |  | | --- | --- | |  | 5 | |  |  | | |  |  | | --- | --- | | λ | 2 | | + |  | | |  |  | | --- | --- | | - | 5 | | 20 |  | | 20 |
|  | Потребности | 40 | 30 | 30 | 50 | 150 |

Таблица 7.6.3 – Таблица нового плана

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  | v1=2 | v2=3 | v3=8 | v4=3 |  |
|  | Пункты | B1 | B2 | B3 | B4 | Запасы |
| u1=0 | A1 | |  |  | | --- | --- | |  | 2 | | 40 |  | | |  |  | | --- | --- | |  | 3 | | 20 |  | | |  |  | | --- | --- | |  | 5 | |  |  | | |  |  | | --- | --- | |  | 1 | |  |  | | 60 |
| u2=1 | A2 | |  |  | | --- | --- | |  | 3 | |  |  | | |  |  | | --- | --- | |  | 4 | | 10 |  | | |  |  | | --- | --- | |  | 9 | | 10 |  | | |  |  | | --- | --- | |  | 4 | | 50 |  | | 70 |
| u3=2 | A3 | |  |  | | --- | --- | |  | 2 | |  |  | | |  |  | | --- | --- | |  | 5 | |  |  | | |  |  | | --- | --- | |  | 2 | | 20 |  | | |  |  | | --- | --- | |  | 5 | |  |  | | 20 |
|  | Потребности | 40 | 30 | 30 | 50 | 150 |

Стоимость перевозок по этому плану:

(ед.)

Вычислим потенциалы и исходя из базисных переменных. Для их нахождения используем условие .

Считая , имеем .

Для каждой свободной клетки вычислим относительные оценки:

Условие оптимальности плана перевозок не выполняется, поэтому построим замкнутый цикл пересчета и определим величины перераспределения груза.

Минимальной оценкой является для клетки (1,3).

Для определения количества груза подлежащего распределению, построим замкнутый цикл (указан стрелками) (Таблица 7.6.4). Одна из вершин цикла находится в незанятой клетке , которую отмечаем знаком . Все остальные вершины цикла находятся в базисных клетках, с чередующимися знаками и . Найдем , равное наименьшему из чисел, стоящих в отрицательных вершинах цикла. Значение записываем в незанятую клетку. Двигаясь далее по означенному циклу, вычитаем из объемов перевозок, расположенных в клетках, которые обозначены знаком , и прибавляем к объемам перевозок, находящихся в клетках, отмеченных знаком . Элементы таблицы, не входящие в цикл, остаются без изменений. Таблица нового плана представлена в Таблице 7.6.5.

Таблица 7.6.4 – Таблица перерасчёта

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  | v1=2 | v2=3 | v3=8 | v4=3 |  |
|  | Пункты | B1 | B2 | B3 | B4 | Запасы |
| u1=0 | A1 | |  |  | | --- | --- | |  | 2 | | 40 |  | | |  |  | | --- | --- | | - | 3 | | 20 |  | | |  |  | | --- | --- | | λ | 5 | | + |  | | |  |  | | --- | --- | |  | 1 | |  |  | | 60 |
| u2=1 | A2 | |  |  | | --- | --- | |  | 3 | |  |  | | |  |  | | --- | --- | | + | 4 | | 10 |  | | |  |  | | --- | --- | | - | 9 | | 10 |  | | |  |  | | --- | --- | |  | 4 | | 50 |  | | 70 |
| u3=-6 | A3 | |  |  | | --- | --- | |  | 2 | |  |  | | |  |  | | --- | --- | |  | 5 | |  |  | | |  |  | | --- | --- | |  | 2 | | 20 |  | | |  |  | | --- | --- | |  | 5 | |  |  | | 20 |
|  | Потребности | 40 | 30 | 30 | 50 | 150 |

Таблица 7.6.5 – Таблица нового плана

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  | v1=2 | v2=3 | v3=8 | v4=3 |  |
|  | Пункты | B1 | B2 | B3 | B4 | Запасы |
| u1=0 | A1 | |  |  | | --- | --- | |  | 2 | | 40 |  | | |  |  | | --- | --- | |  | 3 | | 10 |  | | |  |  | | --- | --- | |  | 5 | | 10 |  | | |  |  | | --- | --- | |  | 1 | |  |  | | 60 |
| u2=1 | A2 | |  |  | | --- | --- | |  | 3 | |  |  | | |  |  | | --- | --- | |  | 4 | | 20 |  | | |  |  | | --- | --- | |  | 9 | |  |  | | |  |  | | --- | --- | |  | 4 | | 50 |  | | 70 |
| u3=-6 | A3 | |  |  | | --- | --- | |  | 2 | |  |  | | |  |  | | --- | --- | |  | 5 | |  |  | | |  |  | | --- | --- | |  | 2 | | 20 |  | | |  |  | | --- | --- | |  | 5 | |  |  | | 20 |
|  | Потребности | 40 | 30 | 30 | 50 | 150 |

Стоимость перевозок по этому плану:

(ед.)

Вычислим потенциалы и исходя из базисных переменных. Для их нахождения используем условие .

Считая , имеем .

Для каждой свободной клетки вычислим относительные оценки:

Условие оптимальности плана перевозок не выполняется, поэтому построим замкнутый цикл пересчета и определим величины перераспределения груза.

Минимальной оценкой является для клетки (1,4).

Для определения количества груза подлежащего распределению, построим замкнутый цикл (указан стрелками) (Таблица 7.6.6). Одна из вершин цикла находится в незанятой клетке , которую отмечаем знаком . Все остальные вершины цикла находятся в базисных клетках, с чередующимися знаками и . Найдем , равное наименьшему из чисел, стоящих в отрицательных вершинах цикла. Значение записываем в незанятую клетку. Двигаясь далее по означенному циклу, вычитаем из объемов перевозок, расположенных в клетках, которые обозначены знаком , и прибавляем к объемам перевозок, находящихся в клетках, отмеченных знаком . Элементы таблицы, не входящие в цикл, остаются без изменений. Таблица нового плана представлена в Таблице 7.6.7.

Таблица 7.6.6 – Таблица перерасчёта

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  | v1=2 | v2=3 | v3=5 | v4=3 |  |
|  | Пункты | B1 | B2 | B3 | B4 | Запасы |
| u1=0 | A1 | |  |  | | --- | --- | |  | 2 | | 40 |  | | |  |  | | --- | --- | | - | 3 | | 10 |  | | |  |  | | --- | --- | |  | 5 | | 10 |  | | |  |  | | --- | --- | | λ | 1 | | + |  | | 60 |
| u2=1 | A2 | |  |  | | --- | --- | |  | 3 | |  |  | | |  |  | | --- | --- | | + | 4 | | 20 |  | | |  |  | | --- | --- | |  | 9 | |  |  | | |  |  | | --- | --- | | - | 4 | | 50 |  | | 70 |
| u3=-3 | A3 | |  |  | | --- | --- | |  | 2 | |  |  | | |  |  | | --- | --- | |  | 5 | |  |  | | |  |  | | --- | --- | |  | 2 | | 20 |  | | |  |  | | --- | --- | |  | 5 | |  |  | | 20 |
|  | Потребности | 40 | 30 | 30 | 50 | 150 |

Таблица 7.7.7 – Таблица нового плана

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  | v1=2 | v2=3 | v3=5 | v4=3 |  |
|  | Пункты | B1 | B2 | B3 | B4 | Запасы |
| u1=0 | A1 | |  |  | | --- | --- | |  | 2 | | 40 |  | | |  |  | | --- | --- | |  | 3 | |  |  | | |  |  | | --- | --- | |  | 5 | | 10 |  | | |  |  | | --- | --- | |  | 1 | | 10 |  | | 60 |
| u2=1 | A2 | |  |  | | --- | --- | |  | 3 | |  |  | | |  |  | | --- | --- | |  | 4 | | 30 |  | | |  |  | | --- | --- | |  | 9 | |  |  | | |  |  | | --- | --- | |  | 4 | | 40 |  | | 70 |
| u3=-3 | A3 | |  |  | | --- | --- | |  | 2 | |  |  | | |  |  | | --- | --- | |  | 5 | |  |  | | |  |  | | --- | --- | |  | 2 | | 20 |  | | |  |  | | --- | --- | |  | 5 | |  |  | | 20 |
|  | Потребности | 40 | 30 | 30 | 50 | 150 |

Стоимость перевозок по этому плану:

(ед.)

Вычислим потенциалы и исходя из базисных переменных. Для их нахождения используем условие .

Считая , имеем .

Для каждой свободной клетки вычислим относительные оценки:

Условие оптимальности плана перевозок не выполняется, поэтому построим замкнутый цикл пересчета и определим величины перераспределения груза.

Минимальной оценкой является для клетки (2,1).

Для определения количества груза подлежащего распределению, построим замкнутый цикл (указан стрелками) (Таблица 7.7.8). Одна из вершин цикла находится в незанятой клетке , которую отмечаем знаком . Все остальные вершины цикла находятся в базисных клетках, с чередующимися знаками и . Найдем , равное наименьшему из чисел, стоящих в отрицательных вершинах цикла. Значение записываем в незанятую клетку. Двигаясь далее по означенному циклу, вычитаем из объемов перевозок, расположенных в клетках, которые обозначены знаком , и прибавляем к объемам перевозок, находящихся в клетках, отмеченных знаком . Элементы таблицы, не входящие в цикл, остаются без изменений. Таблица нового плана представлена в Таблице 7.7.9.

Таблица 7.7.8 – Таблица перерасчёта

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  | v1=2 | v2=1 | v3=5 | v4=1 |  |
|  | Пункты | B1 | B2 | B3 | B4 | Запасы |
| u1=0 | A1 | |  |  | | --- | --- | | - | 2 | | 40 |  | | |  |  | | --- | --- | |  | 3 | |  |  | | |  |  | | --- | --- | |  | 5 | | 10 |  | | |  |  | | --- | --- | | + | 1 | | 10 |  | | 60 |
| u2=3 | A2 | |  |  | | --- | --- | | λ | 3 | | + |  | | |  |  | | --- | --- | |  | 4 | | 30 |  | | |  |  | | --- | --- | |  | 9 | |  |  | | |  |  | | --- | --- | | - | 4 | | 40 |  | | 70 |
| u3=-3 | A3 | |  |  | | --- | --- | |  | 2 | |  |  | | |  |  | | --- | --- | |  | 5 | |  |  | | |  |  | | --- | --- | |  | 2 | | 20 |  | | |  |  | | --- | --- | |  | 5 | |  |  | | 20 |
|  | Потребности | 40 | 30 | 30 | 50 | 150 |

Таблица 7.7.9 – Таблица нового плана

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  | v1=2 | v2=1 | v3=5 | v4=1 |  |
|  | Пункты | B1 | B2 | B3 | B4 | Запасы |
| u1=0 | A1 | |  |  | | --- | --- | |  | 2 | |  |  | | |  |  | | --- | --- | |  | 3 | |  |  | | |  |  | | --- | --- | |  | 5 | | 10 |  | | |  |  | | --- | --- | |  | 1 | | 50 |  | | 60 |
| u2=3 | A2 | |  |  | | --- | --- | |  | 3 | | 40 |  | | |  |  | | --- | --- | |  | 4 | | 30 |  | | |  |  | | --- | --- | |  | 9 | |  |  | | |  |  | | --- | --- | |  | 4 | | 0 |  | | 70 |
| u3=-3 | A3 | |  |  | | --- | --- | |  | 2 | |  |  | | |  |  | | --- | --- | |  | 5 | |  |  | | |  |  | | --- | --- | |  | 2 | | 20 |  | | |  |  | | --- | --- | |  | 5 | |  |  | | 20 |
|  | Потребности | 40 | 30 | 30 | 50 | 150 |

Стоимость перевозок по этому плану:

(ед.)

Вычислим потенциалы и исходя из базисных переменных. Для их нахождения используем условие .

Считая , имеем

Для каждой свободной клетки вычислим относительные оценки:

Отрицательных оценок нет, значит решение является оптимальным. Стоимость перевозок при этом составляет (ед.)

7.7 Результаты выполнения программы

Результаты выполнения программы, реализующей решение транспортной задачи, представлены на Рисунках 7.7.1 – 7.7.8

Изображение выглядит как снимок экрана, текст, линия

Автоматически созданное описание

Рисунок 7.7.1 – Изначальная таблица

Изображение выглядит как текст, снимок экрана, дисплей, линия

Автоматически созданное описание

Рисунок 7.7.2 – Метод минимальной стоимости

Изображение выглядит как снимок экрана, текст, линия

Автоматически созданное описание

Рисунок 7.7.3 – Метод северо-западного угла

Изображение выглядит как снимок экрана, текст, диаграмма, Мультимедийное программное обеспечение

Автоматически созданное описание

Рисунок 7.7.4 – Первая итерация метода потенциалов

Изображение выглядит как снимок экрана, текст, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

Рисунок 7.7.5 – Вторая итерация метода потенциалов

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

Рисунок 7.7.6 – Третья итерация метода потенциалов

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

Рисунок 7.7.7 – Четвёртая итерация метода потенциалов

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

Рисунок 7.7.8 – Последняя итерация метода потенциалов

7.8 Заключение

В ходе выполнения данной работы мной была изучена транспортная задача, её методы решения. Была решена конкретная транспортная задача, т.е. найден оптимальный план перевозок. Также была написана программа для решения транспортных задач.

ЗАКЛЮЧЕНИЕ

В ходе выполнения данной курсовой работы изучено три метода многокритериальной оптимизации – метод Парето и его методы оптимизации, метод Электра II и метод анализа иерархий.

Метод Парето самый простой, однако в нём есть существенный недостаток – по нему сложно получить единственное оптимальное решение. Для устранения этого недостатка существует методы оптимизации – метод указания верхних/нижних границ критериев, метод субоптимизации и лексикографическая оптимизация, однако и они дают либо несколько оптимальных решений, либо дают слишком субъективное решение.

Метод Электра II менее субъективен по сравнению с методом Парето, однако его сложнее реализовать в программе, а также метод всё равно может дать несколько решений. А чтобы избавиться от этого, надо экспериментально подбирать значение порога.

Преимуществом метода анализа иерархий является гарантированное получение единственного оптимального решения, однако недостатками являются высокая субъективность решения, т.к. приоритеты критериев выставляются ЛПР вручную. Также при несогласованности матриц сравнения критериев нужно расставлять все приоритеты заново, что может занять много времени.

Также в ходе выполнения курсовой работы изучено линейное программирование и методы его решения – графический и симплексный. Ещё были изучены двойственные и транспортные задачи.

Графический метод – это очень наглядный метод, который позволяет достаточно просто решить небольшие задачи линейного программирования. Однако недостатком является практическая невозможность решать таким методов задачи более, чем с 2 переменными.

Симплексный метод избавлен от такого недостатка, и позволяет решать задачи линейного программирования с любым количеством переменных и любым количеством неравенств. Но у метода тоже есть недостаток – время решения задачи может существенно увеличиться при неудачных входных данных.

Двойственная задача, по сути, является обратной задачей. Три теоремы двойственности позволяют глубоко проанализировать решение прямой задачи, изучив, какие переменные являются дефицитными, а какие нет, а также насколько можно изменить ограничения в прямой задаче, чтобы можно было увеличить/уменьшить полученную выгоду.

Транспортная задача – это отдельный вид задач линейного программирования. Для неё существуют более оптимальные методы решения. Например, можно применять метод северо-западного угла для получения начального плана, а затем использовать метод потенциалов для оптимизации, или можно использовать метод минимальной стоимости, который может выдать даже более оптимальное решение, чем метод потенциалов.

СПИСОК ИНФОРМАЦИОННЫХ ИСТОЧНИКОВ

1. Болотова Л. С. Многокритериальная оптимизация. Болотова Л. С., Сорокин А. Б. [Электронный ресурс] / Метод. указания по вып. курсовой работы — М.: МИРЭА, 2015.

2. Сорокин А. Б. Методы оптимизации: гибридные генетические алгоритмы. Сорокин А. Б. [Электронный ресурс] / Учебно-метод. пособие — М.: МИРЭА, 2016.

3. Сорокин А. Б. Линейное программирование: практикум. Сорокин А. Б., Бражникова Е. В., Платонова О. В. [Электронный ресурс] / Учебно-метод. пособие — М.: МИРЭА, 2017.

4. Сухарев, А.Г. Курс методов оптимизации: учеб. пособие / А.Г. Сухарев, А.В. Тимохов, В.В. Федоров. –М.: Физматлит, 2011. – 384 с.

5. Афанасьев М.Ю. Прикладные задачи исследования операций: Учеб. пособие / М.Ю. Афанасьев, К.А. Багриновский, В.М. Матюшок – М.: ИНФРАМ, 2006 – 352 с.

6. Ашманов С.А. Теория оптимизации в задачах и упражнениях: Учеб. пособие для вузов / С.А. Ашманов, А.В. Тимохов. – СПб.: Лань, 2012. – 447 с.

ПРИЛОЖЕНИЯ

Приложение А – Код реализации метода Парето на языке Python.

Приложение Б – Код реализации метода Электра II на языке Python.

Приложение В – Код реализации МАИ на языке Python.

Приложение Г – Код реализации симплексного метода на языке Python.

Приложение Д – Код реализации двойственной задачи на языке Python.

Приложение Е – Код реализации транспортной задачи на языке Python.

### Приложение А

Код реализации метода Парето на языке Python

*Листинг А.1. Реализация Парето.*

import csv

import pandas as pd

def print\_table(data):

'''Функция для вывода таблицы'''

print(pd.DataFrame(data).to\_markdown())

def compare\_alternatives(a, b):

'''Функция для сравнения альтернатив по отношению Парето-доминирования'''

counter = 0

for key in a:

if '+' in key:

counter += (float(a[key]) > float(b[key]))

elif '-' in key:

counter += (float(a[key]) < float(b[key]))

return 1 if counter == len(a) - 1 else -1 if counter == 0 else 0

def create\_Pareto\_set(data):

'''Функция для создания оптимального множества Парето по входящему множеству альтернатив'''

losers, winners = [], []

for i in range(len(data)):

for j in range(i+1, len(data)):

n = compare\_alternatives(data[i], data[j])

if n == 1:

losers.append(data[j])

elif n == -1:

losers.append(data[i])

for i in range(len(data)):

if data[i] not in losers:

winners.append(data[i])

return winners

def branches\_and\_boundaries(data, branches):

'''Метод указания верхних и нижних границ критериев'''

winners = []

for i in data:

flag = False

for j in branches:

key, value = list(j.items())[0]

if key.count('-'):

if float(i[key]) > value:

flag = True

else:

if float(i[key]) < value:

flag = True

if not flag:

winners.append(i)

return create\_Pareto\_set(winners)

def suboptimization(data, branches, main\_criteria):

'''Метод субоптимизации'''

*Продолжение Листинга А.1.*

data = branches\_and\_boundaries(data, branches)

maxi = max(data, key=lambda i: i[main\_criteria])

return list(filter(lambda x: x[main\_criteria] == maxi[main\_criteria], data))

def lexical\_optimization(data, priority):

'''Лексикографический метод'''

return [max(data, key = lambda item: tuple(item[key] for key in priority))]

with open('TPR\_PRACT1\_LIST.csv', encoding='utf-8') as file:

data = [d for d in csv.DictReader(file)]

print("Исходная таблица с альтернативами и критериями:".center(201))

print\_table(data)

print("Оптимальное-множество Парето:".center(201))

print\_table(create\_Pareto\_set(data))

print("Установка верхних и нижних границ:".center(201))

branches = [{"Проходной балл (+)": 270}, {"Рейтинг университета (+)": 840},

{"Расстояние до общежития (-)": 14}]

print\_table(branches\_and\_boundaries(data, branches))

print("Субоптимизация:".center(201))

branches = [{"Проходной балл (+)": 290}, {"Расстояние до общежития (-)": 14}]

main\_criteria = "Рейтинг университета (+)"

print\_table(suboptimization(data, branches, main\_criteria))

print("Лексикографическая оптимизация:".center(201))

priority = ("Рейтинг университета (+)", "Проходной балл (+)", "Стоимость обучения (+)",

"Кол-во бюджетных мест (-)", "Расстояние до общежития (-)",

"Размер стипендии (+)")

print\_table(lexical\_optimization(data, priority))

### Приложение Б

Код реализации метода Электра II на языке Python.

*Листинг Б.1. Реализация метода Электра II.*

import csv

import math

from graphviz import Digraph

def print\_matrix(c = 1):

'''Функция для вывода матрицы предпочтений с порогом'''

print('-' \* (10 \* (len(matrix)+1) + 4))

print(11 \* ' ', end='')

for i in range(1, len(matrix) + 1):

print(f'{i}'.ljust(11), end='')

print('\n' + '-' \* (10 \* (len(matrix)+1) + 4))

for i in range(len(matrix)):

print(str(i+1).ljust(10), end='|')

for j in range(len(matrix)):

if matrix[i][j] < c:

matrix[i][j] = 0

print(str(matrix[i][j]).ljust(10), end=' ')

print()

print('-' \* (10 \* (len(matrix)+1) + 4))

def compare\_alternatives(i, j, alt\_i, alt\_j, criteria):

'''Функция для сравнения альтернатив по кодам'''

P, N = 0, 0

P\_STR, N\_STR = f'P{i}{j} =', f'N{i}{j} ='

for i in range(len(criteria)):

counter\_i, counter\_j = 0, 0

code = criteria[i]['Код'].split(';')

for border in criteria[i]['Шкала'].split(';')[1:]:

if alt\_i[i] < float(border):

counter\_i += 1

if alt\_j[i] < float(border):

counter\_j += 1

alt\_i[i] = float(code[counter\_i])

alt\_j[i] = float(code[counter\_j])

if criteria[i]['Стремление'] == '-':

alt\_i[i] \*= (-1)

alt\_j[i] \*= (-1)

if alt\_i[i] > alt\_j[i]:

P += int(criteria[i]['Вес критерия'])

P\_STR += (' ' + criteria[i]['Вес критерия'] + ' +')

N\_STR += (' ' + str(0) + ' +')

elif alt\_i[i] < alt\_j[i]:

N += int(criteria[i]['Вес критерия'])

N\_STR += (' ' + criteria[i]['Вес критерия'] + ' +')

P\_STR += (' ' + str(0) + ' +')

else:

N\_STR += (' ' + str(0) + ' +')

P\_STR += (' ' + str(0) + ' +')

return P\_STR.rstrip(' +') + f' = {P}', N\_STR.rstrip(' +') + f' = {N}', P, N

def generate\_matrix():

'''Функция для генерации матрицы предпочтений'''

*Продолжение Листинга Б.1*

def get\_D(P, N):

'''Функция для расчёта D-стремления'''

if N == 0 and P == 0:

return 1

elif N == 0 and P != 0:

return math.inf

value = P/N

if math.floor(value) == math.ceil(value):

value = int(value)

else:

value = round(value, 2)

return value

def generate\_D\_STR(i, j, P, N):

'''Функция для генерации D-стремления'''

value = get\_D(P, N)

if value <= 1:

return f'D{i}{j} = P{i}{j} / N{i}{j} = {P}/{N} = {value} <= 1 - отбрасываем.'

else:

if value == math.inf:

value = '\u221e'

return f'D{i}{j} = P{i}{j} / N{i}{j} = {P}/{N} = {value} > 1 - принимаем.'

for i in range(1, len(data)+1):

for j in range(i+1, len(data)+1):

print(f'Рассмотрим альтернативы {i} и {j} (i = {i}, j = {j}):')

alt\_i, alt\_j = data[i-1].copy(), data[j-1].copy()

P\_STR, N\_STR, P, N = compare\_alternatives(

i, j, alt\_i, alt\_j, criteria)

print(P\_STR+';', N\_STR+';', sep='\n')

print(generate\_D\_STR(i, j, P, N))

D = get\_D(P, N)

if D > 1:

matrix[i-1][j-1] = D

print(f'P{j}{i}{N\_STR[3:]};', f'N{j}{i}{P\_STR[3:]};', sep='\n')

print(generate\_D\_STR(j, i, N, P))

D = get\_D(N, P)

if D > 1:

matrix[j-1][i-1] = D

def draw\_graph(c=1):

'''Функция для рисования хаотичного графа с порогом'''

dot = Digraph(f'Хаотичный Граф с порогом = {c}')

for i in range(len(matrix)):

dot.node(str(i+1))

for i in range(len(matrix)):

for j in range(len(matrix)):

if matrix[i][j] >= c:

dot.edge(str(i+1), str(j+1))

dot.render(view=True)

def smart\_draw\_graph(levels, c=1):

'''Функция для рисования графа по уровням с порогом'''

dot1 = Digraph(f"Граф с порогом = {c}")

for i in range(len(matrix)):

for j in range(len(matrix)):

if matrix[i][j] >= c:

*Продолжение Листинга Б.1*

dot1.edge(str(i+1), str(j+1))

for i in range(len(levels)):

sub = Digraph(name='Подграф'+str(i))

sub.attr(rank='same')

sub.node(f'{i+1}-ый уровень')

for j in levels[i]:

sub.node(f'{j+1}')

dot1.subgraph(sub)

dot1.render(view=True)

def get\_levels(c=1):

'''Вспомогательная функция для определения уровня вершин'''

levels = [] # массив всех вершин

visited = [] # массив посещённых вершин

while len(visited) < len(matrix):

level = []

for i in range(len(matrix)):

if i in visited:

continue

flag = True

for j in range(len(matrix)):

if matrix[j][i] >= c:

flag = any(j in lev for lev in levels)

if not flag:

break

if flag:

level.append(i)

visited.append(i)

levels.append(level)

print(f'{len(levels)}-ый уровень: ' +

', '.join(map(lambda x: str(x+1), level)))

return levels

with open('TPR\_PRACT2\_LIST.csv', encoding='utf-8') as file, \

open('codes.csv', encoding='utf-8') as criteria\_file:

criteria = [i for i in csv.DictReader(

criteria\_file)] # Информация о критериях

data = list(map(lambda x: [float(i) for i in x], [

i[1:] for i in csv.reader(file)][1:])) # Значения критериев для рассматриваемых альтернатив

matrix = [[0]\*len(data) for \_ in range(len(data))] # Матрица предпочтений

generate\_matrix()

print("Матрица предпочтений:".center(201))

print\_matrix()

draw\_graph()

arg = 1.76

smart\_draw\_graph(get\_levels(c=arg), c=arg)

### Приложение В

Код реализации МАИ на языке Python.

Листинг В.1. Реализация МАИ.

import pandas as pd

import functools

NUM\_CRITERIA = 5 # Количество критериев для сравнения

NUM\_ALTERNATIVES = 5 # Количество альтернатив

СИ = 1.12 # среднее значение индекса случайной согласованности

def print\_table(data):

'''Функция для вывода таблицы'''

print(pd.DataFrame(data).to\_markdown())

def relative\_value(data, MATRIX\_SIZE=NUM\_CRITERIA):

'''Функция для определения относительной ценности элемента (геометрическое среднее)'''

return round(functools.reduce(lambda a, b: a \* b, data) \*\* (1 / MATRIX\_SIZE), 3)

def compare\_by\_criteria(data, index):

'''Функция для сравнения в пределах матрицы сравнения по критерию N'''

V = []

for i in range(NUM\_ALTERNATIVES):

val = relative\_value(data[i])

print(

f'Строка №{i + 1}\nVk{index}{i+1} = ({" \* ".join([str(i) for i in data[i]])}) ^ 1/{NUM\_ALTERNATIVES} = {val}')

V.append(val)

print(

f'Проведена нормализация полученных чисел. Для этого определен нормирующий коэффициент ∑VK{index}Y.')

print(

f'∑VК{index}Y = VK{index}1 + VK{index}2 + VK{index}3 + VK{index}4 + VK{index}5 = {" + ".join([str(i) for i in V])} = {round(sum(V), 3)}.')

print(

f'Найдена важность приоритетов W3К{index}Y, для этого каждое из чисел VK{index}Y разделено на ∑VK{index}Y.')

Y = []

for i in range(NUM\_ALTERNATIVES):

Y.append(round(V[i] / sum(V), 3))

print(

f'Строка №{i + 1}\nW3K{index}{i + 1} = {V[i]} / ∑Vi = {V[i]} / {sum(V)} = {Y[i]};')

print('В результате получаем вектор приоритетов:')

print(f'W3K{index}Y = ({"; ".join([f"Y3{index}{i + 1} = {Y[i]}" for i in range(len(Y))])}), '

+ f'где индекс 3 означает, что вектор приоритетов относится к третьему уровню иерархии критерия K{index}.')

return Y

def check\_matrix\_consistency(data, priority\_vector, index):

*Продолжение Листинга В.1.*

print(

f'Определены индекс согласованности и отношение согласованности для матрицы К{index}')

print('Определяется сумма каждого столбца матрицы суждений.')

counter = 1

S = []

for i in zip(\*data):

S.append(sum(i))

print(

f'S{counter}K{index} = {" + ".join([str(i) for i in list(i)])} = {sum(i)}')

counter += 1

print('Затем полученный результат умножен на компоненту нормализованного вектора приоритетов.')

P = []

for i in range(len(S)):

P.append(round(S[i] \* priority\_vector[i], 3))

print(f'P{i + 1}K{index} = S{i + 1} \* W3K{index}{i + 1} = {P[i]}')

print('Найдена пропорциональность предпочтений.')

print(f'λmaxK{index} = Р1K{index} + Р2K{index} + Р3K{index} + Р4K{index} + Р5K{index} = {round(sum(P), 3)}')

print('Отклонение от согласованности выражается индексом согласованности.')

ИС = round((round(sum(P), 3) - 5) / (5 - 1), 3)

print(

f'ИСk{index} = (λmaxK{index} - n)/(n - 1) = ({round(sum(P), 3)}-5)/(5-1) = {ИС}.')

print('Найдено отношение согласованности ОС.')

print(f'ОСk{index} = ИС/СИ = {round(ИС / СИ, 3)}.')

def synthesis\_of\_alternatives(Y, full\_Y):

print(f'Векторы приоритетов и отношения согласованности определяются для всех матриц суждений, начиная со второго уровня.\n' +

f'Для определения приоритетов альтернатив локальные приоритеты умножены на приоритет соответствующего критерия' +

f'на высшем уровне и найдены суммы по каждому элементу в соответствии с критериями, на которые воздействует этот элемент.')

print(

f'W2i = ({"; ".join([f"Y2{i + 1} = {Y[i]}" for i in range(len(Y))])});')

for i in range(len(full\_Y)):

print(

f'W3K{i + 1}Y = ({"; ".join([f"Y3{i + 1}{j + 1} = {full\_Y[i][j]}" for j in range(len(full\_Y[i]))])});')

print('Приоритеты альтернатив получены следующим образом:')

winners = []

counter = 0

for v in zip(\*full\_Y):

curr\_str = str()

w = 0

counter += 1

curr\_str += f'W{counter} = '

for i in range(len(v)):

curr\_str += f'W2{i + 1} \* W3K{i + 1}{counter} + '

w += (v[i] \* Y[i])

w = round(w, 3)

print(curr\_str.rstrip(' + ') + ' = ' + str(w))

winners.append(w)

print('Таким образом, приоритеты альтернатив равны:')

for i in range (NUM\_ALTERNATIVES):

print(f'альтернатива А{i+1} - W{i + 1} приоритет равен {winners[i]}')

return winners

*Продолжение Листинга B.1.*

with open('TPR\_PRACT3.csv', encoding='utf-8') as file:

title = file.readline().rstrip() # Текущая матрица, которая будет считана

print(title.center(201))

criteria\_paired\_comparison\_matrix = [[float(i) for i in file.readline().rstrip(

).split(',')] for \_ in range(NUM\_CRITERIA)] # Матрица парного сравнения критериев

print\_table(criteria\_paired\_comparison\_matrix)

print('Для определения относительной ценности каждого элемента необходимо найти геометрическое' +

' среднее и с этой целью перемножить n элементов каждой строки и из полученного' +

' результата извлечь корни n-й степени (размерность матрицы n=5).')

V = []

for i in range(NUM\_CRITERIA):

val = relative\_value(criteria\_paired\_comparison\_matrix[i])

print(

f'Строка №{i + 1}\nV{i + 1} = ({" \* ".join([str(i) for i in criteria\_paired\_comparison\_matrix[i]])}) ^ 1/{NUM\_CRITERIA} = {val}')

V.append(val)

print(

f'∑Vi = V1 + V2 + V3 + V4 + V5 = {" + ".join([str(i) for i in V])} = {round(sum(V),3)}')

print('Найдена важность приоритетов W2i, для этого каждое из чисел Vi разделено на ∑Vi.')

Y = [] # Вектор приоритетов W2i

for i in range(NUM\_CRITERIA):

Y.append(round(V[i] / sum(V), 3))

print(

f'Строка №{i + 1}\nW2{i + 1} = {V[i]} / ∑Vi = {Y[i]} = Y{2}{i + 1}')

print(f'В результате получен вектор приоритетов:\nW2i = ({"; ".join([f"Y2{i + 1} = {Y[i]}" for i in range(len(Y))])}), '

+ 'где индекс 2 означает, что вектор приоритетов относится ко второму уровню иерархии.')

big\_data, full\_Y = [], []

for i in range(NUM\_CRITERIA):

title = file.readline().rstrip()

print(title.center(201))

data = [[float(i) for i in file.readline().rstrip().split(',')] for \_ in range(

NUM\_ALTERNATIVES)] # Матрица сравнения по i + 1-ому критерию

big\_data.append(data)

print\_table(data)

Yi = compare\_by\_criteria(data, i + 1)

full\_Y.append(Yi)

print('Определены индекс согласованности и отношение согласованности для матрицы «Выбор лучшего технического вуза»')

counter = 1

S = []

for i in zip(\*criteria\_paired\_comparison\_matrix):

S.append(sum(i))

print(

f'S{counter} = {" + ".join([str(i) for i in list(i)])} = {sum(i)}')

counter += 1

print(f'Полученный результат умножен на компоненту нормализованного вектора приоритетов, ' +

f'т.е. сумму суждений первого столбца на первую компоненту, сумму

*Продолжение Листинга В.1.*

суждений второго столбца - на вторую и т.д.')

P = []

for i in range(len(S)):

P.append(round(S[i] \* Y[i], 3))

print(f'P{i + 1} = S{i + 1} \* W2{i + 1} = {P[i]}')

print(f'Сумма чисел Рj отражает пропорциональность предпочтений, ' +

f'чем ближе эта величина к n (числу объектов и видов действия в матрице парных сравнений), тем более согласованны суждения.')

print(f'λmax = Р1 + Р2 + Р3 + Р4 + Р5 = {round(sum(P), 3)}')

print('Отклонение от согласованности выражается индексом согласованности.')

ИС = round((round(sum(P), 3) - 5) / (5 - 1), 3)

print(f'ИС = (λmax - n)/(n - 1) = ({round(sum(P), 3)}-5)/(5-1) = {ИС}.')

print('Отношение индекса согласованности ИС к среднему значению случайного индекса согласованности СИ называется отношением согласованности ОС.')

print(f'ОС = ИС/СИ = {round(ИС / СИ, 3)}.')

for i in range(NUM\_CRITERIA):

print('\n')

check\_matrix\_consistency(big\_data[i], full\_Y[i], i + 1)

synthesis\_of\_alternatives(Y, full\_Y)

### Приложение Г

Код реализации симплексного метода на языке Python.

*Листинг Г.1. Реализация симплексного метода.*

import re

NUM\_CRITERIA = 4 # Количество ограничений в математической моделе

PRECISION = 4 # Количество знаков после запятой при округлении

SEP = 25 # Разделитель для вывода таблицы

def print\_table(system, coef\_basis, coef\_not\_basis, basis\_values, not\_basis\_values):

print(' '.ljust(SEP), end='')

print('Cj'.ljust(SEP), end='')

for i in range(len(coef\_not\_basis) + 1):

if i == len(coef\_not\_basis):

print(' '.ljust(SEP))

else:

print(str(coef\_not\_basis[i]).ljust(SEP), end='')

print('Cv'.ljust(SEP), end='')

for i in range(len(not\_basis\_values) + 1):

if i == 0:

print(''.ljust(SEP), end='')

else:

print(str(not\_basis\_values[i-1]).ljust(SEP), end='')

print('A0'.ljust(SEP))

system.insert(0, coef\_basis + [' '])

system.insert(1, basis\_values + ['f'])

for column in range(len(system[0])):

for row in range(len(system)):

print(str(system[row][column]).ljust(SEP), end='')

print()

del system[0]

del system[0]

def get\_coefficients(data):

'''Функция для получения списка коэффициентов из системы ограничений'''

criteria\_coefficients, boundaries = list(), list()

for exp in data:

if '<=' in exp:

parse = exp.split('<=')

elif '>=' in exp:

parse = exp.split('>=')

elif '<' in exp:

parse = exp.split('<')

elif '>' in exp:

parse = exp.split('>')

elif '=' in exp:

parse = exp.split('=')

parse = list(map(str.strip, parse))

boundaries.append(float(parse[1]))

criteria\_coefficients.append(list(map(float, [i.group(1) for i in re.finditer(

r'(\d+(\.\d+)?) {0,}[\*]? {0,}\w', parse[0])])))

return criteria\_coefficients, boundaries

*Продолжение Листинга Г.1.*

def count\_scalar\_product(vec1, vec2):

'''Функция для рассчёта скалярного произведения двух векторов'''

res = 0

for i in range(len(vec1)):

res += (vec1[i] \* vec2[i])

return res

def create\_simplex\_table(system, coef\_basis, coef\_not\_basis, basis\_values, not\_basis\_values):

F\_str = [0] \* len(not\_basis\_values)

for i in range(len(not\_basis\_values)):

F\_str[i] = count\_scalar\_product(

coef\_basis, system[i]) - coef\_not\_basis[i]

Q = count\_scalar\_product(coef\_basis, system[-1])

for i in range(len(F\_str)):

system[i].append(F\_str[i])

system[-1].append(Q)

return F\_str, Q

def simplex\_iteration(system, coef\_basis, coef\_not\_basis, basis\_values, not\_basis\_values, F\_str, Q):

index\_column = F\_str.index(min(F\_str))

mini = 1e10

for i in range(len(system[-1]) - 1):

tmp = system[-1][i] / system[index\_column][i]

if tmp < mini:

mini = tmp

index\_row = i

key\_element = system[index\_column][index\_row]

basis\_values.insert(index\_row, not\_basis\_values[index\_column])

not\_basis\_values.insert(index\_column, basis\_values.pop(index\_row + 1))

del not\_basis\_values[index\_row]

coef\_basis[index\_row], coef\_not\_basis[index\_column] = coef\_not\_basis[index\_column], coef\_basis[index\_row]

new\_key\_element = round(1 / key\_element, PRECISION)

data = [[0] \* (len(coef\_basis) + 1)

for \_ in range(len(coef\_not\_basis) + 1)]

for i in range(len(system[index\_column])):

data[index\_column][i] = - round(system[index\_column][i] / key\_element, PRECISION)

for i in range(len(system)):

data[i][index\_row] = round(

system[i][index\_row] / key\_element, PRECISION)

data[index\_column][index\_row] = new\_key\_element

for row in range(len(data[0])):

for column in range(len(data)):

if data[column][row] == 0:

data[column][row] = round(((system[column][row] \* key\_element) - (

system[index\_column][row] \* system[column][index\_row])) / key\_element, PRECISION)

F\_str = [data[i][-1] for i in range(len(data) - 1)]

Q = data[-1][-1]

return data, coef\_basis, coef\_not\_basis, basis\_values, not\_basis\_values, F\_str, Q

with open('TPR\_PRACT5.csv', encoding='utf-8') as file:

target\_function = file.readline().rstrip() # Целевая функция

target\_coefficients = list(map(float, [i.group(1) for i in re.finditer(

*Продолжение Листинга Г.1.*

r'(\d+(\.\d+)?) {0,}[\*]? {0,}\w', target\_function)])) # Список коэффициентов целевой функции

criteria\_function = [file.readline().rstrip() for \_ in range (NUM\_CRITERIA)]

criteria\_coefficients, boundaries = get\_coefficients(criteria\_function)

print('Переходим к задаче линейного программирования:',

target\_function, sep='\n')

for i in criteria\_function:

print("{ " + i)

system = list(map(list, list(zip(\*criteria\_coefficients))))

system.append(boundaries)

# Вектор коэффициентов целевой функции при базисных переменных

coef\_basis = [0, 0, 0, 0]

# Коэффициенты целевой функции, соответствующие небазисным переменным

coef\_not\_basis = target\_coefficients.copy()

not\_basis\_values = re.findall(r'\w\d{1,}', target\_function)

basis\_values = [f'{not\_basis\_values[-1][0]}{i}' for i in range(

int(not\_basis\_values[-1][1]) + 1, NUM\_CRITERIA + int(not\_basis\_values[-1][1]) + 1)]

F\_str, Q = create\_simplex\_table(

system, coef\_basis, coef\_not\_basis, basis\_values, not\_basis\_values)

num\_iteration = 0

while num\_iteration < 50 and min(F\_str) < 0:

print(

('\x1b[6;30;42m' + f"Итерация №{num\_iteration}" + '\x1b[0m').center(201))

# print(f'Итерация №{num\_iteration}'.center(201))

system, coef\_basis, coef\_not\_basis, basis\_values, not\_basis\_values, F\_str, Q = simplex\_iteration(

system, coef\_basis, coef\_not\_basis, basis\_values, not\_basis\_values, F\_str, Q)

print\_table(system, coef\_basis, coef\_not\_basis,

basis\_values, not\_basis\_values)

num\_iteration += 1

if num\_iteration != 50:

print(f'Решение найдено! Общая прибыль составила {Q} денежных единиц')

else:

print('Поставленная задача решения не имеет')

### Приложение Д

Код реализации двойственной задачи на языке Python.

*Листинг Д.1. Реализация двойственной задачи.*

import re

import math

import sympy

import numpy as np

NUM\_CRITERIA = 4 # Количество ограничений в математической моделе

PRECISION = 8 # Количество знаков после запятой при округлении

SEP = 25 # Разделитель для вывода таблицы

def print\_table(system, coef\_basis, coef\_not\_basis, basis\_values, not\_basis\_values):

print(' '.ljust(SEP), end='')

print('Cj'.ljust(SEP), end='')

for i in range(len(coef\_not\_basis) + 1):

if i == len(coef\_not\_basis):

print(' '.ljust(SEP))

else:

print(str(coef\_not\_basis[i]).ljust(SEP), end='')

print('Cv'.ljust(SEP), end='')

for i in range(len(not\_basis\_values) + 1):

if i == 0:

print(''.ljust(SEP), end='')

else:

print(str(not\_basis\_values[i-1]).ljust(SEP), end='')

print('A0'.ljust(SEP))

system.insert(0, coef\_basis + [' '])

system.insert(1, basis\_values + ['f'])

for column in range(len(system[0])):

for row in range(len(system)):

print(str(system[row][column]).ljust(SEP), end='')

print()

del system[0]

del system[0]

def get\_coefficients(data):

'''Функция для получения списка коэффициентов из системы ограничений'''

criteria\_coefficients, boundaries = list(), list()

for exp in data:

if '<=' in exp:

parse = exp.split('<=')

elif '>=' in exp:

parse = exp.split('>=')

elif '<' in exp:

parse = exp.split('<')

elif '>' in exp:

parse = exp.split('>')

elif '=' in exp:

parse = exp.split('=')

parse = list(map(str.strip, parse))

boundaries.append(float(parse[1]))

criteria\_coefficients.append(list(map(float, [i.group(1) for i in re.finditer(

r'(\d+(\.\d+)?) {0,}[\*]? {0,}\w', parse[0])])))

*Продолжение Листинга Д.1.*

return criteria\_coefficients, boundaries

def count\_scalar\_product(vec1, vec2):

'''Функция для рассчёта скалярного произведения двух векторов'''

res = 0

for i in range(len(vec1)):

res += (vec1[i] \* vec2[i])

return res

def create\_simplex\_table(system, coef\_basis, coef\_not\_basis, basis\_values, not\_basis\_values):

F\_str = [0] \* len(not\_basis\_values)

for i in range(len(not\_basis\_values)):

F\_str[i] = count\_scalar\_product(

coef\_basis, system[i]) - coef\_not\_basis[i]

Q = count\_scalar\_product(coef\_basis, system[-1])

for i in range(len(F\_str)):

system[i].append(F\_str[i])

system[-1].append(Q)

return F\_str, Q

def simplex\_iteration(system, coef\_basis, coef\_not\_basis, basis\_values, not\_basis\_values, F\_str, Q):

index\_column = F\_str.index(min(F\_str))

mini = 1e10

for i in range(len(system[-1]) - 1):

tmp = system[-1][i] / system[index\_column][i]

if tmp < mini:

mini = tmp

index\_row = i

key\_element = system[index\_column][index\_row]

basis\_values.insert(index\_row, not\_basis\_values[index\_column])

not\_basis\_values.insert(index\_column, basis\_values.pop(index\_row + 1))

del not\_basis\_values[index\_row]

coef\_basis[index\_row], coef\_not\_basis[index\_column] = coef\_not\_basis[index\_column], coef\_basis[index\_row]

new\_key\_element = round(1 / key\_element, PRECISION)

data = [[0] \* (len(coef\_basis) + 1)

for \_ in range(len(coef\_not\_basis) + 1)]

for i in range(len(system[index\_column])):

data[index\_column][i] = - \

round(system[index\_column][i] / key\_element, PRECISION)

for i in range(len(system)):

data[i][index\_row] = round(

system[i][index\_row] / key\_element, PRECISION)

data[index\_column][index\_row] = new\_key\_element

for row in range(len(data[0])):

for column in range(len(data)):

if data[column][row] == 0:

data[column][row] = round(((system[column][row] \* key\_element) - (

system[index\_column][row] \* system[column][index\_row])) / key\_element, PRECISION)

F\_str = [data[i][-1] for i in range(len(data) - 1)]

Q = data[-1][-1]

return data, coef\_basis, coef\_not\_basis, basis\_values, not\_basis\_values, F\_str, Q

*Продолжение Листинга Д.1.*

def check\_inequality(inequality, variables, индексы\_базисных\_переменных\_оптимального\_плана):

inequality = inequality.replace('\*', '')

for i in range(len(variables)):

if variables[i] not in inequality:

continue

inequality = inequality.replace(

variables[i], '\*' + str(индексы\_базисных\_переменных\_оптимального\_плана[i]))

# Символьное вычисление неравенства

inequality += '- 0.1'

result = str(sympy.sympify(inequality))

# Возвращение True, если неравенство выполняется, иначе False

return eval(result)

def dual\_task():

коэффициенты\_целевой\_функции = np.array(target\_coefficients)

свободные\_члены\_неравенств = np.array(boundaries)

матрица\_ограничений, \_ = get\_coefficients(criteria\_function)

транспонированная\_матрица\_ограничений = np.transpose(матрица\_ограничений)

индексы\_базисных\_переменных\_оптимального\_плана = np.array(system[-1][:-1])

y = np.array([])

D = list()

for i in range(len(basis\_values)):

index = int(basis\_values[i][1:]) - 1

if index < len(транспонированная\_матрица\_ограничений):

D.append(транспонированная\_матрица\_ограничений[index])

else:

D.append(

np.array([1 if i == j else 0 for j in range(NUM\_CRITERIA)]))

D\_inversed = np.linalg.inv(np.transpose(D))

def first\_duality\_theorem():

y = np.dot(np.array(coef\_basis), D\_inversed)

G\_min = np.dot(свободные\_члены\_неравенств, y)

print(f"Gmin is {G\_min} by first\_duality\_theorem")

assert abs(G\_min - Q) < 0.00001

def second\_duality\_theorem():

nonlocal y

zeros = list()

for i in range(NUM\_CRITERIA):

if check\_inequality(

criteria\_function[i], basis\_values, индексы\_базисных\_переменных\_оптимального\_плана):

zeros.append(i)

система\_уравнений = транспонированная\_матрица\_ограничений.copy()

for i in range(len(zeros)):

система\_уравнений = np.delete(

система\_уравнений, zeros[i], 1)

for j in range(i + 1, len(zeros)):

zeros[j] -= 1

y = np.linalg.solve(система\_уравнений, коэффициенты\_целевой\_функции)

for i in range(len(zeros)):

y = np.insert(y, zeros[i], 0)

for j in range(i + 1, len(zeros)):

zeros[j] += 1

G\_min = np.dot(свободные\_члены\_неравенств, y)

*Продолжение Листинга Д.1.*

print(f"Gmin is {G\_min} by second\_duality\_theorem")

assert abs(G\_min - Q) < 0.00001

def third\_duality\_theorem():

нижняя\_граница = list()

верхняя\_граница = list()

b = list()

for i in range(len(D\_inversed) - 1, -1, -1):

pozitive = list()

negative = list()

bH = - math.inf

bB = math.inf

for j in range(len(D\_inversed)):

if D\_inversed[j][i] > 0:

pozitive.append(

(свободные\_члены\_неравенств[j], D\_inversed[j][i]))

elif D\_inversed[j][i] < 0:

negative.append(

(свободные\_члены\_неравенств[j], D\_inversed[j][i]))

if len(pozitive) > 1:

elem = min(pozitive, key=lambda x: abs(

pozitive[0][0] / pozitive[0][1]))

нижняя\_граница.append(elem[0] / elem[1])

elif len(pozitive) == 1:

нижняя\_граница.append(pozitive[0][0] / pozitive[0][1])

else:

нижняя\_граница.append(bH)

if len(negative) > 1:

elem = max(negative, key=lambda x: abs(

negative[0][0] / negative[0][1]))

верхняя\_граница.append(abs(elem[0] / elem[1]))

elif len(negative) == 1:

верхняя\_граница.append(negative[0][0] / negative[0][1])

else:

верхняя\_граница.append(bB)

b.append(свободные\_члены\_неравенств[i])

print(f'Ресурс №{len(D\_inversed)-i}')

print(

f'b{len(D\_inversed)-i} ∈ ({нижняя\_граница[-1]}; {верхняя\_граница[-1]})')

print(f'{len(D\_inversed)-i}-й ресурс изменяется в интервале: ', end='')

if нижняя\_граница[-1] == - math.inf:

print(f'({нижняя\_граница[-1]}; ', end='')

else:

print(f'({b[-1] - нижняя\_граница[-1]}; ', end='')

if верхняя\_граница[-1] == math.inf:

print(f'{верхняя\_граница[-1]})')

else:

print(f'{b[-1] + верхняя\_граница[-1]})')

total = 0

for i in range(len(y)):

if y[i] != 0:

total += y[i] \* верхняя\_граница[i]

print(f'∆Gmax{i + 1} = y{i+1} \* bB{i +

1} = {y[i] \* верхняя\_граница[i]}')

print(f'Совместное влияние изменений этих ресурсов приводит к изменению максимальной стоимости продукции 𝐺𝑚𝑎𝑥 на величину: {total}')

print(f'Следовательно, оптимальное значение целевой функции при

*Продолжение Листинга Д.1.*

максимальном изменении ресурсов: {Q+total}')

first\_duality\_theorem()

second\_duality\_theorem()

third\_duality\_theorem()

with open('TPR\_PRACT5.csv', encoding='utf-8') as file:

target\_function = file.readline().rstrip() # Целевая функция

target\_coefficients = list(map(float, [i.group(1) for i in re.finditer(

# Список коэффициентов целевой функции

r'(\d+(\.\d+)?) {0,}[\*]? {0,}\w', target\_function)]))

criteria\_function = [file.readline().rstrip() for \_ in range(NUM\_CRITERIA)]

criteria\_coefficients, boundaries = get\_coefficients(criteria\_function)

print('Переходим к задаче линейного программирования:',

target\_function, sep='\n')

for i in criteria\_function:

print("{ " + i)

system = list(map(list, list(zip(\*criteria\_coefficients))))

system.append(boundaries.copy())

# Вектор коэффициентов целевой функции при базисных переменных

coef\_basis = [0] \* NUM\_CRITERIA

# Коэффициенты целевой функции, соответствующие небазисным переменным

coef\_not\_basis = target\_coefficients.copy()

not\_basis\_values = re.findall(r'[A-Za-z]\d{1,}', target\_function)

basis\_values = [f'{not\_basis\_values[-1][0]}{i}' for i in range(

int(not\_basis\_values[-1][1]) + 1, NUM\_CRITERIA + int(not\_basis\_values[-1][1]) + 1)]

F\_str, Q = create\_simplex\_table(

system, coef\_basis, coef\_not\_basis, basis\_values, not\_basis\_values)

num\_iteration = 0

while num\_iteration < 50 and min(F\_str) < 0:

print(

('\x1b[6;30;42m' + f"Итерация №{num\_iteration}" + '\x1b[0m').center(201))

system, coef\_basis, coef\_not\_basis, basis\_values, not\_basis\_values, F\_str, Q = simplex\_iteration(

system, coef\_basis, coef\_not\_basis, basis\_values, not\_basis\_values, F\_str, Q)

print\_table(system, coef\_basis, coef\_not\_basis,

basis\_values, not\_basis\_values)

num\_iteration += 1

if num\_iteration != 50:

print(f'Решение найдено! Общая прибыль составила {

round(Q, 3)} денежных единиц')

else:

print('Поставленная задача решения не имеет')

exit(0)

dual\_task()

### Приложение Е

Код реализации транспортной задачи на языке Python.

*Листинг Е.1. Реализация транспортной задачи.*

import csv

import sympy

import copy

import re

import math

import numpy as np

from itertools import product

def transport\_task():

    '''Функция, отвечающая за решение закрытой транспортной задачи'''

    # поставщики, потребители, c = input\_data()

    поставщики, потребители, c = input\_data\_from\_file()

    assert sum(поставщики) == sum(

        потребители), 'Транспортная задача не является закрытой'

    C = np.vstack(c)  # Стоимости перевозок единицы груза из Ai в Bi

    X = np.zeros\_like(C)

    basis = list()  # Базисные переменные (заполненные клетки)

    U = np.zeros\_like(поставщики)  # Потенциалы пунктов Ai

    V = np.zeros\_like(потребители)  # Потенциалы пунктов Bj

    delta = np.zeros\_like(C)  # Относительные оценки клеток

    marks = list(product([f'u{i}' for i in range(

        1, len(поставщики) + 1)], [f'v{i}' for i in range(1, len(потребители) + 1)]))

    num\_iteration = 0  # Номер итерации в методе потенциалов

    def print\_table(potential=True):

        '''Функция для вывода таблицы'''

        print('-' \* (14 \* (len(потребители) + 2) + len(потребители) + 2))

        if potential:

            print(' ' \* 14, end='|')

            for i in range(len(потребители)):

                print(f'v{i + 1} = {V[i]}'.ljust(14), end='|')

            print(''.ljust(14), end='|')

            print('\n' + ' ' \* 14 + ('|' + '-' \* 14)

                  \* len(потребители), end='|')

            print(''.ljust(14), end='|')

            print()

        print('Пункты'.ljust(14), end='|')

        for i in range(len(потребители)):

            print(f'B{i + 1}'.ljust(14), end='|')

        print('Запасы'.ljust(14), end='|')

        for i in range(len(поставщики)):

            print('\n' + '-' \* (14 \* (len(потребители) + 2) + len(потребители) + 2))

            if potential:

                print(f'u{i + 1} = {U[i]}'.ljust(14), end='|')

            else:

                print(f'A{i + 1}'.ljust(14), end='|')

            for j in range(len(потребители)):

                print(f'{C[i][j]}'.rjust(14), end='|')

            print('\033[94m' + f'{поставщики[i]

                                  }'.ljust(14) + '\033[0m', end='|')

            if potential:

*Продолжение Листинга Е.1.*

                print('\n' + f'A{i + 1}'.ljust(14), end='|')

            else:

                print('\n' + ' ' \* 14, end='|')

            for j in range(len(потребители)):

                if (i, j) not in basis:

                    print(f''.ljust(14), end='|')

                else:

                    print('\033[102m' +

                          f'{X[i][j]}'.ljust(14) + '\033[0m', end='|')

            print(''.ljust(14), end='|')

        print('\n' + '-' \* (14 \* (len(потребители) + 2) + len(потребители) + 2))

        print('Потребности'.ljust(14), end='|')

        for i in range(len(потребители)):

            print('\033[94m' + f'{потребители[i]

                                  }'.ljust(14) + '\033[0m', end='|')

        print('\033[95m' + f'{sum(потребители)

                              }'.ljust(14) + '\033[0m', end='|')

        print('\n' + '-' \* (14 \* (len(потребители) + 2) +

              len(потребители) + 2))

    def northwest\_corner\_method():

        '''Метод северо-западного угла нахождения начального опорного решения'''

        for i in range(len(поставщики)):

            for j in range(len(потребители)):

                X[i][j] = min(поставщики[i] - np.sum(X[i, :]),

                              потребители[j] - np.sum(X[:, j]))

                if X[i][j] != 0:

                    basis.append((i, j))

    def get\_min\_indexes(suppliers, consumers):

        '''Вспомогательная функция для определения индексов ячейки с минимальной стоимостью'''

        min\_cost = np.inf  # хранит наименьший элемент, который нашли на текущий момент в матрице С

        min\_indexes = (None, None)  # индексы наименьшего элемента

        for i in range(len(C)):

            for j in range(len(C[0])):

                if min\_cost > C[i][j] and C[i][j] > 0 and X[i][j] == 0:

                    if suppliers[i] > 0 and consumers[j] > 0:

                        # назначаем (i, j) элемент новым наименьшим

                        min\_cost = C[i][j]

                        min\_indexes = (i, j)

        return min\_indexes

    def min\_price\_method():

        suppliers = np.copy(поставщики)  # Копия вектора поставщиков

        consumers = np.copy(потребители)  # Копия вектора потребителей

        '''Метод минимальной стоимости нахождения опорного решения'''

        while True:

            i, j = get\_min\_indexes(suppliers, consumers)

            if i is None and j is None:

                # все потребности удовлетворены и/или все возможности использованы

                break

            resources = min(suppliers[i], consumers[j])

            suppliers[i] = suppliers[i] - resources

            consumers[j] = consumers[j] - resources

            X[i][j] = resources

            basis.append((i, j))

*Продолжение Листинга Е.1.*

    def count\_function():

        '''Расчёт стоимости перевозок по текущему плану'''

        return np.dot(C.reshape(len(поставщики) \* len(потребители)), X.reshape(len(поставщики) \* len(потребители)))

    def count\_potentials(calculation\_output=False):

        ''' Функция для подсчёта потенциалов Ui, Vi'''

        system = []

        for u, v in marks:

            if (int(u[1]) - 1, int(v[1]) - 1) in basis:

                system.append(f'{u} + {v} = {C[int(u[1]) - 1][int(v[1]) - 1]}')

        if calculation\_output:

            print("Вычислим потенциалы ui и vi, исходя из базисных переменных. Для их нахождения используем условия ui + vj = cij")

            print(\*system, sep='\n')

        first\_equation = system[0]

        first\_variable = first\_equation.split()[0]

        variables = set()

        for equation in system:

            for symbol in equation.split():

                if re.fullmatch(r'[uv]\d\*', symbol):

                    variables.add(symbol)

        symbols\_dict = {symbol: sympy.symbols(symbol) for symbol in variables}

        system[0] = first\_equation.replace(first\_variable, '0')

        equations = []

        for equation in system:

            left, right = equation.split('=')

            equations.append(

                sympy.Eq(sympy.sympify(left), sympy.sympify(right)))

        equations.append(sympy.Eq(symbols\_dict[first\_variable], 0))

        solution = sympy.solve(equations, list(symbols\_dict.values()))

        for key, value in solution.items():

            if str(key)[0] == 'u':

                U[int(str(key)[1]) - 1] = value

            else:

                V[int(str(key)[1]) - 1] = value

        if calculation\_output:

            print(f'Считая, что {first\_variable} = 0, имеем:')

            print(\*[f'{key} = {value}' for key,

                  value in solution.items()], sep='; ')

    def count\_delta(calculation\_output=False):

        nonlocal delta

        if calculation\_output:

            print('Для каждой свободной клетки вычислим относительные оценки:')

        delta = np.zeros\_like(C)

        for i in range(len(поставщики)):

            for j in range(len(потребители)):

                if (i, j) not in basis:

                    delta[i][j] = C[i][j] - (U[i] + V[j])

                    if count\_delta:

                        print(f'Δ{i + 1}{j + 1} = {delta[i][j]};')

    def recalculate\_optimal\_plan():

        '''Функия перерасчёта оптимального плана'''

        min\_i, min\_j = math.inf, math.inf

        lowest\_mark = math.inf

        for i in range(len(поставщики)):

*Продолжение Листинга Е.1.*

            for j in range(len(потребители)):

                if delta[i][j] < lowest\_mark:

                    lowest\_mark = delta[i][j]

                    min\_i = i

                    min\_j = j

        basis.append((min\_i, min\_j))

        available\_ways = copy.copy(basis)

        cycle = [(min\_i, min\_j)]

        curr\_i = min\_i

        curr\_j = min\_j

        while True:

            dead\_end = True

            for i, j in available\_ways:

                if i == curr\_i and j != curr\_j and ((i, j) not in cycle or (i, j) == (min\_i, min\_j)):

                    if len(cycle) > 1 and cycle[-2][0] != i:

                        dead\_end = False

                        curr\_j = j

                        break

                    elif len(cycle) == 1:

                        dead\_end = False

                        curr\_j = j

                        break

                elif i != curr\_i and j == curr\_j and ((i, j) not in cycle or (i, j) == (min\_i, min\_j)):

                    if len(cycle) > 1 and cycle[-2][1] != j:

                        dead\_end = False

                        curr\_i = i

                        break

                    elif len(cycle) == 1:

                        dead\_end = False

                        curr\_i = i

                        break

            if not dead\_end:

                cycle.append((curr\_i, curr\_j))

            elif cycle[0] != cycle[-1]:

                del available\_ways[available\_ways.index((curr\_i, curr\_j))]

                curr\_i = min\_i

                curr\_j = min\_j

                cycle = [(min\_i, min\_j)]

            else:

                break

        lam = math.inf

        for index, point in enumerate(cycle[:-1]):

            i, j = point

            if index % 2 and X[i][j] < lam:

                lam = X[i][j]

        deleted\_from\_basis = 0

        for index, point in enumerate(cycle[:-1]):

            i, j = point

            if index % 2:

                X[i][j] -= lam

            else:

                X[i][j] += lam

            if X[i][j] == 0 and deleted\_from\_basis == 0:

                del basis[basis.index((i, j))]

                deleted\_from\_basis += 1

    print('Таблица с исходными данными:')

    print\_table(False)

*Продолжение Листинга Е.1.*

    min\_price\_method()

    print('Начальный опорный план, полученный методом минимальной стоимости:')

    print\_table()

    print(f'Стоимость перевозок по этому плану: {count\_function()} единиц')

    X = np.zeros\_like(C)  # Очитка массива X

    basis.clear()  # Очитка базиса

    northwest\_corner\_method()

    print('\033[101m' + 'Начальный опорный план, полученный методом северо-западного угла:' + '\033[0m')

    print\_table(False)

    print(f'Стоимость перевозок по этому плану: {count\_function()} единиц')

    count\_potentials(True)

    print\_table()

    count\_delta(True)

    while np.min(delta) < 0 and num\_iteration < 20:

        print('Остались отрицательные оценки, произведём перерасчёт плана...')

        num\_iteration += 1

        print('\033[101m' + f'Итерация № {num\_iteration}:' + '\033[0m')

        recalculate\_optimal\_plan()

        print\_table()

        print(f'Стоимость перевозок по этому плану: {count\_function()} единиц')

        count\_potentials(True)

        count\_delta(True)

    if num\_iteration == 20:

        print(f'За {num\_iteration} не удалось найти оптимальное решение(')

    else:

        print('\033[101m' + 'Оптимальное решение найдено!' + '\033[0m')

def input\_data():

    '''Ручной ввод данных'''

    поставщики = np.array(list(map(int, input(

        "Введите запас груза у каждого поставщика через пробел: ").split())))

    потребители = np.array(list(map(int, input(

        "Введите потребность груза у каждого потребителя через пробел: ").split())))

    c = list()  # Коэффициенты Cij

    for i in range(len(поставщики)):

        c.append(np.array(list(map(int, input(f"Введите стоимости перевозки от {

                 i + 1}-ого поставщика к каждому потребителю через пробел: ").split()))))

    return поставщики, потребители, c

def input\_data\_from\_file():

    '''Автоматический ввод данных из файла'''

    with open('TPR\_PRACT7.csv', encoding='utf-8') as file:

        rows = csv.reader(file)

        поставщики = np.array(list(map(int, next(rows))))

        потребители = np.array(list(map(int, next(rows))))

        c = list()  # Коэффициенты Cij

        for row in rows:

            c.append(np.array(list(map(int, row))))

    return поставщики, потребители, c

transport\_task()