

# СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 7](#_Toc166522981)

[1 МЕТОД ПАРЕТО 9](#_Toc166522982)

[1.1 Введение 9](#_Toc166522983)

[1.2 Выбор Парето-оптимального множества 9](#_Toc166522984)

[1.3 Указание верхних/нижних границ критериев 12](#_Toc166522985)

[1.4 Субоптимизация 13](#_Toc166522986)

[1.5 Лексикографическая оптимизация 14](#_Toc166522987)

[1.6 Результаты работы программы 15](#_Toc166522988)

[1.7 Выводы по разделу 16](#_Toc166522989)

[2 МЕТОД ЭЛЕКТРА II 17](#_Toc166522990)

[2.1 Введение 17](#_Toc166522991)

[2.2 Выбор лучшего варианта 17](#_Toc166522992)

[2.3 Веса предпочтений 19](#_Toc166522993)

[2.4 Вывод 42](#_Toc166522994)

[2.5 Результат работы программы 43](#_Toc166522995)

[2.6 Выводы по разделу 44](#_Toc166522996)

[3 МЕТОД АНАЛИЗА ИЕРАРХИЙ 45](#_Toc166522997)

[3.1 Введение 45](#_Toc166522998)

[3.2 Постановка задачи 45](#_Toc166522999)

[3.3 Представление проблемы в виде иерархии 45](#_Toc166523000)

[3.4 Установка приоритетов критериев 47](#_Toc166523001)

[3.5 Синтез приоритетов 47](#_Toc166523002)

[3.6 Согласованность локальных приоритетов 55](#_Toc166523003)

[3.7 Синтез альтернатив 62](#_Toc166523004)

[3.8 Вывод 63](#_Toc166523005)

[3.9 Результаты работы программы 64](#_Toc166523006)

[3.10 Выводы по разделу 66](#_Toc166523007)

[4 ГРАФИЧЕСКИЙ МЕТОД 67](#_Toc166523008)

[4.1 Введение 67](#_Toc166523009)

[4.2 Постановка задачи 67](#_Toc166523010)

[4.3 Данные индивидуального варианта 67](#_Toc166523011)

[4.4 Подготовка данных 67](#_Toc166523012)

[4.5 Построение графика 68](#_Toc166523013)

[4.6 Выделение области допустимых решений 70](#_Toc166523014)

[4.7 Максимум функции 71](#_Toc166523015)

[4.8 Минимум функции 72](#_Toc166523016)

[4.9 Результат работы программы 74](#_Toc166523017)

[4.10 Выводы по разделу 74](#_Toc166523018)

[5 СИМПЛЕКСНЫЙ МЕТОД 75](#_Toc166523019)

[5.1 Введение 75](#_Toc166523020)

[5.2 Постановка задачи 75](#_Toc166523021)

[5.3 Математическая модель задачи 76](#_Toc166523022)

[5.4 Решение задачи 77](#_Toc166523023)

[5.5 Пример работы программы 87](#_Toc166523024)

[5.6 Выводы по разделу 88](#_Toc166523025)

[6 ДВОЙСТВЕННАЯ ЗАДАЧА 89](#_Toc166523026)

[6.1 Введение 89](#_Toc166523027)

[6.2 Постановка задачи 89](#_Toc166523028)

[6.3 Математическая модель 90](#_Toc166523029)

[6.4 Соответствующая исходной двойственная задача 90](#_Toc166523030)

[6.5 Первая теорема двойственности 91](#_Toc166523031)

[6.6 Вторая теорема двойственности 94](#_Toc166523032)

[6.7 Третья теорема двойственности 97](#_Toc166523033)

[6.8 Результаты работы программы 102](#_Toc166523034)

[6.9 Выводы по разделу 104](#_Toc166523035)

[7 ТРАНСПОРТНАЯ ЗАДАЧА 105](#_Toc166523036)

[7.1 Введение 105](#_Toc166523037)

[7.2 Постановка задачи 105](#_Toc166523038)

[7.3 Математическая модель транспортной задачи 106](#_Toc166523039)

[7.4 Метод северо-западного угла 106](#_Toc166523040)

[7.5 Метод минимальной стоимости 107](#_Toc166523041)

[7.6 Метод потенциалов 108](#_Toc166523042)

[7.7 Результаты выполнения программы 117](#_Toc166523043)

[7.8 Выводы по разделу 122](#_Toc166523044)

[ЗАКЛЮЧЕНИЕ 123](#_Toc166523045)

[СПИСОК ИНФОРМАЦИОННЫХ ИСТОЧНИКОВ 125](#_Toc166523046)

[ПРИЛОЖЕНИЯ 126](#_Toc166523047)

# 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: изучить метод Парето и применить его для нахождения оптимальной альтернативы в заданной предметной области.

Предметная область: выбор идеальной популярной песни.

Метод Парето применяется в задачах многокритериальной оптимизации, т.е. если альтернативы нужно сравнивать по двум и более критериям. Например, к таким задачам относится задача выбора оптимальной квартиры, где критериями выступают цена, удалённость от метро, площадь и т.д.

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 оптимальную популярную песню. Критерии выбора:

1. Количество слушателей – взято из стримингового сервиса Spotify.
2. Рейтинг песни – взят с сайта albumoftheyear.org.
3. Темп – взят с сайта tunebat.com.
4. Длина – взята с сайта tunebat.com.
5. Агрессивность лирики – взята на основе информации с сайта genius.com.
6. Громкость – взята с сайта tunebat.com.

Далее было выбрано 14 альтернатив, для которых найдены значения по каждому критерию, и из этого была составлена Таблица 1.2.1. Далее по этой таблице была составлена Таблица 1.2.2, где альтернативы сравниваются друг с другом.

Таблица 1.2.1 – Альтернативы и критерии

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| № | Альтернативы | Критерии | | | | | |
| Количество слушателей (млн.) (+) | Рейтинг песни (+) | Темп песни (BPM) (+) | Длина песни  (-) | Агрессивность лирики (-) | Громкость (dB)  (+) |
| 1 | JAY-Z, Kanye West – N\*\*\*\*s In Paris | 1352 | 93 | 140 | 3:39 | 8 | -6 |
| 2 | Lady Gaga – Poker Face | 1179 | 95 | 119 | 3:57 | 5 | -5 |
| 3 | Taylor Swift - Cruel Summer | 1744 | 90 | 170 | 2:58 | 1 | -6 |
| 4 | The Weeknd - Blinding Lights | 4072 | 94 | 170 | 3:20 | 1 | -6 |
| 5 | Metallica – Enter Sandman | 1302 | 92 | 123 | 5:32 | 2 | -8 |
| 6 | Nirvana – Smells Like Teen Spirit | 1895 | 97 | 117 | 5:02 | 2 | -5 |
| 7 | Kendrick Lamar – HUMBLE. | 2103 | 89 | 150 | 2:57 | 9 | -7 |
| 8 | Tones And I – Dance Monkey | 2994 | 6 | 98 | 3:29 | 2 | -6 |

Продолжение Таблицы 1.2.1

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 9 | Gorillaz – Feel Good Inc. | 1313 | 97 | 139 | 3:43 | 1 | -7 |
| 10 | Michael Jackson – Billie Jean | 1632 | 98 | 117 | 4:54 | 2 | -3 |
| 11 | Linkin Park – In The End | 1872 | 94 | 105 | 3:37 | 1 | -6 |
| 12 | Post Malone – rockstar | 2881 | 87 | 160 | 3:38 | 7 | -6 |
| 13 | Ed Sheeran – Shape Of You | 3789 | 44 | 96 | 3:54 | 3 | -3 |
| 14 | Imagine Dragons - Believer | 2857 | 50 | 125 | 3:24 | 2 | -4 |

Таблица 1.2.2 – Сравнения альтернатив

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | А1 | А2 | А3 | А4 | А5 | А6 | А7 | А8 | А9 | А10 | А11 | А12 | А13 | А14 |
| А1 | x | x | x | x | x | x | x | x | x | x | x | x | x | x |
| А2 | н | x | x | x | x | x | x | x | x | x | x | x | x | x |
| А3 | н | н | x | x | x | x | x | x | x | x | x | x | x | x |
| А4 | А4 | н | н | x | x | x | x | x | x | x | x | x | x | x |
| А5 | н | н | н | А4 | x | x | x | x | x | x | x | x | x | x |
| А6 | н | н | н | н | н | x | x | x | x | x | x | x | x | x |
| А7 | н | н | н | н | н | н | x | x | x | x | x | x | x | x |
| А8 | н | н | н | А4 | н | н | н | x | x | x | x | x | x | x |
| А9 | н | н | н | н | А9 | н | н | н | x | x | x | x | x | x |
| А10 | н | н | н | н | н | н | н | н | н | x | x | x | x | x |
| А11 | н | н | н | А4 | н | н | н | н | н | н | x | x | x | x |
| А12 | н | н | н | А4 | н | н | н | н | н | н | н | x | x | x |
| А13 | н | н | н | н | н | н | н | н | н | н | н | н | x | x |
| А14 | н | н | н | н | н | н | н | н | н | н | н | н | н | x |

Из таблицы получается, что Парето-оптимальное множество: {А2, А3, А4, А6, А7, А8, А9, А10, А13, А14} (Таблица 1.2.3).

Таблица 1.2.3 – Парето-оптимальное множество

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| № | Альтернативы | Критерии | | | | | |
| Количество слушателей (млн.) (+) | Рейтинг песни (+) | Темп песни (BPM) (+) | Длина песни  (-) | Агрессивность лирики (-) | Громкость (dB)  (+) |
| 2 | Lady Gaga – Poker Face | 1179 | 95 | 119 | 3:57 | 5 | -5 |
| 3 | Taylor Swift - Cruel Summer | 1744 | 90 | 170 | 2:58 | 1 | -6 |

Продолжение Таблицы 1.2.3

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 4 | The Weeknd - Blinding Lights | 4072 | 94 | 170 | 3:20 | 1 | -6 |
| 6 | Nirvana – Smells Like Teen Spirit | 1895 | 97 | 117 | 5:02 | 2 | -5 |
| 7 | Kendrick Lamar – HUMBLE. | 2103 | 89 | 150 | 2:57 | 9 | -7 |
| 8 | Tones And I – Dance Monkey | 2994 | 6 | 98 | 3:29 | 2 | -6 |
| 9 | Gorillaz – Feel Good Inc. | 1313 | 97 | 139 | 3:43 | 1 | -7 |
| 10 | Michael Jackson – Billie Jean | 1632 | 98 | 117 | 4:54 | 2 | -3 |
| 13 | Ed Sheeran – Shape Of You | 3789 | 44 | 96 | 3:54 | 3 | -3 |
| 14 | Imagine Dragons - Believer | 2857 | 50 | 125 | 3:24 | 2 | -4 |

Из полученной таблицы следует очевидный недостаток метода Парето – оптимальное множество состоит из нескольких альтернатив, и выбор какой-то одной альтернативы остаётся за ЛПР. Плюсом является простота расчётов в методе.

## 1.3 Указание верхних/нижних границ критериев

Для того, чтобы убрать часть альтернатив из рассмотрения, были поставлены границы на часть критериев:

1. Рейтинг должен быть не ниже 90.
2. Длина песни должна быть меньше 4 минут.
3. Количество прослушиваний должно быть больше 1500 млн.

Варианты, удовлетворяющие этим границам, представлены в Таблице 1.3.1.

Таблица 1.3.1 – Результат указания границ критериев

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| № | Альтернативы | Критерии | | | | | |
| Количество слушателей (млн.) (+) | Рейтинг песни (+) | Темп песни (BPM) (+) | Длина песни  (-) | Агрессивность лирики (-) | Громкость (dB)  (+) |
| 3 | Taylor Swift - Cruel Summer | 1744 | 90 | 170 | 2:58 | 1 | -6 |
| 4 | The Weeknd - Blinding Lights | 4072 | 94 | 170 | 3:20 | 1 | -6 |
| 11 | Linkin Park – In The End | 1872 | 94 | 105 | 3:37 | 1 | -6 |

Альтернатива А11 не является оптимальной по Парето, т.к. альтернатива А4 доминирует над ней, следовательно, Парето-оптимальное множество состоит из А3 и А4.

Как и в предыдущем методе, не получилось выбрать одну оптимальную альтернативу, но в этом методе размер множества получился меньше.

## 1.4 Субоптимизация

Главный критерий: количество слушателей. Для двух других критериев были установлены границы:

1. Рейтинг песни не ниже 90.
2. Длина песни меньше 4 минут.

Применив границы для критериев, останутся следующие альтерантивы (Таблица 1.4.1)

Таблица 1.4.1 – Альтернативы и критерии

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| № | Альтернативы | Критерии | | | | | |
| Количество слушателей (млн.) (+) | Рейтинг песни (+) | Темп песни (BPM) (+) | Длина песни  (-) | Агрессивность лирики (-) | Громкость (dB)  (+) |
| 1 | JAY-Z, Kanye West – N\*\*\*\*s In Paris | 1352 | 93 | 140 | 3:39 | 8 | -6 |
| 2 | Lady Gaga – Poker Face | 1179 | 95 | 119 | 3:57 | 5 | -5 |
| 3 | Taylor Swift - Cruel Summer | 1744 | 90 | 170 | 2:58 | 1 | -6 |
| 4 | The Weeknd - Blinding Lights | 4072 | 94 | 170 | 3:20 | 1 | -6 |
| 9 | Gorillaz – Feel Good Inc. | 1313 | 97 | 139 | 3:43 | 1 | -7 |
| 11 | Linkin Park – In The End | 1872 | 94 | 105 | 3:37 | 1 | -6 |

Среди оставшихся альтернатив максимальное количество слушателей у А4, следовательно, она является оптимальной.

Преимуществом субоптимизации является гарантированное получение единственного решения, однако выделение одного из критериев и установка границ для других критериев вносят субъективный характер в принимаемое решение.

## 1.5 Лексикографическая оптимизация

Приоритеты критериев: рейтинг песни; количество слушателей; длина песни; темп песни; количество нецензурных слов.

Наибольший рейтинг (98) есть только у альтернативы А10. Следовательно, эта альтернатива является оптимальной (Таблица 1.5.1).

Таблица 1.5.1 – Результат лексикографической оптимизации

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| № | Альтернативы | Критерии | | | | | |
| Количество слушателей (млн.) (+) | Рейтинг песни (+) | Темп песни (BPM) (+) | Длина песни  (-) | Агрессивность лирики (-) | Громкость (dB)  (+) |
| 10 | Michael Jackson – Billie Jean | 1632 | 98 | 117 | 4:54 | 2 | -3 |

Плюсом метода является выделение единственной оптимальной альтернативы, однако это происходит за счёт выделения какого-то одного или нескольких критериев, а остальные могут вообще не рассматриваться.

## 1.6 Результаты работы программы

Метод Парето и методы его оптимизации были реализованы в программе, результаты работы которой представлены на Рисунках 1.6.1 – 1.6.4.

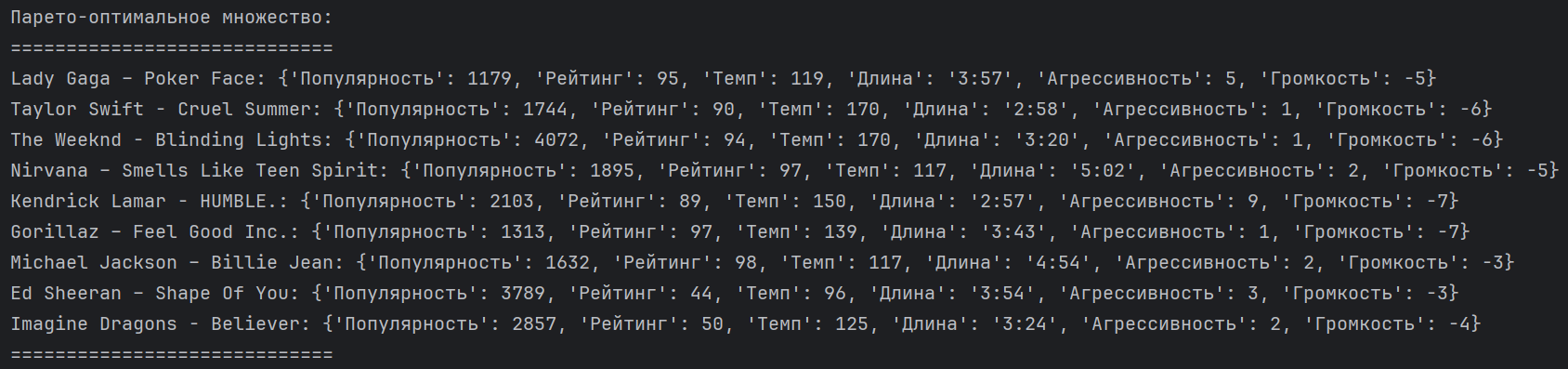


Рисунок 1.6.1 – Парето-оптимальное множество

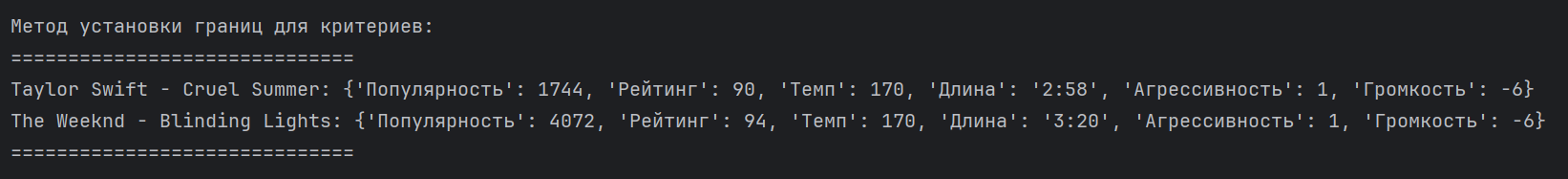


Рисунок 1.6.2 – Установки верхних/нижних границ для критериев

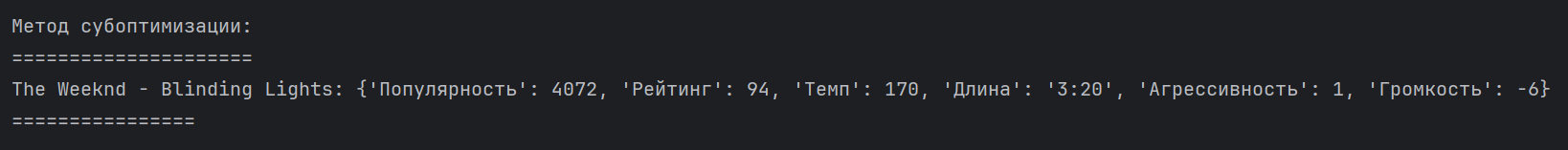


Рисунок 1.6.3 – Метод субоптимизации

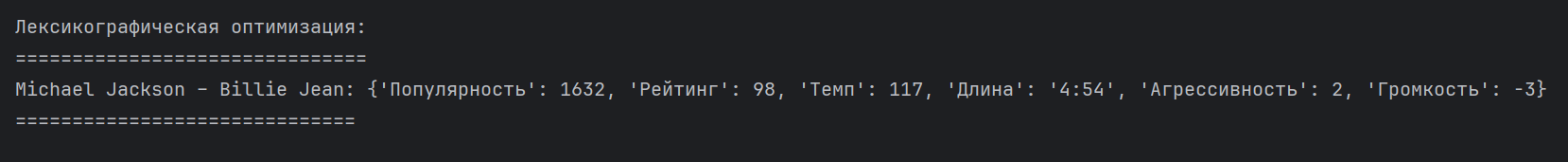


Рисунок 1.6.4 – Лексикографическая оптимизация

## 1.7 Выводы по разделу

В ходе выполнения данной практической работы изучен метод Парето, применён на практике для определения оптимального решения в задаче нахождения идеальной популярной песни, 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 идеальной популярной песни.

Метод Электра II состоит из нескольких этапов. На первом этапе каждому критерию устанавливается определённый вес, характеризующий его важность, а его значения разделяются на диапазоны, которые затем кодируются. 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 циклы, то назначается порог, который отбрасывает слабые связи, т.е. те пары альтернатив, которые не сильно отличаются друг от друга. Если в графе не осталось циклов и граф остался целостным, то выбираются те альтернативы, к которым не идёт ни одно ребро на графе. Они являются оптимальными.

## 2.2 Выбор лучшего варианта

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

Таблица 2.2.1 – Таблица критериев для оценки альтернатив

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Критерии | Вес критерия | Шкала | Код | Стремление |
| Количество слушателей (млн.) (+) | 4 | Более 2500 млн прослушиваний  От 1500 до 2500 млн прослушиваний  До 1500 млн прослушиваний | 15  10  5 | max |
| Рейтинг (+) | 6 | Больше или равно 95  От 80 до 95  Меньше 80 | 15  10  5 | max |
| Темп (BPM) (+) | 2 | Больше или равно 120 BPM  Меньше 120 BPM | 10  5 | max |
| Длина (мин.:сек.) (-) | 2 | Больше 4 минут  Меньше или равно 4 минут | 10  5 | min |
| Агрессивность лирики (-) | 1 | Больше 7  От 4 до 7  Меньше 4 | 15  10  5 | min |
| Громкость (dB) (+) | 3 | Больше -6  Меньше или равно -6 | 10  5 | max |

Составлена таблица оценок выбора лучшей популярной песни. Для 14-ти альтернатив заполнена Таблицу 2.2.2.

Таблица 2.2.2 – Таблица оценок по критериям

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| № | Альтернативы | Критерии | | | | | |
| Количество слушателей (млн.) (+) | Рейтинг песни (+) | Темп песни (BPM) (+) | Длина песни  (-) | Агрессивность лирики (-) | Громкость (dB)  (+) |
| 1 | JAY-Z, Kanye West – N\*\*\*\*s In Paris | 1352 | 93 | 140 | 3:39 | 8 | -6 |
| 2 | Lady Gaga – Poker Face | 1179 | 95 | 119 | 3:57 | 5 | -5 |

Продолжение Таблицы 2.2.2

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 3 | Taylor Swift - Cruel Summer | 1744 | 90 | 170 | 2:58 | 1 | -6 |
| 4 | The Weeknd - Blinding Lights | 4072 | 94 | 170 | 3:20 | 1 | -6 |
| 5 | Metallica – Enter Sandman | 1302 | 92 | 123 | 5:32 | 2 | -8 |
| 6 | Nirvana – Smells Like Teen Spirit | 1895 | 97 | 117 | 5:02 | 2 | -5 |
| 7 | Kendrick Lamar – HUMBLE. | 2103 | 89 | 150 | 2:57 | 9 | -7 |
| 8 | Tones And I – Dance Monkey | 2994 | 6 | 98 | 3:29 | 2 | -6 |
| 9 | Gorillaz – Feel Good Inc. | 1313 | 97 | 139 | 3:43 | 1 | -7 |
| 10 | Michael Jackson – Billie Jean | 1632 | 98 | 117 | 4:54 | 2 | -3 |
| 11 | Linkin Park – In The End | 1872 | 94 | 105 | 3:37 | 1 | -6 |
| 12 | Post Malone – rockstar | 2881 | 87 | 160 | 3:38 | 7 | -6 |
| 13 | Ed Sheeran – Shape Of You | 3789 | 44 | 96 | 3:54 | 3 | -3 |
| 14 | Imagine Dragons - Believer | 2857 | 50 | 125 | 3:24 | 2 | -4 |

## 2.3 Веса предпочтений

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

P12 = 0 + 0 + 2 + 0 + 0 + 0 = 2;

N12 = 0 + 6 + 0 + 0 + 1 + 3 = 10;

D12 = P12 / N12 = 2/10 = 0.2 < 1 - отбрасываем.

P21 = 0 + 6 + 0 + 0 + 1 + 3 = 10;

N21 = 0 + 0 + 2 + 0 + 0 + 0 = 2;

D21 = P21 / N21 = 10/2 = 5.0 > 1 - принимаем.

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

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

N13 = 4 + 0 + 0 + 0 + 1 + 0 = 5;

D13 = P13 / N13 = 0/5 = 0.0 < 1 - отбрасываем.

P31 = 4 + 0 + 0 + 0 + 1 + 0 = 5;

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

D31 = P31 / N31 = 5/0 = ∞ > 1 - принимаем.

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

P14 = 0 + 0 + 0 + 0 + 0 + 0 = 0;

N14 = 4 + 0 + 0 + 0 + 1 + 0 = 5;

D14 = P14 / N14 = 0/5 = 0.0 < 1 - отбрасываем.

P41 = 4 + 0 + 0 + 0 + 1 + 0 = 5;

N41 = 0 + 0 + 0 + 0 + 0 + 0 = 0;

D41 = P41 / N41 = 5/0 = ∞ > 1 - принимаем.

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

P15 = 0 + 0 + 0 + 2 + 0 + 0 = 2;

N15 = 0 + 0 + 0 + 0 + 1 + 0 = 1;

D15 = P15 / N15 = 2/1 = 2.0 > 1 - принимаем.

P51 = 0 + 0 + 0 + 0 + 1 + 0 = 1;

N51 = 0 + 0 + 0 + 2 + 0 + 0 = 2;

D51 = P51 / N51 = 1/2 = 0.5 < 1 - отбрасываем.

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

P16 = 0 + 0 + 2 + 2 + 0 + 0 = 4;

N16 = 4 + 6 + 0 + 0 + 1 + 3 = 14;

D16 = P16 / N16 = 4/14 = 0.29 < 1 - отбрасываем.

P61 = 4 + 6 + 0 + 0 + 1 + 3 = 14;

N61 = 0 + 0 + 2 + 2 + 0 + 0 = 4;

D61 = P61 / N61 = 14/4 = 3.5 > 1 - принимаем.

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

P17 = 0 + 0 + 0 + 0 + 0 + 0 = 0;

N17 = 4 + 0 + 0 + 0 + 0 + 0 = 4;

D17 = P17 / N17 = 0/4 = 0.0 < 1 - отбрасываем.

P71 = 4 + 0 + 0 + 0 + 0 + 0 = 4;

N71 = 0 + 0 + 0 + 0 + 0 + 0 = 0;

D71 = P71 / N71 = 4/0 = ∞ > 1 - принимаем.

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

P18 = 0 + 6 + 2 + 0 + 0 + 0 = 8;

N18 = 4 + 0 + 0 + 0 + 1 + 0 = 5;

D18 = P18 / N18 = 8/5 = 1.6 > 1 - принимаем.

P81 = 4 + 0 + 0 + 0 + 1 + 0 = 5;

N81 = 0 + 6 + 2 + 0 + 0 + 0 = 8;

D81 = P81 / N81 = 5/8 = 0.62 < 1 - отбрасываем.

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

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

N19 = 0 + 6 + 0 + 0 + 1 + 0 = 7;

D19 = P19 / N19 = 0/7 = 0.0 < 1 - отбрасываем.

P91 = 0 + 6 + 0 + 0 + 1 + 0 = 7;

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

D91 = P91 / N91 = 7/0 = ∞ > 1 - принимаем.

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

P110 = 0 + 0 + 2 + 2 + 0 + 0 = 4;

N110 = 4 + 6 + 0 + 0 + 1 + 3 = 14;

D110 = P110 / N110 = 4/14 = 0.29 < 1 - отбрасываем.

P101 = 4 + 6 + 0 + 0 + 1 + 3 = 14;

N101 = 0 + 0 + 2 + 2 + 0 + 0 = 4;

D101 = P101 / N101 = 14/4 = 3.5 > 1 - принимаем.

Рассмотрим альтернативы 1 и 11 (i=1,j=11):

P111 = 0 + 0 + 2 + 0 + 0 + 0 = 2;

N111 = 4 + 0 + 0 + 0 + 1 + 0 = 5;

D111 = P111 / N111 = 2/5 = 0.4 < 1 - отбрасываем.

P111 = 4 + 0 + 0 + 0 + 1 + 0 = 5;

N111 = 0 + 0 + 2 + 0 + 0 + 0 = 2;

D111 = P111 / N111 = 5/2 = 2.5 > 1 - принимаем.

Рассмотрим альтернативы 1 и 12 (i=1,j=12):

P112 = 0 + 0 + 0 + 0 + 0 + 0 = 0;

N112 = 4 + 0 + 0 + 0 + 0 + 0 = 4;

D112 = P112 / N112 = 0/4 = 0.0 < 1 - отбрасываем.

P121 = 4 + 0 + 0 + 0 + 0 + 0 = 4;

N121 = 0 + 0 + 0 + 0 + 0 + 0 = 0;

D121 = P121 / N121 = 4/0 = ∞ > 1 - принимаем.

Рассмотрим альтернативы 1 и 13 (i=1,j=13):

P113 = 0 + 6 + 2 + 0 + 0 + 0 = 8;

N113 = 4 + 0 + 0 + 0 + 1 + 3 = 8;

D113 = P113 / N113 = 8/8 = 1 - отбрасываем.

P131 = 4 + 0 + 0 + 0 + 1 + 3 = 8;

N131 = 0 + 6 + 2 + 0 + 0 + 0 = 8;

D131 = P131 / N131 = 8/8 = 1 - отбрасываем.

Рассмотрим альтернативы 1 и 14 (i=1,j=14):

P114 = 0 + 6 + 0 + 0 + 0 + 0 = 6;

N114 = 4 + 0 + 0 + 0 + 1 + 3 = 8;

D114 = P114 / N114 = 6/8 = 0.75 < 1 - отбрасываем.

P141 = 4 + 0 + 0 + 0 + 1 + 3 = 8;

N141 = 0 + 6 + 0 + 0 + 0 + 0 = 6;

D141 = P141 / N141 = 8/6 = 1.33 > 1 - принимаем.

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

P23 = 0 + 6 + 0 + 0 + 0 + 3 = 9;

N23 = 4 + 0 + 2 + 0 + 1 + 0 = 7;

D23 = P23 / N23 = 9/7 = 1.29 > 1 - принимаем.

P32 = 4 + 0 + 2 + 0 + 1 + 0 = 7;

N32 = 0 + 6 + 0 + 0 + 0 + 3 = 9;

D32 = P32 / N32 = 7/9 = 0.78 < 1 - отбрасываем.

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

P24 = 0 + 6 + 0 + 0 + 0 + 3 = 9;

N24 = 4 + 0 + 2 + 0 + 1 + 0 = 7;

D24 = P24 / N24 = 9/7 = 1.29 > 1 - принимаем.

P42 = 4 + 0 + 2 + 0 + 1 + 0 = 7;

N42 = 0 + 6 + 0 + 0 + 0 + 3 = 9;

D42 = P42 / N42 = 7/9 = 0.78 < 1 - отбрасываем.

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

P25 = 0 + 6 + 0 + 2 + 0 + 3 = 11;

N25 = 0 + 0 + 2 + 0 + 1 + 0 = 3;

D25 = P25 / N25 = 11/3 = 3.67 > 1 - принимаем.

P52 = 0 + 0 + 2 + 0 + 1 + 0 = 3;

N52 = 0 + 6 + 0 + 2 + 0 + 3 = 11;

D52 = P52 / N52 = 3/11 = 0.27 < 1 - отбрасываем.

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

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

N26 = 4 + 0 + 0 + 0 + 1 + 0 = 5;

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

P62 = 4 + 0 + 0 + 0 + 1 + 0 = 5;

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

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

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

P27 = 0 + 6 + 0 + 0 + 1 + 3 = 10;

N27 = 4 + 0 + 2 + 0 + 0 + 0 = 6;

D27 = P27 / N27 = 10/6 = 1.67 > 1 - принимаем.

P72 = 4 + 0 + 2 + 0 + 0 + 0 = 6;

N72 = 0 + 6 + 0 + 0 + 1 + 3 = 10;

D72 = P72 / N72 = 6/10 = 0.6 < 1 - отбрасываем.

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

P28 = 0 + 6 + 0 + 0 + 0 + 3 = 9;

N28 = 4 + 0 + 0 + 0 + 1 + 0 = 5;

D28 = P28 / N28 = 9/5 = 1.8 > 1 - принимаем.

P82 = 4 + 0 + 0 + 0 + 1 + 0 = 5;

N82 = 0 + 6 + 0 + 0 + 0 + 3 = 9;

D82 = P82 / N82 = 5/9 = 0.56 < 1 - отбрасываем.

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

P29 = 0 + 0 + 0 + 0 + 0 + 3 = 3;

N29 = 0 + 0 + 2 + 0 + 1 + 0 = 3;

D29 = P29 / N29 = 3/3 = 1 - отбрасываем.

P92 = 0 + 0 + 2 + 0 + 1 + 0 = 3;

N92 = 0 + 0 + 0 + 0 + 0 + 3 = 3;

D92 = P92 / N92 = 3/3 = 1 - отбрасываем.

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

P210 = 0 + 0 + 0 + 2 + 0 + 0 = 2;

N210 = 4 + 0 + 0 + 0 + 1 + 0 = 5;

D210 = P210 / N210 = 2/5 = 0.4 < 1 - отбрасываем.

P102 = 4 + 0 + 0 + 0 + 1 + 0 = 5;

N102 = 0 + 0 + 0 + 2 + 0 + 0 = 2;

D102 = P102 / N102 = 5/2 = 2.5 > 1 - принимаем.

Рассмотрим альтернативы 2 и 11 (i=2,j=11):

P211 = 0 + 6 + 0 + 0 + 0 + 3 = 9;

N211 = 4 + 0 + 0 + 0 + 1 + 0 = 5;

D211 = P211 / N211 = 9/5 = 1.8 > 1 - принимаем.

P112 = 4 + 0 + 0 + 0 + 1 + 0 = 5;

N112 = 0 + 6 + 0 + 0 + 0 + 3 = 9;

D112 = P112 / N112 = 5/9 = 0.56 < 1 - отбрасываем.

Рассмотрим альтернативы 2 и 12 (i=2,j=12):

P212 = 0 + 6 + 0 + 0 + 1 + 3 = 10;

N212 = 4 + 0 + 2 + 0 + 0 + 0 = 6;

D212 = P212 / N212 = 10/6 = 1.67 > 1 - принимаем.

P122 = 4 + 0 + 2 + 0 + 0 + 0 = 6;

N122 = 0 + 6 + 0 + 0 + 1 + 3 = 10;

D122 = P122 / N122 = 6/10 = 0.6 < 1 - отбрасываем.

Рассмотрим альтернативы 2 и 13 (i=2,j=13):

P213 = 0 + 6 + 0 + 0 + 0 + 0 = 6;

N213 = 4 + 0 + 0 + 0 + 1 + 0 = 5;

D213 = P213 / N213 = 6/5 = 1.2 > 1 - принимаем.

P132 = 4 + 0 + 0 + 0 + 1 + 0 = 5;

N132 = 0 + 6 + 0 + 0 + 0 + 0 = 6;

D132 = P132 / N132 = 5/6 = 0.83 < 1 - отбрасываем.

Рассмотрим альтернативы 2 и 14 (i=2,j=14):

P214 = 0 + 6 + 0 + 0 + 0 + 0 = 6;

N214 = 4 + 0 + 2 + 0 + 1 + 0 = 7;

D214 = P214 / N214 = 6/7 = 0.86 < 1 - отбрасываем.

P142 = 4 + 0 + 2 + 0 + 1 + 0 = 7;

N142 = 0 + 6 + 0 + 0 + 0 + 0 = 6;

D142 = P142 / N142 = 7/6 = 1.17 > 1 - принимаем.

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

P34 = 0 + 0 + 0 + 0 + 0 + 0 = 0;

N34 = 4 + 0 + 0 + 0 + 0 + 0 = 4;

D34 = P34 / N34 = 0/4 = 0.0 < 1 - отбрасываем.

P43 = 4 + 0 + 0 + 0 + 0 + 0 = 4;

N43 = 0 + 0 + 0 + 0 + 0 + 0 = 0;

D43 = P43 / N43 = 4/0 = ∞ > 1 - принимаем.

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

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

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

D35 = P35 / N35 = 6/0 = ∞ > 1 - принимаем.

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

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

D53 = P53 / N53 = 0/6 = 0.0 < 1 - отбрасываем.

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

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

N36 = 0 + 6 + 0 + 0 + 0 + 3 = 9;

D36 = P36 / N36 = 4/9 = 0.44 < 1 - отбрасываем.

P63 = 0 + 6 + 0 + 0 + 0 + 3 = 9;

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

D63 = P63 / N63 = 9/4 = 2.25 > 1 - принимаем.

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

P37 = 0 + 0 + 0 + 0 + 1 + 0 = 1;

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

D37 = P37 / N37 = 1/0 = ∞ > 1 - принимаем.

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

N73 = 0 + 0 + 0 + 0 + 1 + 0 = 1;

D73 = P73 / N73 = 0/1 = 0.0 < 1 - отбрасываем.

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

P38 = 0 + 6 + 2 + 0 + 0 + 0 = 8;

N38 = 4 + 0 + 0 + 0 + 0 + 0 = 4;

D38 = P38 / N38 = 8/4 = 2.0 > 1 - принимаем.

P83 = 4 + 0 + 0 + 0 + 0 + 0 = 4;

N83 = 0 + 6 + 2 + 0 + 0 + 0 = 8;

D83 = P83 / N83 = 4/8 = 0.5 < 1 - отбрасываем.

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

P39 = 4 + 0 + 0 + 0 + 0 + 0 = 4;

N39 = 0 + 6 + 0 + 0 + 0 + 0 = 6;

D39 = P39 / N39 = 4/6 = 0.67 < 1 - отбрасываем.

P93 = 0 + 6 + 0 + 0 + 0 + 0 = 6;

N93 = 4 + 0 + 0 + 0 + 0 + 0 = 4;

D93 = P93 / N93 = 6/4 = 1.5 > 1 - принимаем.

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

P310 = 0 + 0 + 2 + 2 + 0 + 0 = 4;

N310 = 0 + 6 + 0 + 0 + 0 + 3 = 9;

D310 = P310 / N310 = 4/9 = 0.44 < 1 - отбрасываем.

P103 = 0 + 6 + 0 + 0 + 0 + 3 = 9;

N103 = 0 + 0 + 2 + 2 + 0 + 0 = 4;

D103 = P103 / N103 = 9/4 = 2.25 > 1 - принимаем.

Рассмотрим альтернативы 3 и 11 (i=3,j=11):

P311 = 0 + 0 + 2 + 0 + 0 + 0 = 2;

N311 = 0 + 0 + 0 + 0 + 0 + 0 = 0;

D311 = P311 / N311 = 2/0 = ∞ > 1 - принимаем.

P113 = 0 + 0 + 0 + 0 + 0 + 0 = 0;

N113 = 0 + 0 + 2 + 0 + 0 + 0 = 2;

D113 = P113 / N113 = 0/2 = 0.0 < 1 - отбрасываем.

Рассмотрим альтернативы 3 и 12 (i=3,j=12):

P312 = 0 + 0 + 0 + 0 + 1 + 0 = 1;

N312 = 4 + 0 + 0 + 0 + 0 + 0 = 4;

D312 = P312 / N312 = 1/4 = 0.25 < 1 - отбрасываем.

P123 = 4 + 0 + 0 + 0 + 0 + 0 = 4;

N123 = 0 + 0 + 0 + 0 + 1 + 0 = 1;

D123 = P123 / N123 = 4/1 = 4.0 > 1 - принимаем.

Рассмотрим альтернативы 3 и 13 (i=3,j=13):

P313 = 0 + 6 + 2 + 0 + 0 + 0 = 8;

N313 = 4 + 0 + 0 + 0 + 0 + 3 = 7;

D313 = P313 / N313 = 8/7 = 1.14 > 1 - принимаем.

P133 = 4 + 0 + 0 + 0 + 0 + 3 = 7;

N133 = 0 + 6 + 2 + 0 + 0 + 0 = 8;

D133 = P133 / N133 = 7/8 = 0.88 < 1 - отбрасываем.

Рассмотрим альтернативы 3 и 14 (i=3,j=14):

P314 = 0 + 6 + 0 + 0 + 0 + 0 = 6;

N314 = 4 + 0 + 0 + 0 + 0 + 3 = 7;

D314 = P314 / N314 = 6/7 = 0.86 < 1 - отбрасываем.

P143 = 4 + 0 + 0 + 0 + 0 + 3 = 7;

N143 = 0 + 6 + 0 + 0 + 0 + 0 = 6;

D143 = P143 / N143 = 7/6 = 1.17 > 1 - принимаем.

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

P45 = 4 + 0 + 0 + 2 + 0 + 0 = 6;

N45 = 0 + 0 + 0 + 0 + 0 + 0 = 0;

D45 = P45 / N45 = 6/0 = ∞ > 1 - принимаем.

P54 = 0 + 0 + 0 + 0 + 0 + 0 = 0;

N54 = 4 + 0 + 0 + 2 + 0 + 0 = 6;

D54 = P54 / N54 = 0/6 = 0.0 < 1 - отбрасываем.

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

P46 = 4 + 0 + 2 + 2 + 0 + 0 = 8;

N46 = 0 + 6 + 0 + 0 + 0 + 3 = 9;

D46 = P46 / N46 = 8/9 = 0.89 < 1 - отбрасываем.

P64 = 0 + 6 + 0 + 0 + 0 + 3 = 9;

N64 = 4 + 0 + 2 + 2 + 0 + 0 = 8;

D64 = P64 / N64 = 9/8 = 1.12 > 1 - принимаем.

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

P47 = 4 + 0 + 0 + 0 + 1 + 0 = 5;

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

D47 = P47 / N47 = 5/0 = ∞ > 1 - принимаем.

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

N74 = 4 + 0 + 0 + 0 + 1 + 0 = 5;

D74 = P74 / N74 = 0/5 = 0.0 < 1 - отбрасываем.

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

P48 = 0 + 6 + 2 + 0 + 0 + 0 = 8;

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

D48 = P48 / N48 = 8/0 = ∞ > 1 - принимаем.

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

N84 = 0 + 6 + 2 + 0 + 0 + 0 = 8;

D84 = P84 / N84 = 0/8 = 0.0 < 1 - отбрасываем.

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

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

N49 = 0 + 6 + 0 + 0 + 0 + 0 = 6;

D49 = P49 / N49 = 4/6 = 0.67 < 1 - отбрасываем.

P94 = 0 + 6 + 0 + 0 + 0 + 0 = 6;

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

D94 = P94 / N94 = 6/4 = 1.5 > 1 - принимаем.

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

P410 = 4 + 0 + 2 + 2 + 0 + 0 = 8;

N410 = 0 + 6 + 0 + 0 + 0 + 3 = 9;

D410 = P410 / N410 = 8/9 = 0.89 < 1 - отбрасываем.

P104 = 0 + 6 + 0 + 0 + 0 + 3 = 9;

N104 = 4 + 0 + 2 + 2 + 0 + 0 = 8;

D104 = P104 / N104 = 9/8 = 1.12 > 1 - принимаем.

Рассмотрим альтернативы 4 и 11 (i=4,j=11):

P411 = 4 + 0 + 2 + 0 + 0 + 0 = 6;

N411 = 0 + 0 + 0 + 0 + 0 + 0 = 0;

D411 = P411 / N411 = 6/0 = ∞ > 1 - принимаем.

P114 = 0 + 0 + 0 + 0 + 0 + 0 = 0;

N114 = 4 + 0 + 2 + 0 + 0 + 0 = 6;

D114 = P114 / N114 = 0/6 = 0.0 < 1 - отбрасываем.

Рассмотрим альтернативы 4 и 12 (i=4,j=12):

P412 = 0 + 0 + 0 + 0 + 1 + 0 = 1;

N412 = 0 + 0 + 0 + 0 + 0 + 0 = 0;

D412 = P412 / N412 = 1/0 = ∞ > 1 - принимаем.

P124 = 0 + 0 + 0 + 0 + 0 + 0 = 0;

N124 = 0 + 0 + 0 + 0 + 1 + 0 = 1;

D124 = P124 / N124 = 0/1 = 0.0 < 1 - отбрасываем.

Рассмотрим альтернативы 4 и 13 (i=4,j=13):

P413 = 0 + 6 + 2 + 0 + 0 + 0 = 8;

N413 = 0 + 0 + 0 + 0 + 0 + 3 = 3;

D413 = P413 / N413 = 8/3 = 2.67 > 1 - принимаем.

P134 = 0 + 0 + 0 + 0 + 0 + 3 = 3;

N134 = 0 + 6 + 2 + 0 + 0 + 0 = 8;

D134 = P134 / N134 = 3/8 = 0.38 < 1 - отбрасываем.

Рассмотрим альтернативы 4 и 14 (i=4,j=14):

P414 = 0 + 6 + 0 + 0 + 0 + 0 = 6;

N414 = 0 + 0 + 0 + 0 + 0 + 3 = 3;

D414 = P414 / N414 = 6/3 = 2.0 > 1 - принимаем.

P144 = 0 + 0 + 0 + 0 + 0 + 3 = 3;

N144 = 0 + 6 + 0 + 0 + 0 + 0 = 6;

D144 = P144 / N144 = 3/6 = 0.5 < 1 - отбрасываем.

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

P56 = 0 + 0 + 2 + 0 + 0 + 0 = 2;

N56 = 4 + 6 + 0 + 0 + 0 + 3 = 13;

D56 = P56 / N56 = 2/13 = 0.15 < 1 - отбрасываем.

P65 = 4 + 6 + 0 + 0 + 0 + 3 = 13;

N65 = 0 + 0 + 2 + 0 + 0 + 0 = 2;

D65 = P65 / N65 = 13/2 = 6.5 > 1 - принимаем.

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

P57 = 0 + 0 + 0 + 0 + 1 + 0 = 1;

N57 = 4 + 0 + 0 + 2 + 0 + 0 = 6;

D57 = P57 / N57 = 1/6 = 0.17 < 1 - отбрасываем.

P75 = 4 + 0 + 0 + 2 + 0 + 0 = 6;

N75 = 0 + 0 + 0 + 0 + 1 + 0 = 1;

D75 = P75 / N75 = 6/1 = 6.0 > 1 - принимаем.

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

P58 = 0 + 6 + 2 + 0 + 0 + 0 = 8;

N58 = 4 + 0 + 0 + 2 + 0 + 0 = 6;

D58 = P58 / N58 = 8/6 = 1.33 > 1 - принимаем.

P85 = 4 + 0 + 0 + 2 + 0 + 0 = 6;

N85 = 0 + 6 + 2 + 0 + 0 + 0 = 8;

D85 = P85 / N85 = 6/8 = 0.75 < 1 - отбрасываем.

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

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

N59 = 0 + 6 + 0 + 2 + 0 + 0 = 8;

D59 = P59 / N59 = 0/8 = 0.0 < 1 - отбрасываем.

P95 = 0 + 6 + 0 + 2 + 0 + 0 = 8;

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

D95 = P95 / N95 = 8/0 = ∞ > 1 - принимаем.

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

P510 = 0 + 0 + 2 + 0 + 0 + 0 = 2;

N510 = 4 + 6 + 0 + 0 + 0 + 3 = 13;

D510 = P510 / N510 = 2/13 = 0.15 < 1 - отбрасываем.

P105 = 4 + 6 + 0 + 0 + 0 + 3 = 13;

N105 = 0 + 0 + 2 + 0 + 0 + 0 = 2;

D105 = P105 / N105 = 13/2 = 6.5 > 1 - принимаем.

Рассмотрим альтернативы 5 и 11 (i=5,j=11):

P511 = 0 + 0 + 2 + 0 + 0 + 0 = 2;

N511 = 4 + 0 + 0 + 2 + 0 + 0 = 6;

D511 = P511 / N511 = 2/6 = 0.33 < 1 - отбрасываем.

P115 = 4 + 0 + 0 + 2 + 0 + 0 = 6;

N115 = 0 + 0 + 2 + 0 + 0 + 0 = 2;

D115 = P115 / N115 = 6/2 = 3.0 > 1 - принимаем.

Рассмотрим альтернативы 5 и 12 (i=5,j=12):

P512 = 0 + 0 + 0 + 0 + 1 + 0 = 1;

N512 = 4 + 0 + 0 + 2 + 0 + 0 = 6;

D512 = P512 / N512 = 1/6 = 0.17 < 1 - отбрасываем.

P125 = 4 + 0 + 0 + 2 + 0 + 0 = 6;

N125 = 0 + 0 + 0 + 0 + 1 + 0 = 1;

D125 = P125 / N125 = 6/1 = 6.0 > 1 - принимаем.

Рассмотрим альтернативы 5 и 13 (i=5,j=13):

P513 = 0 + 6 + 2 + 0 + 0 + 0 = 8;

N513 = 4 + 0 + 0 + 2 + 0 + 3 = 9;

D513 = P513 / N513 = 8/9 = 0.89 < 1 - отбрасываем.

P135 = 4 + 0 + 0 + 2 + 0 + 3 = 9;

N135 = 0 + 6 + 2 + 0 + 0 + 0 = 8;

D135 = P135 / N135 = 9/8 = 1.12 > 1 - принимаем.

Рассмотрим альтернативы 5 и 14 (i=5,j=14):

P514 = 0 + 6 + 0 + 0 + 0 + 0 = 6;

N514 = 4 + 0 + 0 + 2 + 0 + 3 = 9;

D514 = P514 / N514 = 6/9 = 0.67 < 1 - отбрасываем.

P145 = 4 + 0 + 0 + 2 + 0 + 3 = 9;

N145 = 0 + 6 + 0 + 0 + 0 + 0 = 6;

D145 = P145 / N145 = 9/6 = 1.5 > 1 - принимаем.

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

P67 = 0 + 6 + 0 + 0 + 1 + 3 = 10;

N67 = 0 + 0 + 2 + 2 + 0 + 0 = 4;

D67 = P67 / N67 = 10/4 = 2.5 > 1 - принимаем.

P76 = 0 + 0 + 2 + 2 + 0 + 0 = 4;

N76 = 0 + 6 + 0 + 0 + 1 + 3 = 10;

D76 = P76 / N76 = 4/10 = 0.4 < 1 - отбрасываем.

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

P68 = 0 + 6 + 0 + 0 + 0 + 3 = 9;

N68 = 4 + 0 + 0 + 2 + 0 + 0 = 6;

D68 = P68 / N68 = 9/6 = 1.5 > 1 - принимаем.

P86 = 4 + 0 + 0 + 2 + 0 + 0 = 6;

N86 = 0 + 6 + 0 + 0 + 0 + 3 = 9;

D86 = P86 / N86 = 6/9 = 0.67 < 1 - отбрасываем.

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

P69 = 4 + 0 + 0 + 0 + 0 + 3 = 7;

N69 = 0 + 0 + 2 + 2 + 0 + 0 = 4;

D69 = P69 / N69 = 7/4 = 1.75 > 1 - принимаем.

P96 = 0 + 0 + 2 + 2 + 0 + 0 = 4;

N96 = 4 + 0 + 0 + 0 + 0 + 3 = 7;

D96 = P96 / N96 = 4/7 = 0.57 < 1 - отбрасываем.

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

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

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

D610 = P610 / N610 = 0/0 = 0 < 1 - отбрасываем.

P106 = 0 + 0 + 0 + 0 + 0 + 0 = 0;

N106 = 0 + 0 + 0 + 0 + 0 + 0 = 0;

D106 = P106 / N106 = 0/0 = 0 < 1 - отбрасываем.

Рассмотрим альтернативы 6 и 11 (i=6,j=11):

P611 = 0 + 6 + 0 + 0 + 0 + 3 = 9;

N611 = 0 + 0 + 0 + 2 + 0 + 0 = 2;

D611 = P611 / N611 = 9/2 = 4.5 > 1 - принимаем.

P116 = 0 + 0 + 0 + 2 + 0 + 0 = 2;

N116 = 0 + 6 + 0 + 0 + 0 + 3 = 9;

D116 = P116 / N116 = 2/9 = 0.22 < 1 - отбрасываем.

Рассмотрим альтернативы 6 и 12 (i=6,j=12):

P612 = 0 + 6 + 0 + 0 + 1 + 3 = 10;

N612 = 4 + 0 + 2 + 2 + 0 + 0 = 8;

D612 = P612 / N612 = 10/8 = 1.25 > 1 - принимаем.

P126 = 4 + 0 + 2 + 2 + 0 + 0 = 8;

N126 = 0 + 6 + 0 + 0 + 1 + 3 = 10;

D126 = P126 / N126 = 8/10 = 0.8 < 1 - отбрасываем.

Рассмотрим альтернативы 6 и 13 (i=6,j=13):

P613 = 0 + 6 + 0 + 0 + 0 + 0 = 6;

N613 = 4 + 0 + 0 + 2 + 0 + 0 = 6;

D613 = P613 / N613 = 6/6 = 1 - отбрасываем.

P136 = 4 + 0 + 0 + 2 + 0 + 0 = 6;

N136 = 0 + 6 + 0 + 0 + 0 + 0 = 6;

D136 = P136 / N136 = 6/6 = 1 - отбрасываем.

Рассмотрим альтернативы 6 и 14 (i=6,j=14):

P614 = 0 + 6 + 0 + 0 + 0 + 0 = 6;

N614 = 4 + 0 + 2 + 2 + 0 + 0 = 8;

D614 = P614 / N614 = 6/8 = 0.75 < 1 - отбрасываем.

P146 = 4 + 0 + 2 + 2 + 0 + 0 = 8;

N146 = 0 + 6 + 0 + 0 + 0 + 0 = 6;

D146 = P146 / N146 = 8/6 = 1.33 > 1 - принимаем.

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

P78 = 0 + 6 + 2 + 0 + 0 + 0 = 8;

N78 = 4 + 0 + 0 + 0 + 1 + 0 = 5;

D78 = P78 / N78 = 8/5 = 1.6 > 1 - принимаем.

P87 = 4 + 0 + 0 + 0 + 1 + 0 = 5;

N87 = 0 + 6 + 2 + 0 + 0 + 0 = 8;

D87 = P87 / N87 = 5/8 = 0.62 < 1 - отбрасываем.

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

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

N79 = 0 + 6 + 0 + 0 + 1 + 0 = 7;

D79 = P79 / N79 = 4/7 = 0.57 < 1 - отбрасываем.

P97 = 0 + 6 + 0 + 0 + 1 + 0 = 7;

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

D97 = P97 / N97 = 7/4 = 1.75 > 1 - принимаем.

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

P710 = 0 + 0 + 2 + 2 + 0 + 0 = 4;

N710 = 0 + 6 + 0 + 0 + 1 + 3 = 10;

D710 = P710 / N710 = 4/10 = 0.4 < 1 - отбрасываем.

P107 = 0 + 6 + 0 + 0 + 1 + 3 = 10;

N107 = 0 + 0 + 2 + 2 + 0 + 0 = 4;

D107 = P107 / N107 = 10/4 = 2.5 > 1 - принимаем.

Рассмотрим альтернативы 7 и 11 (i=7,j=11):

P711 = 0 + 0 + 2 + 0 + 0 + 0 = 2;

N711 = 0 + 0 + 0 + 0 + 1 + 0 = 1;

D711 = P711 / N711 = 2/1 = 2.0 > 1 - принимаем.

P117 = 0 + 0 + 0 + 0 + 1 + 0 = 1;

N117 = 0 + 0 + 2 + 0 + 0 + 0 = 2;

D117 = P117 / N117 = 1/2 = 0.5 < 1 - отбрасываем.

Рассмотрим альтернативы 7 и 12 (i=7,j=12):

P712 = 0 + 0 + 0 + 0 + 0 + 0 = 0;

N712 = 4 + 0 + 0 + 0 + 0 + 0 = 4;

D712 = P712 / N712 = 0/4 = 0.0 < 1 - отбрасываем.

P127 = 4 + 0 + 0 + 0 + 0 + 0 = 4;

N127 = 0 + 0 + 0 + 0 + 0 + 0 = 0;

D127 = P127 / N127 = 4/0 = ∞ > 1 - принимаем.

Рассмотрим альтернативы 7 и 13 (i=7,j=13):

P713 = 0 + 6 + 2 + 0 + 0 + 0 = 8;

N713 = 4 + 0 + 0 + 0 + 1 + 3 = 8;

D713 = P713 / N713 = 8/8 = 1 - отбрасываем.

P137 = 4 + 0 + 0 + 0 + 1 + 3 = 8;

N137 = 0 + 6 + 2 + 0 + 0 + 0 = 8;

D137 = P137 / N137 = 8/8 = 1 - отбрасываем.

Рассмотрим альтернативы 7 и 14 (i=7,j=14):

P714 = 0 + 6 + 0 + 0 + 0 + 0 = 6;

N714 = 4 + 0 + 0 + 0 + 1 + 3 = 8;

D714 = P714 / N714 = 6/8 = 0.75 < 1 - отбрасываем.

P147 = 4 + 0 + 0 + 0 + 1 + 3 = 8;

N147 = 0 + 6 + 0 + 0 + 0 + 0 = 6;

D147 = P147 / N147 = 8/6 = 1.33 > 1 - принимаем.

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

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

N89 = 0 + 6 + 2 + 0 + 0 + 0 = 8;

D89 = P89 / N89 = 4/8 = 0.5 < 1 - отбрасываем.

P98 = 0 + 6 + 2 + 0 + 0 + 0 = 8;

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

D98 = P98 / N98 = 8/4 = 2.0 > 1 - принимаем.

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

P810 = 4 + 0 + 0 + 2 + 0 + 0 = 6;

N810 = 0 + 6 + 0 + 0 + 0 + 3 = 9;

D810 = P810 / N810 = 6/9 = 0.67 < 1 - отбрасываем.

P108 = 0 + 6 + 0 + 0 + 0 + 3 = 9;

N108 = 4 + 0 + 0 + 2 + 0 + 0 = 6;

D108 = P108 / N108 = 9/6 = 1.5 > 1 - принимаем.

Рассмотрим альтернативы 8 и 11 (i=8,j=11):

P811 = 4 + 0 + 0 + 0 + 0 + 0 = 4;

N811 = 0 + 6 + 0 + 0 + 0 + 0 = 6;

D811 = P811 / N811 = 4/6 = 0.67 < 1 - отбрасываем.

P118 = 0 + 6 + 0 + 0 + 0 + 0 = 6;

N118 = 4 + 0 + 0 + 0 + 0 + 0 = 4;

D118 = P118 / N118 = 6/4 = 1.5 > 1 - принимаем.

Рассмотрим альтернативы 8 и 12 (i=8,j=12):

P812 = 0 + 0 + 0 + 0 + 1 + 0 = 1;

N812 = 0 + 6 + 2 + 0 + 0 + 0 = 8;

D812 = P812 / N812 = 1/8 = 0.12 < 1 - отбрасываем.

P128 = 0 + 6 + 2 + 0 + 0 + 0 = 8;

N128 = 0 + 0 + 0 + 0 + 1 + 0 = 1;

D128 = P128 / N128 = 8/1 = 8.0 > 1 - принимаем.

Рассмотрим альтернативы 8 и 13 (i=8,j=13):

P813 = 0 + 0 + 0 + 0 + 0 + 0 = 0;

N813 = 0 + 0 + 0 + 0 + 0 + 3 = 3;

D813 = P813 / N813 = 0/3 = 0.0 < 1 - отбрасываем.

P138 = 0 + 0 + 0 + 0 + 0 + 3 = 3;

N138 = 0 + 0 + 0 + 0 + 0 + 0 = 0;

D138 = P138 / N138 = 3/0 = ∞ > 1 - принимаем.

Рассмотрим альтернативы 8 и 14 (i=8,j=14):

P814 = 0 + 0 + 0 + 0 + 0 + 0 = 0;

N814 = 0 + 0 + 2 + 0 + 0 + 3 = 5;

D814 = P814 / N814 = 0/5 = 0.0 < 1 - отбрасываем.

P148 = 0 + 0 + 2 + 0 + 0 + 3 = 5;

N148 = 0 + 0 + 0 + 0 + 0 + 0 = 0;

D148 = P148 / N148 = 5/0 = ∞ > 1 - принимаем.

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

P910 = 0 + 0 + 2 + 2 + 0 + 0 = 4;

N910 = 4 + 0 + 0 + 0 + 0 + 3 = 7;

D910 = P910 / N910 = 4/7 = 0.57 < 1 - отбрасываем.

P109 = 4 + 0 + 0 + 0 + 0 + 3 = 7;

N109 = 0 + 0 + 2 + 2 + 0 + 0 = 4;

D109 = P109 / N109 = 7/4 = 1.75 > 1 - принимаем.

Рассмотрим альтернативы 9 и 11 (i=9,j=11):

P911 = 0 + 6 + 2 + 0 + 0 + 0 = 8;

N911 = 4 + 0 + 0 + 0 + 0 + 0 = 4;

D911 = P911 / N911 = 8/4 = 2.0 > 1 - принимаем.

P119 = 4 + 0 + 0 + 0 + 0 + 0 = 4;

N119 = 0 + 6 + 2 + 0 + 0 + 0 = 8;

D119 = P119 / N119 = 4/8 = 0.5 < 1 - отбрасываем.

Рассмотрим альтернативы 9 и 12 (i=9,j=12):

P912 = 0 + 6 + 0 + 0 + 1 + 0 = 7;

N912 = 4 + 0 + 0 + 0 + 0 + 0 = 4;

D912 = P912 / N912 = 7/4 = 1.75 > 1 - принимаем.

P129 = 4 + 0 + 0 + 0 + 0 + 0 = 4;

N129 = 0 + 6 + 0 + 0 + 1 + 0 = 7;

D129 = P129 / N129 = 4/7 = 0.57 < 1 - отбрасываем.

Рассмотрим альтернативы 9 и 13 (i=9,j=13):

P913 = 0 + 6 + 2 + 0 + 0 + 0 = 8;

N913 = 4 + 0 + 0 + 0 + 0 + 3 = 7;

D913 = P913 / N913 = 8/7 = 1.14 > 1 - принимаем.

P139 = 4 + 0 + 0 + 0 + 0 + 3 = 7;

N139 = 0 + 6 + 2 + 0 + 0 + 0 = 8;

D139 = P139 / N139 = 7/8 = 0.88 < 1 - отбрасываем.

Рассмотрим альтернативы 9 и 14 (i=9,j=14):

P914 = 0 + 6 + 0 + 0 + 0 + 0 = 6;

N914 = 4 + 0 + 0 + 0 + 0 + 3 = 7;

D914 = P914 / N914 = 6/7 = 0.86 < 1 - отбрасываем.

P149 = 4 + 0 + 0 + 0 + 0 + 3 = 7;

N149 = 0 + 6 + 0 + 0 + 0 + 0 = 6;

D149 = P149 / N149 = 7/6 = 1.17 > 1 - принимаем.

Рассмотрим альтернативы 10 и 11 (i=10,j=11):

P1011 = 0 + 6 + 0 + 0 + 0 + 3 = 9;

N1011 = 0 + 0 + 0 + 2 + 0 + 0 = 2;

D1011 = P1011 / N1011 = 9/2 = 4.5 > 1 - принимаем.

P1110 = 0 + 0 + 0 + 2 + 0 + 0 = 2;

N1110 = 0 + 6 + 0 + 0 + 0 + 3 = 9;

D1110 = P1110 / N1110 = 2/9 = 0.22 < 1 - отбрасываем.

Рассмотрим альтернативы 10 и 12 (i=10,j=12):

P1012 = 0 + 6 + 0 + 0 + 1 + 3 = 10;

N1012 = 4 + 0 + 2 + 2 + 0 + 0 = 8;

D1012 = P1012 / N1012 = 10/8 = 1.25 > 1 - принимаем.

P1210 = 4 + 0 + 2 + 2 + 0 + 0 = 8;

N1210 = 0 + 6 + 0 + 0 + 1 + 3 = 10;

D1210 = P1210 / N1210 = 8/10 = 0.8 < 1 - отбрасываем.

Рассмотрим альтернативы 10 и 13 (i=10,j=13):

P1013 = 0 + 6 + 0 + 0 + 0 + 0 = 6;

N1013 = 4 + 0 + 0 + 2 + 0 + 0 = 6;

D1013 = P1013 / N1013 = 6/6 = 1 - отбрасываем.

P1310 = 4 + 0 + 0 + 2 + 0 + 0 = 6;

N1310 = 0 + 6 + 0 + 0 + 0 + 0 = 6;

D1310 = P1310 / N1310 = 6/6 = 1 - отбрасываем.

Рассмотрим альтернативы 10 и 14 (i=10,j=14):

P1014 = 0 + 6 + 0 + 0 + 0 + 0 = 6;

N1014 = 4 + 0 + 2 + 2 + 0 + 0 = 8;

D1014 = P1014 / N1014 = 6/8 = 0.75 < 1 - отбрасываем.

P1410 = 4 + 0 + 2 + 2 + 0 + 0 = 8;

N1410 = 0 + 6 + 0 + 0 + 0 + 0 = 6;

D1410 = P1410 / N1410 = 8/6 = 1.33 > 1 - принимаем.

Рассмотрим альтернативы 11 и 12 (i=11,j=12):

P1112 = 0 + 0 + 0 + 0 + 1 + 0 = 1;

N1112 = 4 + 0 + 2 + 0 + 0 + 0 = 6;

D1112 = P1112 / N1112 = 1/6 = 0.17 < 1 - отбрасываем.

P1211 = 4 + 0 + 2 + 0 + 0 + 0 = 6;

N1211 = 0 + 0 + 0 + 0 + 1 + 0 = 1;

D1211 = P1211 / N1211 = 6/1 = 6.0 > 1 - принимаем.

Рассмотрим альтернативы 11 и 13 (i=11,j=13):

P1113 = 0 + 6 + 0 + 0 + 0 + 0 = 6;

N1113 = 4 + 0 + 0 + 0 + 0 + 3 = 7;

D1113 = P1113 / N1113 = 6/7 = 0.86 < 1 - отбрасываем.

P1311 = 4 + 0 + 0 + 0 + 0 + 3 = 7;

N1311 = 0 + 6 + 0 + 0 + 0 + 0 = 6;

D1311 = P1311 / N1311 = 7/6 = 1.17 > 1 - принимаем.

Рассмотрим альтернативы 11 и 14 (i=11,j=14):

P1114 = 0 + 6 + 0 + 0 + 0 + 0 = 6;

N1114 = 4 + 0 + 2 + 0 + 0 + 3 = 9;

D1114 = P1114 / N1114 = 6/9 = 0.67 < 1 - отбрасываем.

P1411 = 4 + 0 + 2 + 0 + 0 + 3 = 9;

N1411 = 0 + 6 + 0 + 0 + 0 + 0 = 6;

D1411 = P1411 / N1411 = 9/6 = 1.5 > 1 - принимаем.

Рассмотрим альтернативы 12 и 13 (i=12,j=13):

P1213 = 0 + 6 + 2 + 0 + 0 + 0 = 8;

N1213 = 0 + 0 + 0 + 0 + 1 + 3 = 4;

D1213 = P1213 / N1213 = 8/4 = 2.0 > 1 - принимаем.

P1312 = 0 + 0 + 0 + 0 + 1 + 3 = 4;

N1312 = 0 + 6 + 2 + 0 + 0 + 0 = 8;

D1312 = P1312 / N1312 = 4/8 = 0.5 < 1 - отбрасываем.

Рассмотрим альтернативы 12 и 14 (i=12,j=14):

P1214 = 0 + 6 + 0 + 0 + 0 + 0 = 6;

N1214 = 0 + 0 + 0 + 0 + 1 + 3 = 4;

D1214 = P1214 / N1214 = 6/4 = 1.5 > 1 - принимаем.

P1412 = 0 + 0 + 0 + 0 + 1 + 3 = 4;

N1412 = 0 + 6 + 0 + 0 + 0 + 0 = 6;

D1412 = P1412 / N1412 = 4/6 = 0.67 < 1 - отбрасываем.

Рассмотрим альтернативы 13 и 14 (i=13,j=14):

P1314 = 0 + 0 + 0 + 0 + 0 + 0 = 0;

N1314 = 0 + 0 + 2 + 0 + 0 + 0 = 2;

D1314 = P1314 / N1314 = 0/2 = 0.0 < 1 - отбрасываем.

P1413 = 0 + 0 + 2 + 0 + 0 + 0 = 2;

N1413 = 0 + 0 + 0 + 0 + 0 + 0 = 0;

D1413 = P1413 / N1413 = 2/0 = ∞ > 1 - принимаем.

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 | 11 | 12 | 13 | 14 |
| 1 | - | - | - | - | 2.0 | - | - | 1.6 | - | - | - | - | - | - |
| 2 | 5.0 | - | 1.29 | 1.29 | 3.67 | - | 1.67 | 1.8 | - | - | 1.8 | 1.67 | 1.2 | - |
| 3 | ∞ | - | - | - | ∞ | - | ∞ | 2.0 | - | - | ∞ | - | 1.14 | - |
| 4 | ∞ | - | ∞ | - | ∞ | - | ∞ | ∞ | - | - | ∞ | ∞ | 2.67 | 2.0 |
| 5 | - | - | - | - | - | - | - | 1.33 | - | - | - | - | - | - |
| 6 | 3.5 | 2.5 | 2.25 | 1.12 | 6.5 | - | 2.5 | 1.5 | 1.75 | - | 4.5 | 1.25 | - | - |
| 7 | ∞ | - | - | - | 6.0 | - | - | 1.6 | - | - | 2.0 | - | - | - |
| 8 | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 9 | ∞ | - | 1.5 | 1.5 | ∞ | - | 1.75 | 2.0 | - | - | 2.0 | 1.75 | 1.14 | - |
| 10 | 3.5 | 2.5 | 2.25 | 1.12 | 6.5 | - | 2.5 | 1.5 | 1.75 | - | 4.5 | 1.25 | - | - |
| 11 | 2.5 | - | - | - | 3.0 | - | - | 1.5 | - | - | - | - | - | - |
| 12 | ∞ | - | 4.0 | - | 6.0 | - | ∞ | 8.0 | - | - | 6.0 | - | 2.0 | 1.5 |
| 13 | - | - | - | - | 1.12 | - | - | ∞ | - | - | 1.17 | - | - | - |
| 14 | 1.33 | 1.17 | 1.17 | - | 1.5 | 1.33 | 1.33 | ∞ | 1.17 | 1.33 | 1.5 | - | ∞ | - |

По матрице построен граф предпочтений (Рисунок 2.3.1).

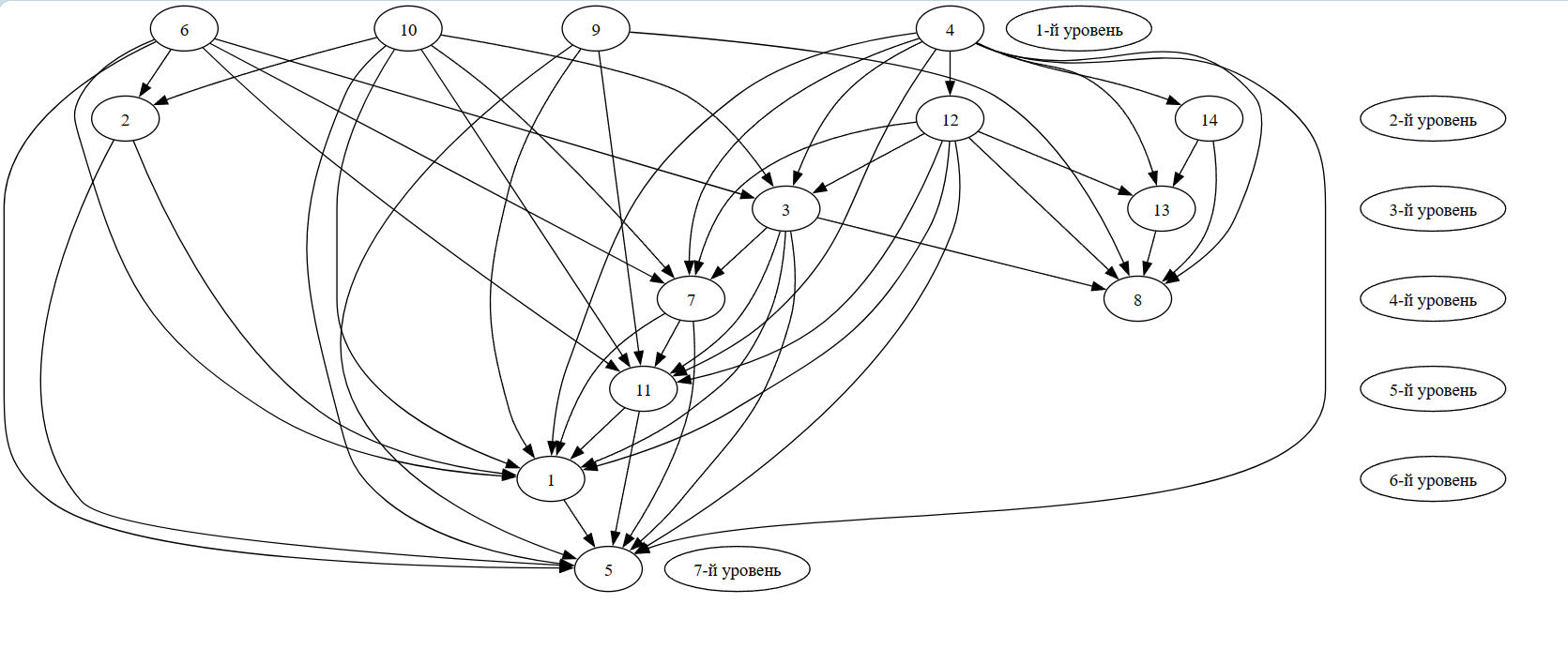


Рисунок 2.3.1 – Вид графа предпочтений

Назначен порог отбора предпочтений C = 2 (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=2

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 |
| 1 | - | - | - | - | 2.0 | - | - | - | - | - | - | - | - | - |
| 2 | 5.0 | - | - | - | 3.67 | - | - | - | - | - | - | - | - | - |
| 3 | ∞ | - | - | - | ∞ | - | ∞ | 2.0 | - | - | ∞ | - | - | - |
| 4 | ∞ | - | ∞ | - | ∞ | - | ∞ | ∞ | - | - | ∞ | ∞ | 2.67 | 2.0 |
| 5 | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 6 | 3.5 | 2.5 | 2.25 | - | 6.5 | - | 2.5 | - | - | - | 4.5 | - | - | - |
| 7 | ∞ | - | - | - | 6.0 | - | - | - | - | - | 2.0 | - | - | - |
| 8 | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 9 | ∞ | - | - | - | ∞ | - | - | 2.0 | - | - | 2.0 | - | - | - |
| 10 | 3.5 | 2.5 | 2.25 | - | 6.5 | - | 2.5 | - | - | - | 4.5 | - | - | - |
| 11 | 2.5 | - | - | - | 3.0 | - | - | - | - | - | - | - | - | - |
| 12 | ∞ | - | 4.0 | - | 6.0 | - | ∞ | 8.0 | - | - | 6.0 | - | 2.0 | - |
| 13 | - | - | - | - | - | - | - | ∞ | - | - | - | - | - | - |
| 14 | - | - | - | - | - | - | - | ∞ | - | - | - | - | ∞ | - |

По этой матрице построен граф предпочтений (Рисунок 2.3.2).

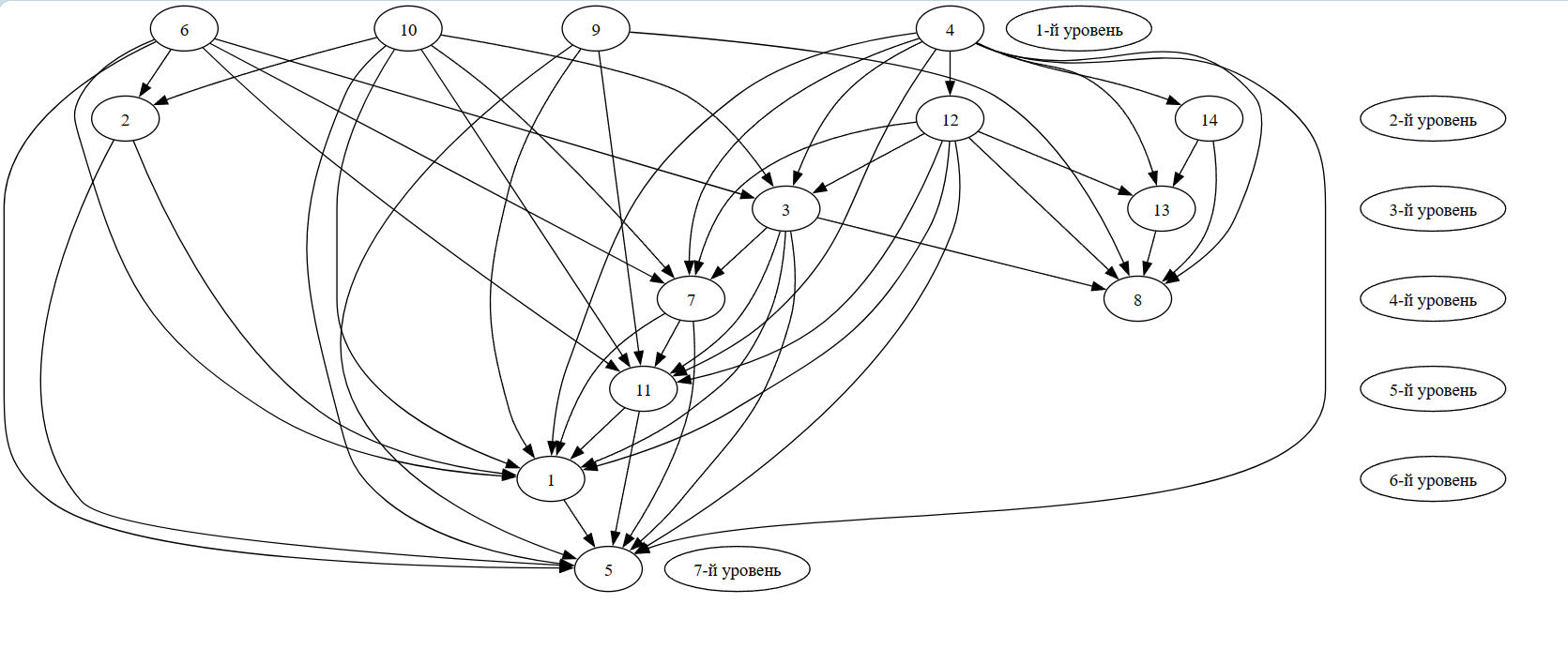


Рисунок 2.3.2 – Вид графа предпочтений для случая порога принятия решений C = 2

Циклов в графе нет, при этом граф остался целостным. Оптимальными решениями являются альтернативы А4, А6, А9, А10.

## 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 появляются 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 – 2.5.3.

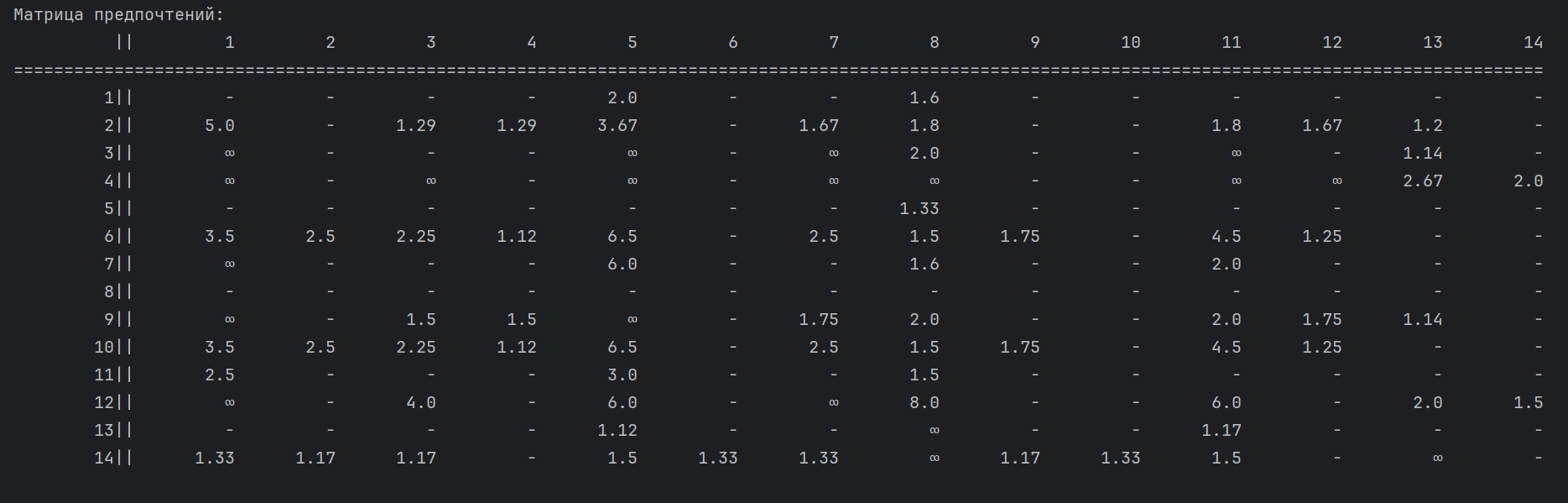
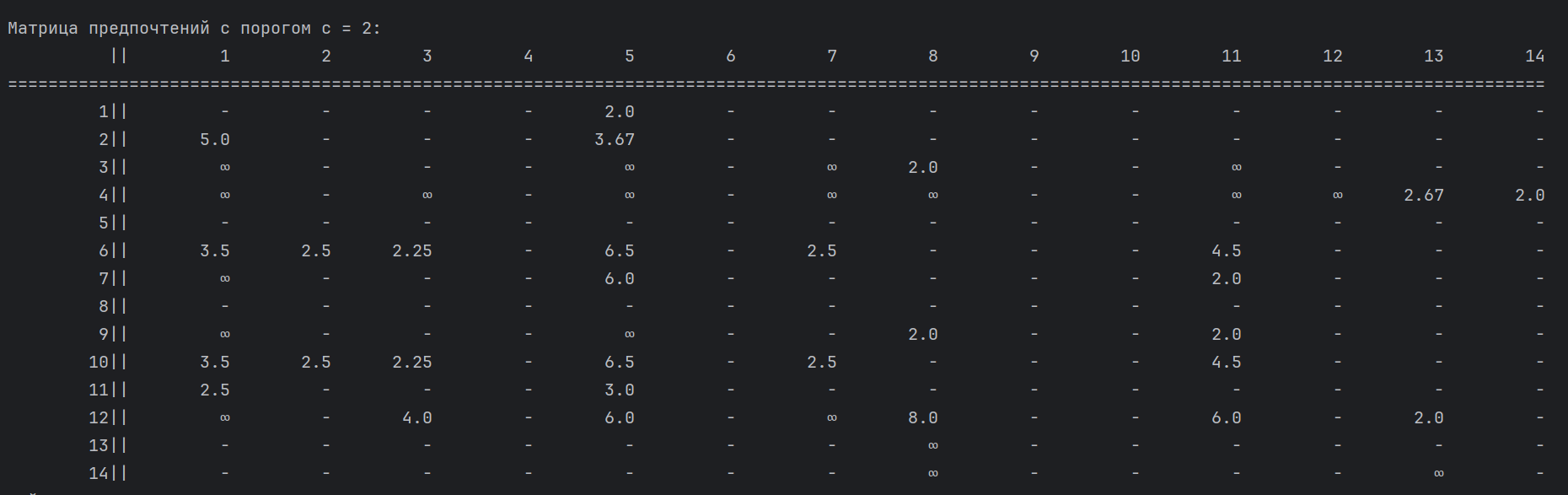


Рисунок 2.5.1 – Вывод матрицы предпочтений Рисунок 2.5.2 – Вывод матрицы предпочтений с порогом = 1.5

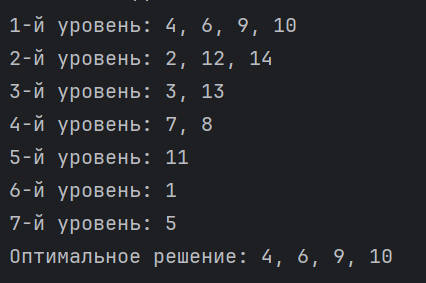
****

Рисунок 2.5.3 – Результат выполнения программы

## 2.6 Выводы по разделу

В ходе данной практической работы изучен метод Электра II и применён для нахождения идеальной популярной песни. Преимуществами метода является большая объективность по сравнению с методом Парето и его методами сужения, однако чтобы метод дал единственное решение, нужно экспериментально подбирать значение порога для того, чтобы на графе предпочтений не образовывалось циклов. Очень часто алгоритм выдаёт несколько оптимальных решений. Алгоритм имеет квадратичную сложность, как и метод Парето.

# 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 идеальную популярную песню.

## 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.

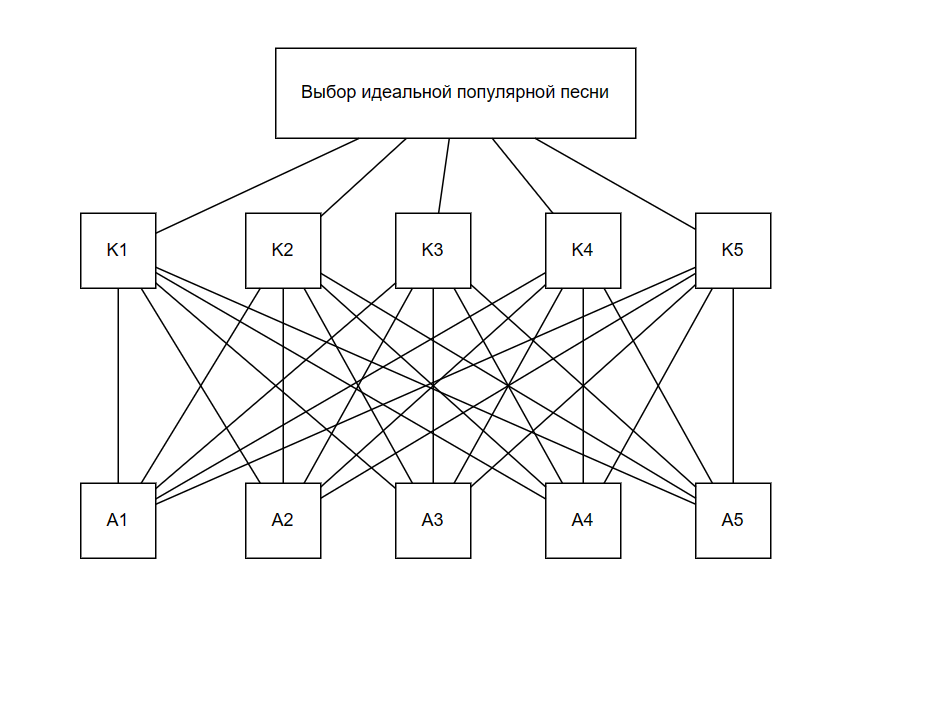
**

Рисунок 3.3.1 – Полная доминантная иерархия

Критерии:

К 1 – количество слушателей (млн) (+);

К 2 – рейтинг (+);

К 3 – темп (BPM) (+);

К 4 – длина (мин.:сек.) (-);

К 5 – агрессивность (-).

Альтернативы:

А 1 – Kendrick Lamar – HUMBLE.;

А 2 – The Weeknd – Blinding Lights;

А 3 – Tones And I – Dancing Monkey;

А 4 – Michael Jackson – Billie Jean;

А 5 – Ed Sheeran – Shape Of You.

## 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 | 1/3 | 5 | 3 | 7 | 2.036 | 0.273 |
| К 2 | 3 | 1 | 6 | 4 | 9 | 3.65 | 0.49 |
| К 3 | 1/5 | 1/6 | 1 | 1/2 | 3 | 0.549 | 0.074 |
| К 4 | 1/3 | 1/4 | 2 | 1 | 5 | 0.964 | 0.129 |
| К 5 | 1/7 | 1/9 | 1/3 | 1/5 | 1 | 0.254 | 0.034 |
| ∑Vi | | | | | | 7.453 |

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\*1/3\*5\*3\*7)1/5 = 2.036;

Строка № 2

V2=(3\*1\*6\*4\*9)1/5 = 3.65;

Строка № 3

V3=(1/5\*1/6\*1\*1/2\*3)1/5 = 0.725;

Строка № 4

V4=(1/3\*1/4\*2\*1\*5)1/5 = 0.964;

Строка № 5

V5=(1/7\*1/9\*1/3\*1/5\*1)1/5 =.

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

∑Vi = V1 + V2 + V3 + V4 + V5 = 2.036 + 3.65 + 0.549 + 0.964 + 0.254 = 7.453.

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

***Строка № 1***

W21= 2.036 / ∑Vi = 0.273 = Y21;

Строка № 2

W22= 3.65 / ∑Vi = 0.49 = Y22;

***Строка № 3***

W23= 0.549 / ∑Vi = 0.074 = Y23;

***Строка № 4***

W24= 0.964 / ∑Vi = 0.129 = Y24;

***Строка № 5***

W25= 0.254 / ∑Vi = 0.034 = Y25.

В результате получен вектор приоритетов:

W2i = (Y21=0.273; Y22=0.49; Y23=0.074; Y24=0.129; Y25=0.034), где индекс 2 означает, что вектор приоритетов относится ко второму уровню иерархии.

К 1 – количество прослушиваний (Таблица 3.5.2);

Таблица 3.5.2 – Матрица сравнения по критерию 1.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| К1 | А1 | А2 | А3 | А4 | А5 | VК1Y | W3К1Y |
| А1 | 1 | 1/7 | 1/4 | 3 | 1/5 | 0.464 | 0.063 |
| А2 | 7 | 1 | 3 | 9 | 2 | 3.277 | 0.444 |
| А3 | 4 | 1/3 | 1 | 5 | 1/3 | 1.173 | 0.159 |
| А4 | 1/3 | 1/9 | 1/5 | 1 | 1/7 | 0.254 | 0.034 |
| А5 | 5 | 1/2 | 3 | 7 | 1 | 2.208 | 0.299 |
| ∑VК1Y | | | | | | 7.376 |

Определена относительная ценность каждого элемента.

Строка № 1

VК11=(1\*1/7\*1/4\*3\*1/5)1/5= 0.464;

Строка № 2

VК12=(7\*1\*3\*9\*2)1/5= 3.277;

Строка № 3

VК13=(4\*1/3\*1\*1/5\*1/3)1/5= 1.173;

Строка № 4

VК14=(1/3\*1/9\*1/5\*1\*1/7)1/5= 0.254;

Строка № 5

VК15=(5\*1/2\*3\*7\*1)1/5= 2.208.

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

∑VК1Y = VК11 + VК12 + VК13 + VК14 + VК15 = 0.464 + 3.277 + 1.173 + 0.254 + 2.208 = 7.376.

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

Строка № 1

W3К11= 0.464 / ∑Vi = 0.464 / 7.277 = 0.063;

Строка № 2

W3К12= 3.277 / ∑Vi = 3.178 / 7.277 = 0.444;

Строка № 3

W3К13= 1.173 / ∑Vi = 1.173 / 7.277 = 0.159;

Строка № 4

W3К14= 0.254 / ∑Vi = 0.254 / 7.277 = 0.034;

Строка № 5

W3К15= 2.208 / ∑Vi = 2.208 / 7.277 = 0.299;

В результате получаем вектор приоритетов:

W3К1Y = (Y311=0.063; Y312=0.444; Y313=0.159; Y314=0.034; Y315=0.299),

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

К 2 – рейтинг песни (Таблица 3.5.3):

Таблица 3.5.3 – Матрица сравнения по критерию 2.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| К2 | А1 | А2 | А3 | А4 | А5 | VК2Y | W3К2Y |
| А1 | 1 | 1/4 | 7 | 1/5 | 5 | 1.118 | 0.137 |
| А2 | 4 | 1 | 9 | 1/2 | 7 | 2.631 | 0.323 |
| А3 | 1/7 | 1/9 | 1 | 1/9 | 1/6 | 0.197 | 0.024 |
| А4 | 5 | 2 | 9 | 1 | 8 | 3.728 | 0.458 |
| А5 | 1/5 | 1/7 | 6 | 1/8 | 1 | 0.464 | 0.057 |
| ∑VК2Y | | | | | | 8.138 |  |

Определена относительная ценность каждого элемента.

Строка № 1

VК21=(1\*1/4\*7\*1/5\*5)1/5= 1.118;

Строка № 2

VК22=(4\*1\*9\*1/2\*7)1/5= 2.631;

Строка № 3

VК23=(1/7\*1/9\*1\*1/9\*1/6)1/5= 0.197;

Строка № 4

VК24=(5\*2\*9\*1\*8)1/5= 3.728;

Строка № 5

VК25=(1/5\*1/7\*6\*1/8\*1)1/5= 0.464.

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

∑VК2Y = VК21 + VК22 + VК23 + VК24 + VК25 = 1.118 + 2.631 + 0.197 + 3.728 + 0.464 = 8.138.

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

Строка № 1

W3К21= 1.118 / ∑Vi = 0.137;

Строка № 2

W3К22= 2.631 / ∑Vi = 0.323;

Строка № 3

W3К23= 0.197 / ∑Vi = 0.024;

Строка № 4

W3К24= 3.728 / ∑Vi = 0.458;

Строка № 5

W3К25= 0.464 / ∑Vi = 0.057.

В результате получаем вектор приоритетов:

W3К2Y = (Y321=0.137; Y322=0.323; Y323=0.024; Y324=0.458; Y325=0.057),

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

К 3 – темп песни (Таблица 3.5.4):

Таблица 3.5.4 – Матрица сравнения по критерию 3.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| К3 | А1 | А2 | А3 | А4 | А5 | VК3Y | W3К3Y |
| А1 | 1 | 1/3 | 7 | 4 | 7 | 2.307 | 0.295 |
| А2 | 3 | 1 | 8 | 5 | 8 | 3.949 | 0.505 |
| А3 | 1/7 | 1/8 | 1 | 1/3 | 1 | 0.359 | 0.046 |
| А4 | 1/4 | 1/5 | 3 | 1 | 3 | 0.852 | 0.109 |
| А5 | 1/7 | 1/8 | 1 | 1/3 | 1 | 0.359 | 0.046 |
| VК35 | | | | | | 7.826 |

Определена относительная ценность каждого элемента.

Строка № 1

VК31=(1\*1/3\*7\*4\*7)1/5= 2.307;

Строка № 2

VК32=(3\*1\*8\*5\*8)1/5= 3.949;

Строка № 3

VК33=(1/7\*1/8\*1\*1/3\*1)1/5= 0.359;

Строка № 4

VК34=(1/4\*1/5\*3\*1\*3)1/5= 0.852;

Строка № 5

VК35=(1/7\*1/8\*1\*1/3\*1)1/5= 0.359.

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

∑VК3Y = VК31 + VК32 + VК33 + VК34 + VК35 = 7.826.

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

Строка № 1

W3К31= 2.307 / ∑Vi = 0.295;

Строка № 2

W3К32= 3.949 / ∑Vi = 0.505;

Строка № 3

W3К33= 0.359 / ∑Vi = 0.046;

Строка № 4

W3К34= 0.852 / ∑Vi = 0.109;

Строка № 5

W3К35= 0.359 / ∑Vi = 0.046.

В результате получаем вектор приоритетов:

W3К3Y = (Y331=0.295; Y332=0.505; Y333=0.046; Y334=0.109; Y335=0.109),

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

К 4 – Длина песни (Таблица 3.5.5);

Таблица 3.5.5 – Матрица сравнения по критерию 4.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| К4 | А1 | А2 | А3 | А4 | А5 | VК4Y | W3К4Y |
| А1 | 1 | 3 | 3 | 8 | 6 | 3.366 | 0.473 |
| А2 | 1/3 | 1 | 1 | 6 | 4 | 1.516 | 0.213 |
| А3 | 1/3 | 1 | 1 | 5 | 4 | 1.461 | 0.205 |
| А4 | 1/8 | 1/6 | 1/5 | 1 | 1/3 | 0.268 | 0.038 |
| А5 | 1/6 | 1/4 | 1/4 | 3 | 1 | 0.5 | 0.07 |
| ∑VК4Y | | | | | | 7.111 |

Определена относительная ценность каждого элемента.

Строка № 1

VК41=(1\*3\*3\*8\*6)1/5 = 3.366;

Строка № 2

VК42=(1/3\*1\*1\*6\*4)1/5 = 1.516;

Строка № 3

VК43=(1/3\*1\*1\*5\*4)1/5 = 1.461;

Строка № 4

VК44=(1/8\*1/6\*1/5\*1\*1/3)1/5 = 0.268;

Строка № 5

VК45=(1/6\*1/4\*1/4\*3\*1)1/5 = 0.5.

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

∑VК4Y = VК41 + VК42 + VК43 + VК44 + VК45 = 7.111­.

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

Строка № 1

W3К41= 3.366 / ∑Vi = 0.473;

Строка № 2

W3К42= 1.516 / ∑Vi = 0.213;

Строка № 3

W3К43= 1.461 / ∑Vi = 0.205;

Строка № 4

W3К44= 0.268 / ∑Vi = 0.038;

Строка № 5

W3К45= 0.5 / ∑Vi = 0.07.

В результате получаем вектор приоритетов:

W3К4Y = (Y341=0.473; Y342=0.213; Y343=0.205; Y344=0.038; Y345=0.07),

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

К 5 – Агрессивность лирики (Таблица 3.5.6).

Таблица 3.5.6 – Матрица сравнения по критерию 5.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| К5 | А1 | А2 | А3 | А4 | А5 | VК5Y | W3К5Y |
| А1 | 1 | 1/9 | 1/9 | 1/9 | 1/8 | 0.176 | 0.026 |
| А2 | 9 | 1 | 2 | 2 | 3 | 2.551 | 0.378 |
| А3 | 9 | 1/2 | 1 | 1 | 2 | 1.552 | 0.23 |
| А4 | 9 | 1/2 | 1 | 1 | 2 | 1.552 | 0.23 |
| А5 | 8 | 1/3 | 1/2 | 1/2 | 1 | 0.922 | 0.137 |
| ∑VК5Y | | | | | | 6.753 |

Определена относительная ценность каждого элемента.

Строка № 1

VК51=(1\*1/9\*1/9\*1/9\*1/8)1/5 = 0.176;

Строка № 2

VК52=(9\*1\*2\*2\*3)1/5 = 2.551;

Строка № 3

VК53=(9\*1/2\*1\*1\*2)1/5 = 1.552;

Строка № 4

VК54=(9\*1/2\*1\*1\*2)1/5 = 1.552;

Строка № 5

VК55=(8\*1/3\*1/2\*1/2)1/5 = 0.922.

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

∑VК5Y = VК51 + VК52 + VК53 + VК54 + VК55 = 6.753.

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

Строка № 1

W3К51= 0.176 / ∑Vi = 0.026;

Строка № 2

W3К52= 2.551 / ∑Vi = 0.378;

Строка № 3

W3К53= 1.552 / ∑Vi = 0.23;

Строка № 4

W3К54= 1.552 / ∑Vi = 0.23;

Строка № 5

W3К55= 0.922 / ∑Vi = 0.137.

В результате получаем вектор приоритетов:

W3К5Y = (Y351=0.026; Y352=0.378; Y353=0.23; Y354=0.23; Y355=0.137),

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 идеальной популярной песни» (Таблица 3.6.1).

Таблица 3.6.1 – Матрица «Выбор идеальной популярной песни».

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Цель | К 1 | К 2 | К 3 | К 4 | К 5 | W2i |
| К 1 | 1 | 1/3 | 5 | 3 | 7 | 0.273 |
| К 2 | 3 | 1 | 6 | 4 | 9 | 0.49 |
| К 3 | 1/5 | 1/6 | 1 | 1/2 | 3 | 0.074 |
| К 4 | 1/3 | 1/4 | 2 | 1 | 5 | 0.129 |
| К 5 | 1/7 | 1/9 | 1/3 | 1/5 | 1 | 0.034 |

Определена сумма каждого столбца матрицы суждений.

S1 = 1 + 3 + 1/5 + 1/3 + 1/7 = 4.676;

S2 = 1/3 + 1 + 1/6 + 1/4 + 1/9 = 1.861;

S3 = 5 + 6 + 1 + 2 + 1/3 = 14.333;

S4 = 3 + 4 + 1/2 + 1 + 1/5 = 8.7;

S5 = 7 + 9 + 3 + 5 + 1 = 25.

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 = 1.277;

Р2 = S2 х W22 = 0.912;

Р3 = S3 х W23 = 1.061;

Р4 = S4 х W24 = 1.122;

Р5 = S5 х W25 = 0.85.

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 = 5.222.

Отклонение от согласованности выражается индексом согласованности.

ИС = (λmax - n)/(n - 1) = (5.222-5)(5-1) = 0.056.

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

ОС = ИС/СИ = 0.049.

Значение ОС меньше или равное 0.10 считается приемлемым, значит матрица «Выбор идеальной популярной песни» согласована.

Определены индекс согласованности и отношение согласованности для матрицы К 1 – количество слушателей (Таблица 3.6.2).

Таблица 3.6.2 – Матрица сравнения по критерию 1.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| К1 | А1 | А2 | А3 | А4 | А5 | W3К1Y |
| А1 | 1 | 1/7 | 1/4 | 3 | 1/5 | 0.063 |
| А2 | 7 | 1 | 3 | 9 | 2 | 0.444 |
| А3 | 4 | 1/3 | 1 | 5 | 1/3 | 0.159 |
| А4 | 1/3 | 1/9 | 1/5 | 1 | 1/7 | 0.034 |
| А5 | 5 | 1/2 | 3 | 7 | 1 | 0.299 |

Определяется сумма каждого столбца матрицы суждений.

S1К1 = 1 + 7 + 4 + 1/3 + 5 = 17.333;

S2 К1 = 1/7 + 1 + 1/3 + 1/9 + 1/2 = 2.087;

S3 К1 = 1/4 + 3 + 1 + 1/5 + 3 = 7.45;

S4 К1 = 3 + 9 + 5 + 1 + 7 = 25;

S5 К1 = 1/5 + 2 + 1/3 + 1/7 + 1 = 3.676.

Затем полученный результат умножен на компоненту нормализованного вектора приоритетов.

Р1 К1 = S1 х W3К11 = 1.092;

Р2 К1 = S2 х W3К12 = 0.927;

Р3 К1 = S3 х W3К13 = 1.185;

Р4 К1 = S1 х W3К14 = 0.85;

Р5 К1 = S1 х W3К15 = 1.099.

Найдена пропорциональность предпочтений.

λmax К1 = Р1К1 + Р2К1 + Р3К1 + Р4К1 + Р5К1 = 5.153.

Отклонение от согласованности выражается индексом согласованности.

ИС К1 = (λmax К1 - n)/(n - 1) = 0.038.

Найдено отношение согласованности ОС.

ОС К1 = ИС/СИ = 0.034.

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

Определены индекс согласованности и отношение согласованности для матрицы К 2 – рейтинг песни (Таблица 3.6.3).

Таблица 3.6.3 – Матрица сравнения по критерию 2.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| К2 | А1 | А2 | А3 | А4 | А5 | W3К2Y |
| А1 | 1 | 1/4 | 7 | 1/5 | 5 | 0.137 |
| А2 | 4 | 1 | 9 | 1/2 | 7 | 0.323 |
| А3 | 1/7 | 1/9 | 1 | 1/9 | 1/6 | 0.024 |
| А4 | 5 | 2 | 9 | 1 | 8 | 0.458 |
| А5 | 1/5 | 1/7 | 6 | 1/8 | 1 | 0.057 |

Определена сумма каждого столбца матрицы суждений.

S1К2 = 1 + 4 + 1/7 + 5 + 1/5 = 10.343;

S2 К2 = 1/4 + 1 + 1/9 + 2 + 1/7 = 3.504;

S3 К2 = 7 + 9 + 1 + 9 + 6 = 32;

S4 К2 = 1/5 + 1/2 + 1/9 + 1 + 1/8 = 1.936;

S5 К2 = 5 + 7 + 1/6 + 8 + 1 = 21.167.

Затем полученный результат умножен на компоненту нормализованного вектора приоритетов.

Р1 К2 = S1 х W3 К21 = 1.417;

Р2 К2 = S2 х W3 К22 = 1.132;

Р3 К2 = S3 х W3 К23 = 0.768;

Р4 К2 = S4 х W3 К24 = 0.887;

Р5 К2 = S5 х W3 К25 = 1.207.

Найдена пропорциональность предпочтений.

λmax К2 = Р1К2 + Р2К2 + Р3К2 + Р4К2 + Р5К2 = 5.411.

Отклонение от согласованности выражается индексом согласованности.

ИС К2 = (λmax К2 - n)/(n - 1) = (5.411 – 5) / (5 – 1) = 0.103.

Найдено отношение согласованности ОС.

ОС К2 = ИС/СИ = 0.092.

Значение ОС меньше или равное 0.10 считается приемлемым, значит матрица К 2 (рейтинг песни) согласована.

Определены индекс согласованности и отношение согласованности для матрицы К 3 – темп песни (Таблица 3.6.4).

Таблица 3.6.4 – Матрица сравнения по критерию 3.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| К3 | А1 | А2 | А3 | А4 | А5 | W3К3Y |
| А1 | 1 | 1/3 | 7 | 4 | 7 | 0.295 |
| А2 | 3 | 1 | 8 | 5 | 8 | 0.505 |
| А3 | 1/7 | 1/8 | 1 | 1/3 | 1 | 0.046 |
| А4 | 1/4 | 1/5 | 3 | 1 | 3 | 0.109 |
| А5 | 1/7 | 1/8 | 1 | 1/3 | 1 | 0.046 |

Определена сумма каждого столбца матрицы суждений.

S1К3 = 1 + 3 + 1/7 + 1/4 + 1/7 = 4.536;

S2 К3 = 1/3 + 1 + 1/8 + 1/5 + 1/8 = 1.783;

S3 К3 = 7 + 8 + 1 + 3 + 1 = 20;

S4 К3 = 4 + 5 + 1/3 + 1 + 1/3 = 10.667;

S5 К3 = 7 + 8 + 1 + 3 + 1 = 20.

Затем полученный результат умножен на компоненту нормализованного вектора приоритетов.

Р1 К3 = S1 х W3 К31 = 1.338;

Р2 К3 = S2 х W3 К32 = 0.9;

Р3 К3 = S3 х W3 К33 = 0.92;

Р4 К3 = S4 х W3 К34 = 1.163;

Р5 К3 = S5 х W3 К35 = 0.92.

Найдем пропорциональность предпочтений.

λmax К3 = Р1К3 + Р2К3 + Р3К3 + Р4К3 + Р5К3 = 5.241.

Отклонение от согласованности выражается индексом согласованности.

ИС К3 = (λmax К3 - n)/(n - 1) = 0.06.

Найдено отношение согласованности ОС.

ОС К3 = ИС/СИ = 0.054.

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

Определены индекс согласованности и отношение согласованности для матрицы К 4 – длина песни (Таблица 3.6.5).

Таблица 3.6.5 – Матрица сравнения по критерию 4.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| К4 | А1 | А2 | А3 | А4 | А5 | W3К4Y |
| А1 | 1 | 3 | 3 | 8 | 6 | 0.473 |
| А2 | 1/3 | 1 | 1 | 6 | 4 | 0.213 |
| А3 | 1/3 | 1 | 1 | 5 | 4 | 0.205 |
| А4 | 1/8 | 1/6 | 1/5 | 1 | 1/3 | 0.038 |
| А5 | 1/6 | 1/4 | 1/4 | 3 | 1 | 0.07 |

Определена сумма каждого столбца матрицы суждений.

S1К4 = 1 + 1/3 + 1/3 + 1/8 + 1/6 = 1.958;

S2К4 = 3 + 1 + 1 + 1/6 + 1/4 = 5.417;

S3К4 = 3 + 1 + 1 + 1/5 + 1/4 = 5.45;

S4К4 = 8 + 6 + 5 + 1 + 3 = 23;

S5К4 = 6 + 4 + 4 + 1/3 + 1 = 15.333.

Затем полученный результат умножен на компоненту нормализованного вектора приоритетов.

Р1К4 = S1 х W3 К41 = 0.926;

Р2К4 = S2 х W3 К42 = 1.154;

Р3К4 = S3 х W3 К43 = 1.117;

Р4К4 = S4 х W3 К44 = 0.874;

Р5К4 = S5 х W3 К45 = 1.073.

Найдена пропорциональность предпочтений.

λmax К4 = Р1К4 + Р2К4 + Р3К4 + Р4К4 + Р5К4 = 5.144.

Отклонение от согласованности выражается индексом согласованности.

ИС К4 = (λmax К4 - n)/(n - 1) = 0.036.

Найдено отношение согласованности ОС.

ОС К4 = ИС/СИ = 0.032.

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

Определены индекс согласованности и отношение согласованности для матрицы К 5 – агрессивность (Таблица 3.6.6).

Таблица 3.6.6 – Матрица сравнения по критерию 5.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| К5 | А1 | А2 | А3 | А4 | А5 | W3К5Y |
| А1 | 1 | 1/9 | 1/9 | 1/9 | 1/8 | 0.026 |
| А2 | 9 | 1 | 2 | 2 | 3 | 0.378 |
| А3 | 9 | 1/2 | 1 | 1 | 2 | 0.23 |
| А4 | 9 | 1/2 | 1 | 1 | 2 | 0.23 |
| А5 | 8 | 1/3 | 1/2 | 1/2 | 1 | 0.137 |

Определена сумма каждого столбца матрицы суждений.

S1К5 = 1 + 9 + 9 + 9 + 8 = 36;

S2К5 = 1/9 + 1 + 1/2 + 1/2 + 1/3 = 2.444;

S3К5 = 1/9 + 2 + 1 + 1 + 1/2 = 4.611;

S4К5 = 1/8 + 3 + 2 + 2 + 1 = 4.611;

S5К5 = 1/8 + 3 + 2 + 2 + 1 = 8.125.

Затем полученный результат умножен на компоненту нормализованного вектора приоритетов.

Р1К5 = S1 х W3 К41 = 0.936;

Р2К5 = S2 х W3 К42 = 0.924;

Р3К5 = S3 х W3 К43 = 1.061;

Р4К5 = S1 х W3 К44 = 1.061;

Р5К5 = S1 х W3 К45 = 1.113.

Найдена пропорциональность предпочтений.

λmax К5 = Р1К5 + Р2К5 + Р3К5 + Р4К5 + Р5К5 = 5.095.

Отклонение от согласованности выражается индексом согласованности.

ИС К5 = (λmax К5 - n)/(n - 1) = 0.024.

Найдено отношение согласованности ОС.

ОС К5 = ИС/СИ = 0.021.

Значение ОС меньше или равное 0,10 считается приемлемым, значит матрица К 5 (агрессивность) 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=0.273; Y22=0.49; Y23=0.074; Y24=0.129; Y25=0.034);

W3К1Y = (Y311=0.063; Y312=0.444; Y313=0.159; Y314=0.034; Y315=0.299);

W3К2Y = (Y321=0.137; Y322=0.323; Y323=0.024; Y324=0.458; Y325=0.057);

W3К3Y = (Y331=0.295; Y332=0.505; Y333=0.046; Y334=0.109; Y335=0.109);

W3К4Y = (Y341=0.473; Y342=0.213; Y343=0.205; Y344=0.038; Y345=0.07);

W3К5Y = (Y351=0.026; Y352=0.378; Y353=0.23; Y354=0.23; Y355=0.137).

Приоритеты альтернатив получены следующим образом:

W1 = W21 х W3К11 + W22 х W3К21 + W23 х W3К31 + W24 х W3К41 + W25 х W3К51 = 0.273 \* 0.063 + 0.49 \* 0.137 + 0.074 \* 0.295 + 0.129 \* 0.473 + 0.034 \* 0.026 = 0.168.

W2 = W21 х W3К12 + W22 х W3К22 + W23 х W3К32 + W24 х W3К42 + W25 х W3К52 = 0.273 \* 0.444 + 0.49 \* 0.323 + 0.074 \* 0.505 + 0.129 \* 0.213 + 0.034 \* 0.378 = 0.357.

W3 = W21 х W3К13 + W22 х W3К23 + W23 х W3К33 + W24 х W3К43 + W25 х W3К53 = 0.273 \* 0.159 + 0.49 \* 0.024 + 0.074 \* 0.046 + 0.129 \* 0.205 + 0.034 \* 0.23 = 0.093.

W4 = W21 х W3К14 + W22 х W3К24 + W23 х W3К34 + W24 х W3К44 + W25 х W3К54 = 0.273 \* 0.034 + 0.49 \* 0.458 + 0.074 \* 0.109 + 0.129 \* 0.038 + 0.034 \* 0.23 = 0.254.

W5 = W21 х W3К15 + W22 х W3К25 + W23 х W3К35 + W24 х W3К45 + W25 х W3К55 = 0.273 \* 0.299 + 0.49 \* 0.057 + 0.074 \* 0.046 + 0.129 \* 0.07 + 0.034 \* 0.137 = 0.127.

Таким образом, приоритеты альтернатив равны:

альтернатива А1 (Kendrick Lamar – HUMBLE.) - W1 приоритет равен = 0.168;

альтернатива А2 (The Weeknd – Blinding Lights)- W2 приоритет равен = 0.357;

альтернатива А3 (Tones And I – Dance Monkey) - W3 приоритет равен = 0.093;

альтернатива А4 (Michael Jackson – Billie Jean) – W4 приоритет равен = 0.254;

альтернатива А5 (Ed Sheeran – Shape Of You) - W5 приоритет равен = 0.127.

## 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 – 3.9.8.

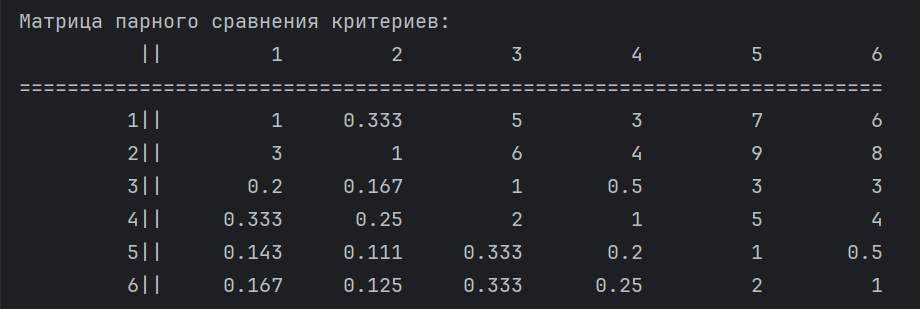


Рисунок 3.9.1 – Матрица парного сравнения критериев

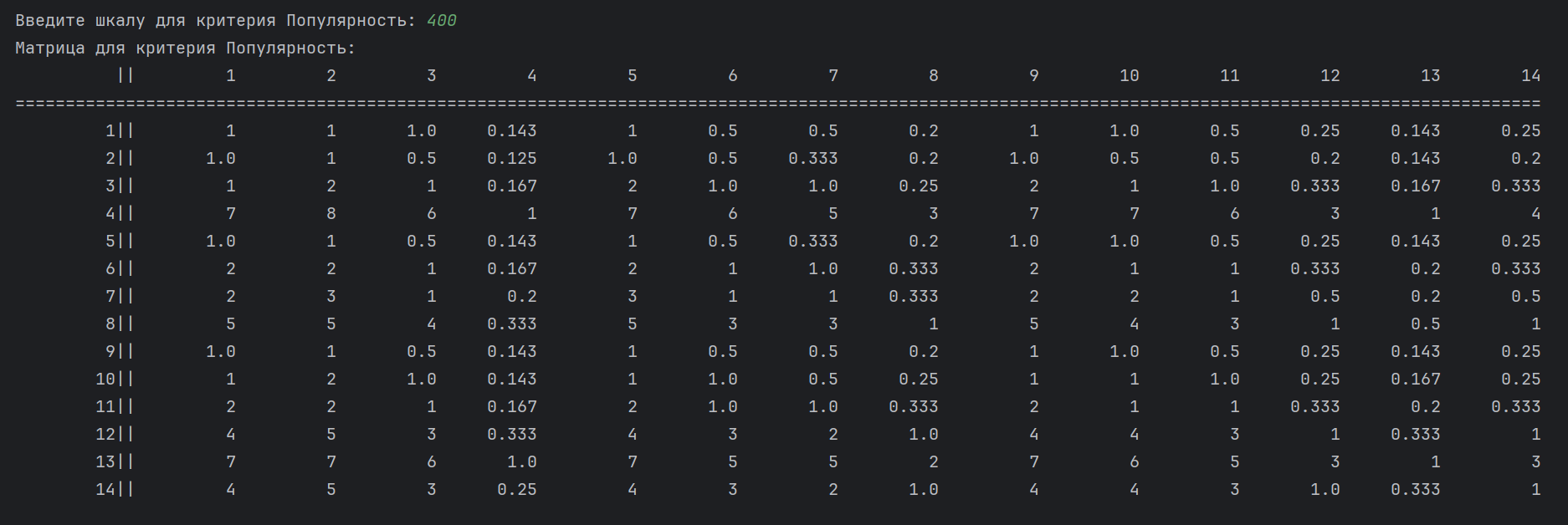


Рисунок 3.9.2 – Сгенерированная матрица для первого критерия

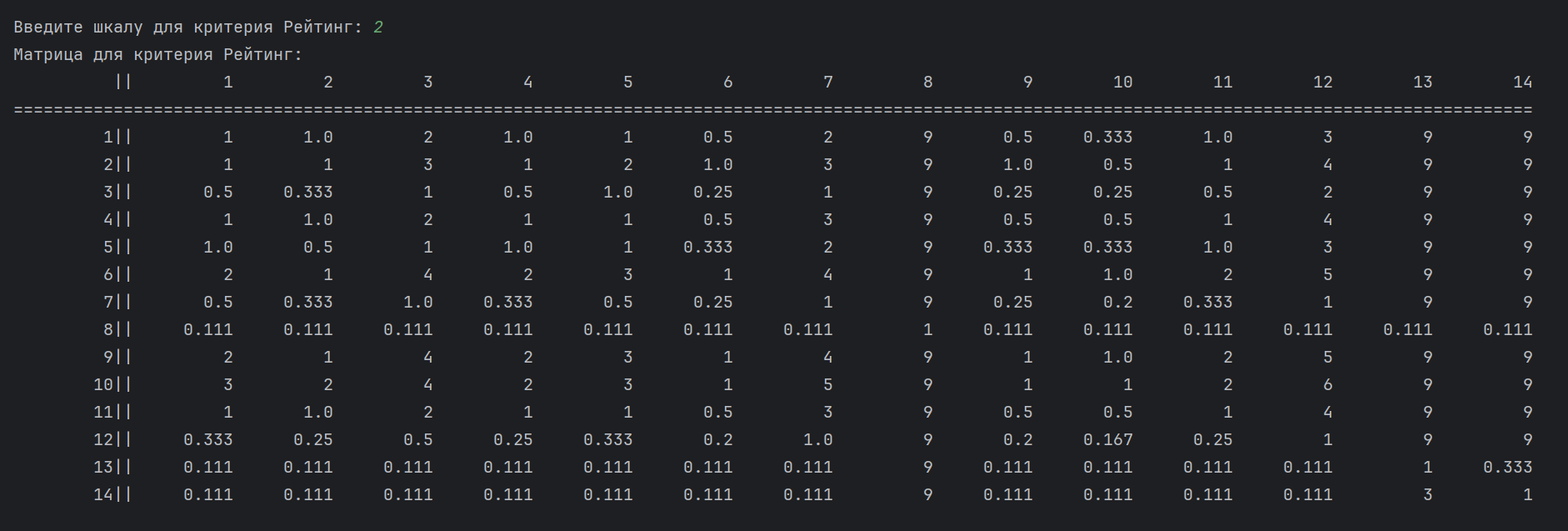


Рисунок 3.9.3 – Сгенерированная матрица для второго критерия

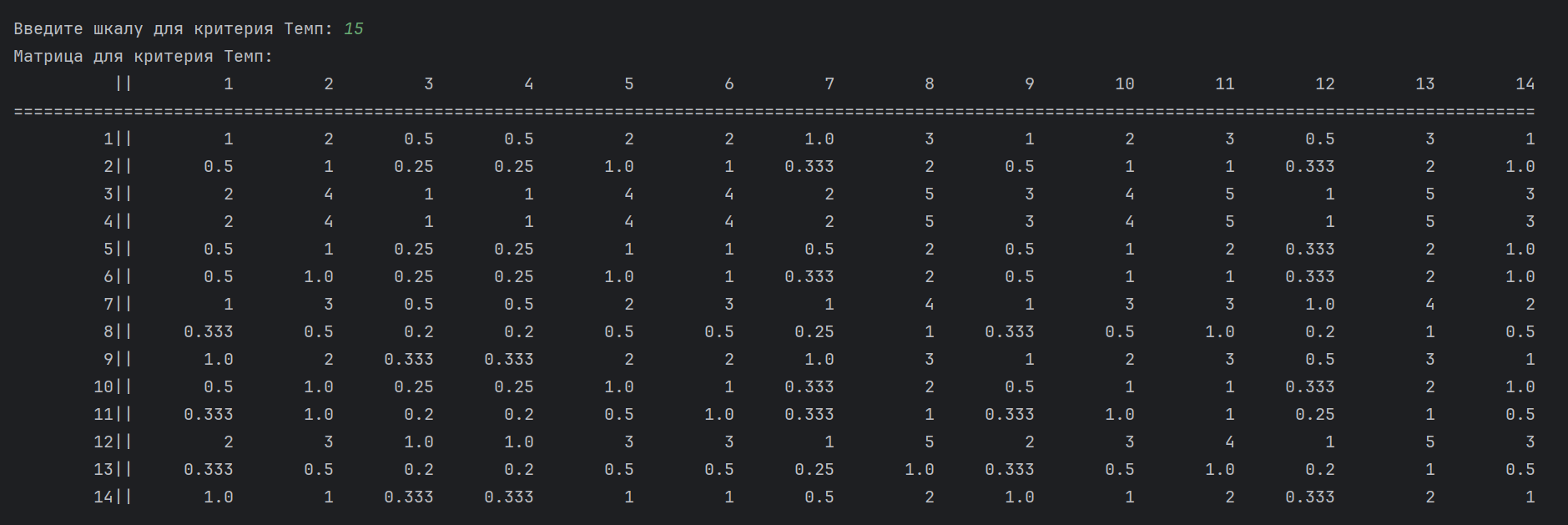


Рисунок 3.9.4 – Сгенерированная матрица для третьего критерия

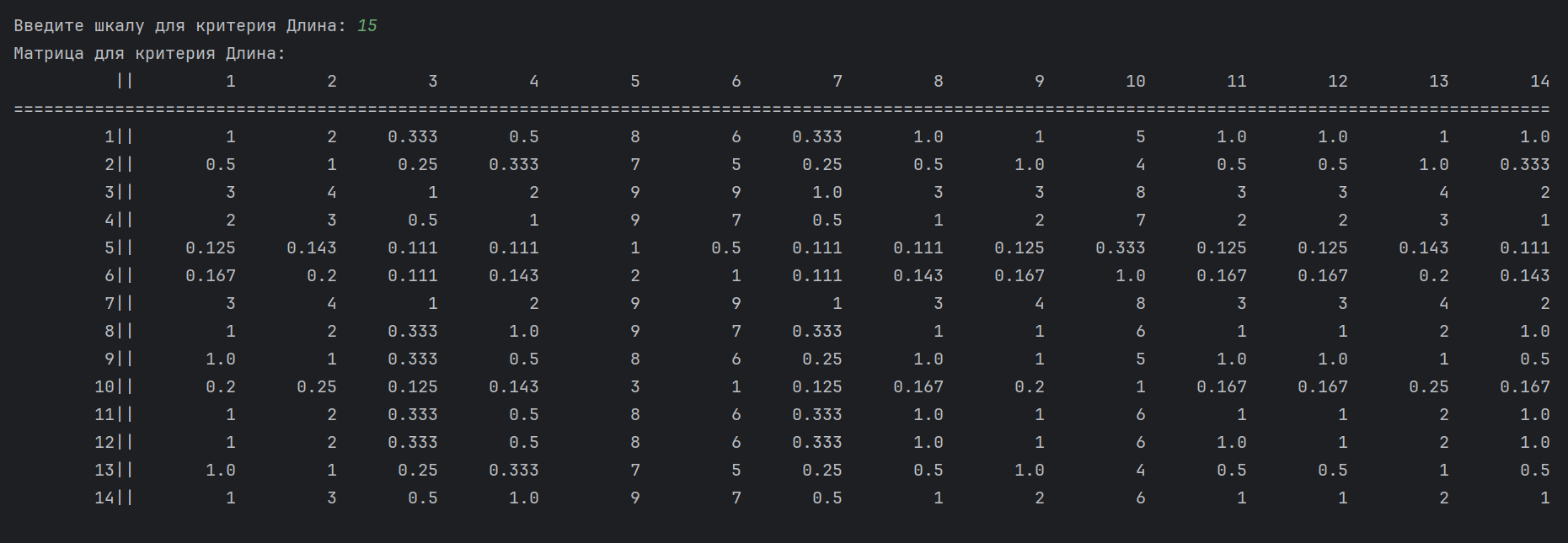


Рисунок 3.9.5 – Сгенерированная матрица для четвёртого критерия

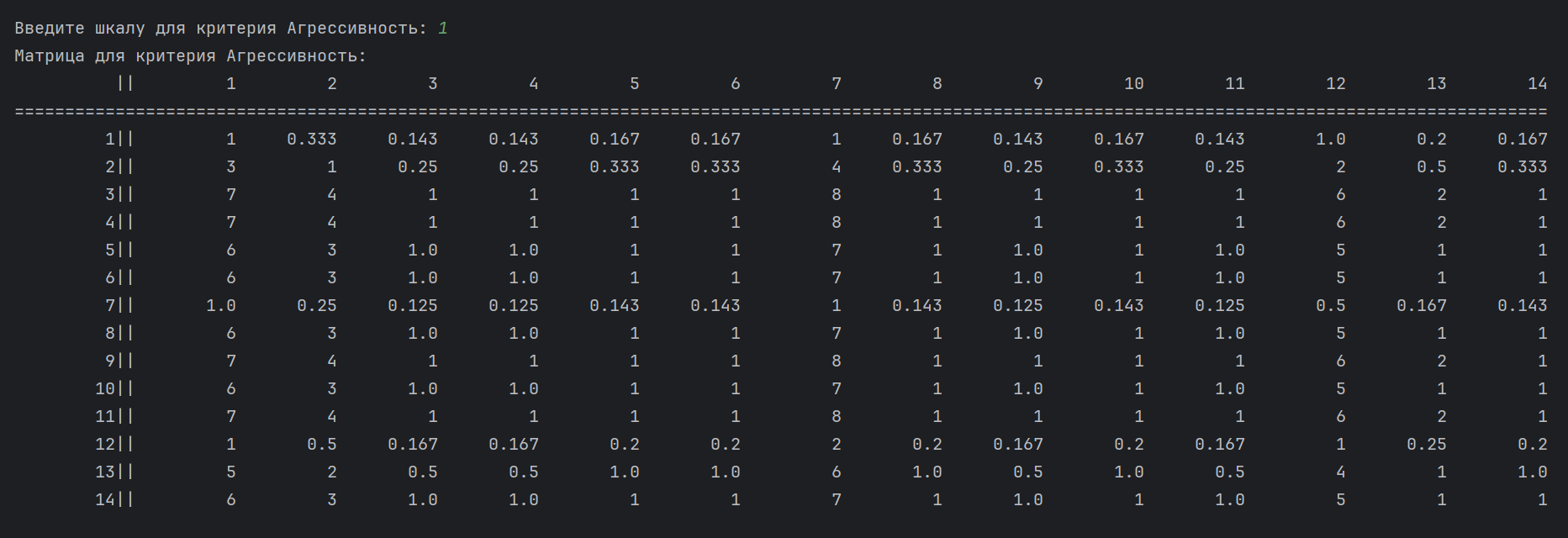


Рисунок 3.9.6 – Сгенерированная матрица для пятого критерия

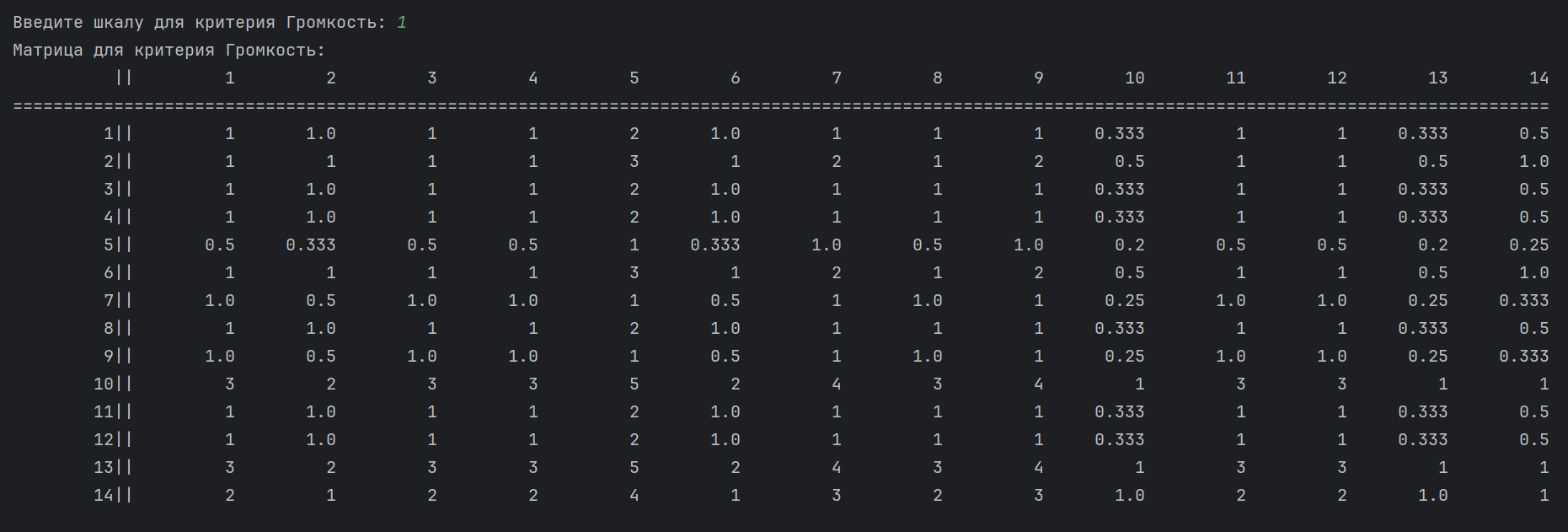


Рисунок 3.9.7 – Сгенерированная матрица для шестого критерия

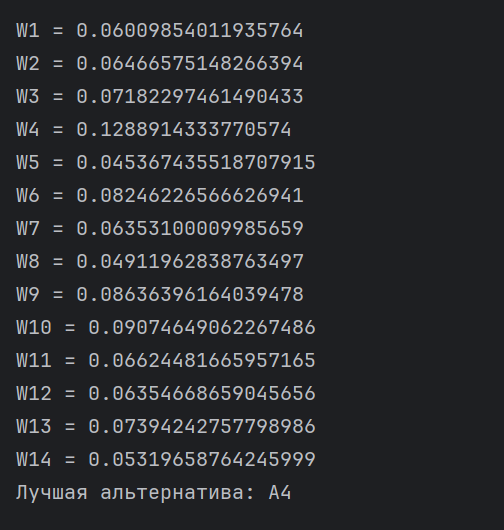


Рисунок 3.9.8 – Результат работы программы

## 3.10 Выводы по разделу

В ходе данной работы 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

Задача линейного программирования состоит в нахождении оптимального решения, т.е. вектора, на который накладывают ряд ограничений. К ним относятся система линейных неравенств, требование на неотрицательность переменных. Оптимальность заключается в нахождение минимума или максимума целевой линейной функции.

Графический метод применяется для задач, в которых присутствует 2 переменные. Он заключается в построении области допустимых решений на плоскости и графика целевой функции, и по построенному чертежу определяются максимум или минимум целевой функции.

## 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:

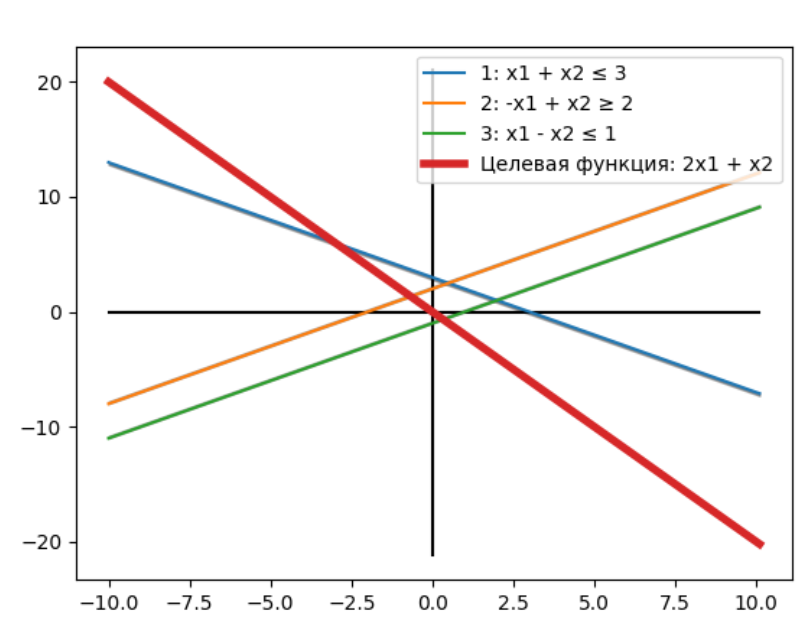
1. – значения от 0 до 10 с шагом 0,5;
2. – значения ограничения.
3. – значения ограничения.
4. – значения ограничения.
5. – значения целевой функции при условии = 0

Таблица 4.4.1 – Данные для графика

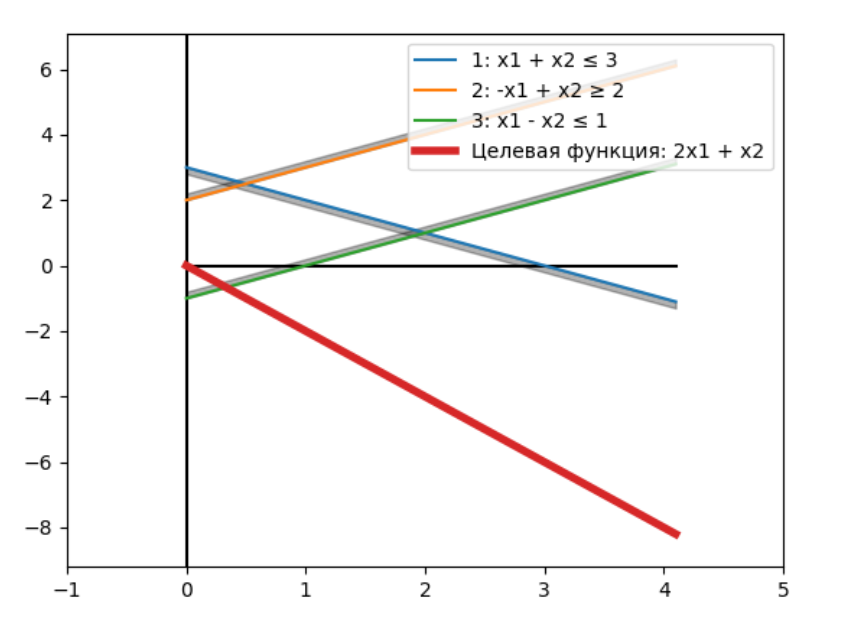
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| x1 | x2=3-х1 | x2 =2+х1 | x2=х1-1 | x2=-2х1 |
| 0 | 3 | 2 | -1 | 0 |
| 0,5 | 2,5 | 2,5 | -0,5 | -1 |
| 1 | 2 | 3 | 0 | -2 |
| 1,5 | 1,5 | 3,5 | 0,5 | -3 |
| 2 | 1 | 4 | 1 | -4 |
| 2,5 | 0,5 | 4,5 | 1,5 | -5 |
| 3 | 0 | 5 | 2 | -6 |
| 3,5 | -0,5 | 5,5 | 2,5 | -7 |
| 4 | -1 | 6 | 3 | -8 |
| 4,5 | -1,5 | 6,5 | 3,5 | -9 |
| 5 | -2 | 7 | 4 | -10 |
| 5,5 | -2,5 | 7,5 | 4,5 | -11 |
| 6 | -3 | 8 | 5 | -12 |
| 6,5 | -3,5 | 8,5 | 5,5 | -13 |
| 7 | -4 | 9 | 6 | -14 |
| 7,5 | -4,5 | 9,5 | 6,5 | -15 |
| 8 | -5 | 10 | 7 | -16 |
| 8,5 | -5,5 | 10,5 | 7,5 | -17 |
| 9 | -6 | 11 | 8 | -18 |
| 9,5 | -6,5 | 11,5 | 8,5 | -19 |
| 10 | -7 | 12 | 9 | -20 |

## 4.5 Построение графика

Выделим таблицу подготовленных данных и построим гладкий график (Рисунок 4.5.1). Произведем настройку шага координатной оси x1 и получим следующий график (Рисунок 4.5.2)

**

**Рисунок 4.5.1 – Построение графиков по данным**

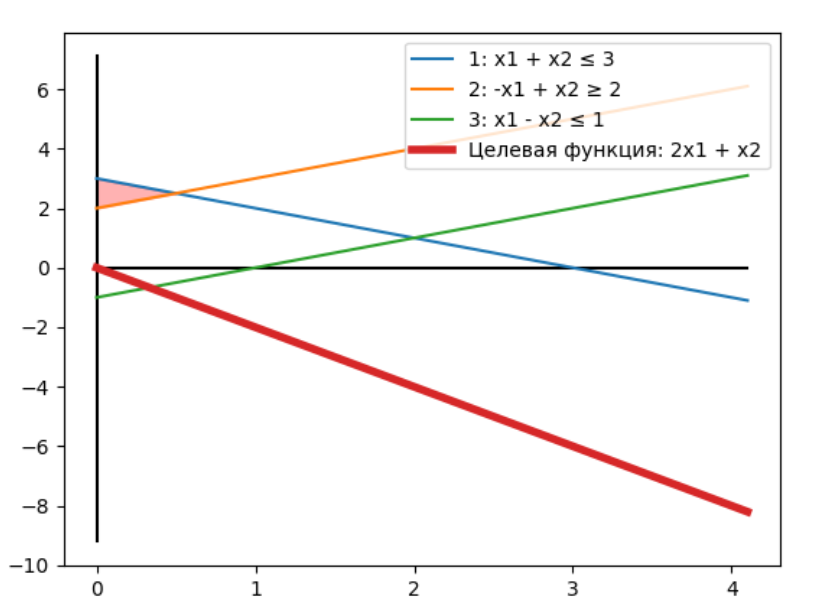
****

**Рисунок 4.5.2 – Отмасштабированный график**

## 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.

**

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

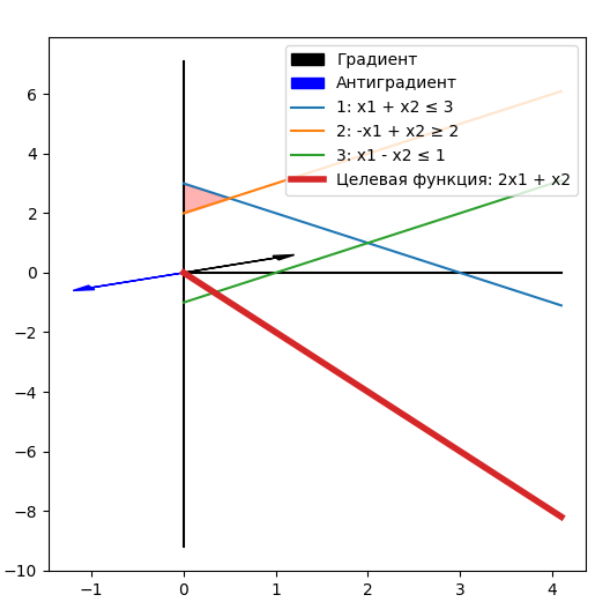
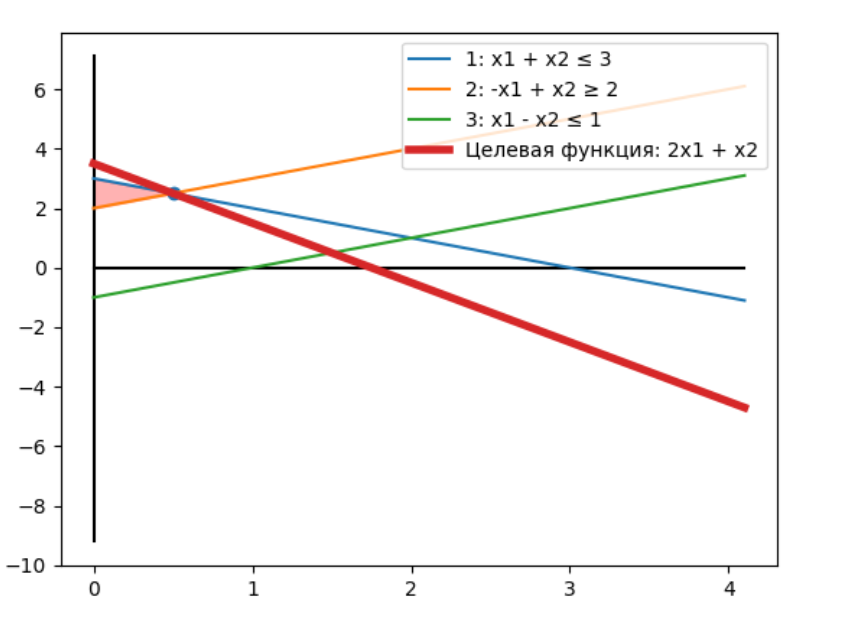
****

Рисунок 4.7.1 – Градиент и антиградиент целевой функции

Теперь начинаем мысленно сдвигать прямую целевой функции в направлении градиента, и определяем последнюю точку ОДР, которая лежит на пути прямой. Найдем её координаты:

Изобразим эту точку на графике (Рисунок 4.7.2).

**

**Рисунок 4.7.2 – Точка максимума функции**

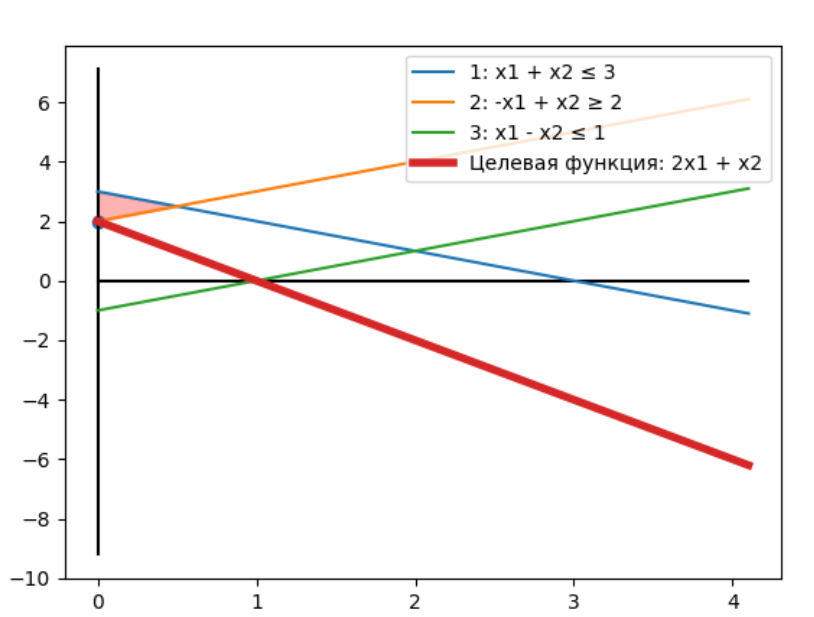
Найдем значение функции в точке максимума.

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

Получим значение равное F(x)max = 2 \* 0.5 + 2.5 = 3.5.

## 4.8 Минимум функции

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



**Рисунок 4.8.1 – Точка минимума функции**

Найдем координаты точки минимума:

В результате получим точку с координатами (0,2). 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 = 2 \* 0 + 2 = 2

Ответ:

F(x)max = 3.5.

F(x)min = 2.

## 4.9 Результат работы программы

На Рисунке 4.9.1 продемонстрированы финальные расчёты программы, реализующей графический метод.

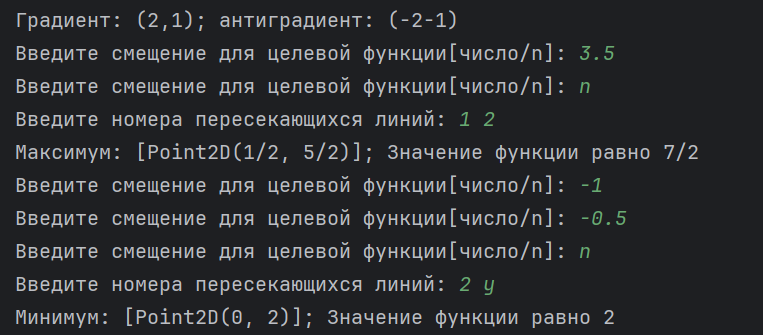


Рисунок 4.9.1 – Результат работы программы

## 4.10 Выводы по разделу

В ходе выполнения данной работы изучен графический метод решения задач линейного программирования, применён на практике, а также была разработана программа для решения данного метода.

Плюсом метода является его простота и очевидность, однако применять его возможно только для маленького числа переменных.

# 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

Задача. Фирма поставляет компьютеры под ключ четырех базовых комплектаций: «домашний», «игровой», «офисный» и «экстрим». Известны средние затраты времени на сборку, проверку и подключение компьютеров. Каждый компьютер приносит определенный уровень прибыли, но спрос ограничен. Кроме того, в плановом периоде ограничен ресурс человеко-часов, отведенных на выполнение каждой производственной операции (Таблица 5.2.1).

Таблица 5.2.1. Исходные данные задачи.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Компьютер | Прибыль за модель  у.е. | Максимальный спрос на товар | Требуется часов на подключение | Требуется часов на сборку | Требуется часов на проверку |
| Домашний | 33 | 87 | 0,9 | 1,2 | 1,3 |
| Игровой | 39 | 67 | 1,1 | 1,5 | 1,5 |
| Офисный | 36 | 110 | 0,7 | 0,9 | 0,9 |
| Экстрим | 43 | 45 | 1,3 | 1,1 | 1,2 |
| Доступно чел.-час. на каждую  операцию | | | 70 | 55 | 35 |

Определить, сколько компьютеров каждого типа необходимо произвести в плановом периоде, имея целью максимизировать прибыль.

## 5.3 Математическая модель задачи

Пусть х1 – количество домашних компьютеров, х2 – количество игровых компьютеров, х3 – количество офисных компьютеров, х4 – количество экстрим компьютеров. Прибыль от продажи компьютеров составит 33х1 + 39х2 + 36х3 + 43х4, 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: х5 ≥ 0, х6 ≥ 0, х7 ≥ 0, х8 ≥ 0, х9 ≥ 0, х10 ≥ 0, х11 ≥ 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 𝐴5, 𝐴6, 𝐴7, A8, A9, A10, A11 являются линейно независимыми единичными векторами 7-мерного пространства и образуют базис этого пространства.

Поэтому за базисные переменные выбираем переменные 𝑥5, 𝑥6, 𝑥7, x8, x9, x10, x11. Небазисными переменными являются 𝑥1, 𝑥2, х3, x4. Разложение позволяет найти первое базисное допустимое решение.

Для этого свободные переменные 𝑥1, 𝑥2, х3, x4 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 𝑥5, 𝑥6, x7, x8, x9, x10, x11, образующие базис, в верхней строке – небазисные переменные 𝑥1, 𝑥2, x3, x4. В строке 𝑐j запишем коэффициенты целевой функции, соответствующие небазисным переменным с1 = 33, с2 = 39, c3 = 36, c4 = 43. 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 . И столбец, определяемый переменной х3, состоит из коэффициентов вектора , столбец, определяемый переменной х4, 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, ∆3, ∆4 и значение целевой функции 𝑄.

Таблица 5.4.1 – Начальная симплекс-таблица задачи о максимальном доходе

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | сj | 33 | 39 | 36 | 43 |  |
|  |  | X1 | X2 | X3 | X4 |  |
| 0 | X5 | 0.9 | 1.1 | 0.7 | 1.3 | 70 |
| 0 | X6 | 1.2 | 1.5 | 0.9 | 1.1 | 55 |
| 0 | X7 | 1.3 | 1.5 | 0.9 | 1.2 | 35 |
| 0 | X8 | 1 | 0 | 0 | 0 | 87 |
| 0 | X9 | 0 | 1 | 0 | 0 | 67 |
| 0 | X10 | 0 | 0 | 1 | 0 | 110 |
| 0 | X11 | 0 | 0 | 0 | 1 | 45 |
|  | f |  |  |  |  |  |
|  |  | Δ1 | Δ2 | Δ3 | Δ­­4 | Q |

Таблица 5.4.2 – Заполнение f-строки

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | сj | 33 | 39 | 36 | 43 |  |  |
|  |  | X1 | X2 | X3 | X4 |  |  |
| 0 | X5 | 0.9 | 1.1 | 0.7 | 1.3 | 70 | 70 / 1.3 = 53.846 |
| 0 | X6 | 1.2 | 1.5 | 0.9 | 1.1 | 55 | 55 / 1.1 = 50 |
| 0 | X7 | 1.3 | 1.5 | 0.9 | 1.2 | 35 | 35 / 1.2 = 29.167 min |
| 0 | X8 | 1 | 0 | 0 | 0 | 87 | - |
| 0 | X9 | 0 | 1 | 0 | 0 | 67 | - |
| 0 | X10 | 0 | 0 | 1 | 0 | 110 | - |
| 0 | X11 | 0 | 0 | 0 | 1 | 45 | 45 / 1 = 45 |
|  | f | -33 | -39 | -36 | -43 | 0 |  |
|  |  | Δ1 | Δ2 | Δ3 | Δ­­4 | Q |  |

Для оптимальности опорного решения в задаче на максимум требуется выполнение неотрицательности всех относительных оценок ∆i ≥ 0. Так как оценки ∆1= −33, ∆2= −39, ∆3= −36 и ∆4= −43 в f-строке отрицательны, то это свидетельствуют о возможности улучшения полученного решения. Наибольшая по модулю отрицательная оценка ∆4 = −43. В базис будет включена соответствующая ей небазисная переменная 𝑥4. Составим отношения свободных членов к положительным элементам разрешающего столбца. Данные отношения приведены справа от таблицы. Наименьшему частному соответствует строка с переменной 𝑥7. Эта переменная исключается из базиса. В Таблице 1.3 разрешающий столбец и разрешающая строка выделены. Разрешающим элементом является число 𝑎34 = 1.2.

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 | 33 | 39 | 36 | 0 |  |
|  |  | X1 | X2 | X3 | X7 |  |
| 0 | X5 |  |  |  |  | 70 |
| 0 | X6 |  |  |  |  | 55 |
| 43 | X4 |  |  |  | 5/6 | 35 |
| 0 | X8 |  |  |  |  | 87 |
| 0 | X9 |  |  |  |  | 67 |
| 0 | X10 |  |  |  |  | 110 |
| 0 | X11 |  |  |  |  | 45 |
|  | f |  |  |  |  |  |
|  |  | Δ1 | Δ2 | Δ3 | Δ­­4 | Q |

В Таблице 5.4.3 переменные x7 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 | 33 | 39 | 36 | 0 |  |
|  |  | X1 | X2 | X3 | X7 |  |
| 0 | X5 |  |  |  | -13/12 | 70 |
| 0 | X6 |  |  |  | -11/12 | 55 |
| 43 | X4 | 13/12 | 5/4 | 3/4 | 5/6 | 35 |
| 0 | X8 |  |  |  | 0 | 87 |
| 0 | X9 |  |  |  | 0 | 67 |
| 0 | X10 |  |  |  | 0 | 110 |
| 0 | X11 |  |  |  | -5/6 | 45 |
|  | f |  |  |  | 35.833 |  |
|  |  | Δ1 | Δ2 | Δ3 | Δ­­4 | Q |

Таблица 5.4.5 – Итерация 0

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | сj | 33 | 39 | 36 | 0 |  |  |
|  |  | X1 | X2 | X3 | X7 |  |  |
| 0 | X5 | -0.508 | -0.525 | -0.275 | -1.083 | 32.083 | - |
| 0 | X6 | 0.008 | 0.125 | 0.075 | -0.917 | 22.917 | 22.917 / 0.075 = 305.56 |
| 43 | X4 | 1.083 | 1.25 | 0.75 | 0.833 | 29.167 | 29.167 / 0.75 = 38.889 min |
| 0 | X8 | 1 | 0 | 0 | 0 | 87 | - |
| 0 | X9 | 0 | 1 | 0 | 0 | 67 | - |
| 0 | X10 | 0 | 0 | 1 | 0 | 110 | 110 / 1 = 110 |
| 0 | X11 | -1.083 | -1.25 | -0.75 | -0.833 | 15.833 | - |
|  | f | 13.583 | 14.75 | -3.75 | 35.833 | 1254.167 |  |
|  |  | Δ1 | Δ2 | Δ3 | Δ­­4 | Q |  |

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 ∆3. Наибольшая по модулю отрицательная оценка ∆3 = −3.75. В базис будет включена соответствующая ей небазисная переменная 𝑥3. 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 5.4.6 разрешающий столбец и разрешающая строка выделены. Разрешающим элементом является число 𝑎33 = 0.75.

Далее построим новую симплекс-таблицу. Ниже поэтапно демонстрируется процесс заполнения новой симплекс-таблицы (Таблицы 5.4.6).

Таблица 5.4.6 – Новая симплекс-таблица

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | сj | 33 | 39 | 43 | 0 |  |
|  |  | X1 | X2 | X4 | X7 |  |
| 0 | X5 |  |  |  |  |  |
| 0 | X6 |  |  |  |  |  |
| 36 | X3 |  |  | 1.33 |  |  |
| 0 | X8 |  |  |  |  |  |
| 0 | X9 |  |  |  |  |  |
| 0 | X10 |  |  |  |  |  |
| 0 | X11 |  |  |  |  |  |
|  | f |  |  |  |  |  |
|  |  | Δ1 | Δ2 | Δ3 | Δ­­4 | Q |

В Таблице 5.4.6 переменные 𝑥3 и 𝑥4 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 | 33 | 39 | 43 | 0 |  |
|  |  | X1 | X2 | X4 | X7 |  |
| 0 | X5 |  |  | 0.367 |  |  |
| 0 | X6 |  |  | -0.1 |  |  |
| 36 | X3 | 1.444 | 1.666 | 1.33 | 1.111 | 38.889 |
| 0 | X8 |  |  | 0 |  |  |
| 0 | X9 |  |  | 0 |  |  |
| 0 | X10 |  |  | -1.33 |  |  |
| 0 | X11 |  |  | 1 |  |  |
|  | f |  |  | 5 |  |  |
|  |  | Δ1 | Δ2 | Δ3 | Δ­­4 | Q |

Таблица 5.4.8 – Итерация 1

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | сj | 33 | 39 | 43 | 0 |  |  |
|  |  | X1 | X2 | X4 | X7 |  |  |
| 0 | X5 | -0.111 | -0.067 | 0.367 | -0.778 | 42.778 |  |
| 0 | X6 | -0.1 | 0 | -0.1 | -1 | 20 |  |
| 36 | X3 | 1.444 | 1.666 | 1.33 | 1.111 | 38.889 |  |
| 0 | X8 | 1 | 0 | 0 | 0 | 87 |  |
| 0 | X9 | 0 | 1 | 0 | 0 | 67 |  |
| 0 | X10 | -1.444 | -1.666 | -1.33 | -1.111 | 71.111 |  |
| 0 | X11 | 0 | 0 | 1 | 0 | 45 |  |
|  | f | 18.998 | 21 | 5 | 40 | 1400.004 |  |
|  |  | Δ1 | Δ2 | Δ3 | Δ­­4 | 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, фабрика должна выпускать в течении недели 𝑥3 = 38.889 шт. «офисных» компьютеров. Тогда фабрика получит максимальный доход от продажи 1400 [тыс. ден.ед].

## 5.5 Пример работы программы

Результаты выполнения программы, реализующей симплексный метод, представлены на Рисунках 5.5.1 – 5.5.4.



Рисунок 5.5.1 – Исходная симплекс-таблица

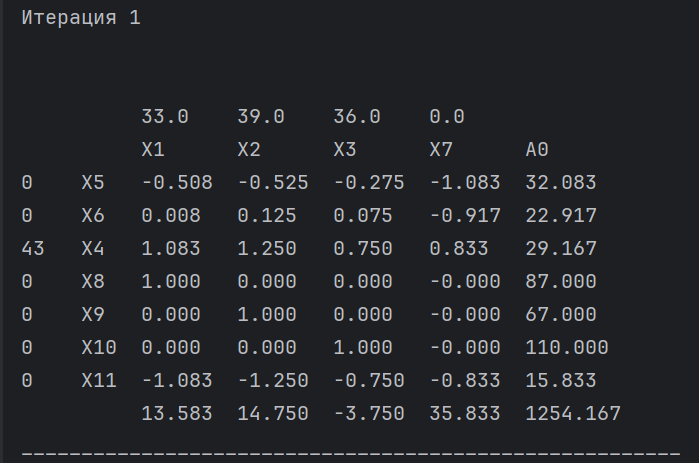


Рисунок 5.5.2 – Выполнение 1 итерации

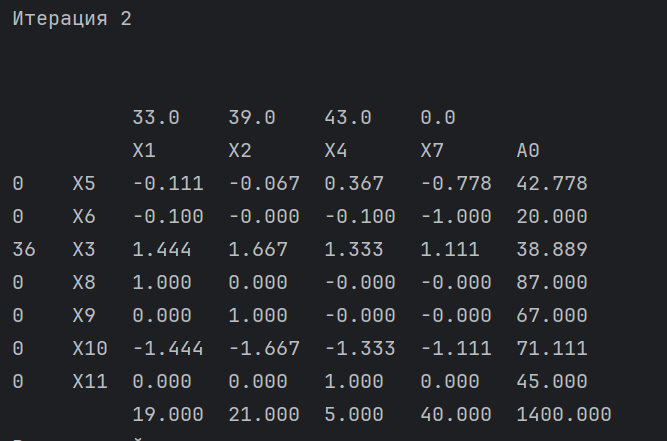


Рисунок 5.5.3 – Выполнение последней итерации

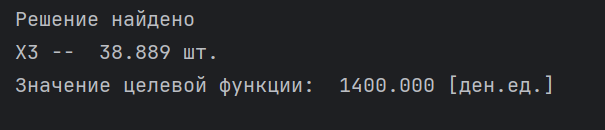


Рисунок 5.5.4 – Результат расчётов

## 5.6 Выводы по разделу

В ходе данной работы 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

Задача. Фирма поставляет компьютеры под ключ четырех базовых комплектаций: «домашний», «игровой», «офисный» и «экстрим». Известны средние затраты времени на сборку, проверку и подключение компьютеров. Каждый компьютер приносит определенный уровень прибыли, но спрос ограничен. Кроме того, в плановом периоде ограничен ресурс человеко-часов, отведенных на выполнение каждой производственной операции (Таблица 6.2.1).

Таблица 6.2.1. Исходные данные задачи.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Компьютер | Прибыль за модель  у.е. | Максимальный спрос на товар | Требуется часов на подключение | Требуется часов на сборку | Требуется часов на проверку |
| Домашний | 33 | 87 | 0,9 | 1,2 | 1,3 |
| Игровой | 39 | 67 | 1,1 | 1,5 | 1,5 |
| Офисный | 36 | 110 | 0,7 | 0,9 | 0,9 |
| Экстрим | 43 | 45 | 1,3 | 1,1 | 1,2 |
| Доступно чел.-час. на каждую  операцию | | | 70 | 55 | 35 |

Определить, сколько компьютеров каждого типа необходимо произвести в плановом периоде, имея целью максимизировать прибыль.

## 6.3 Математическая модель

Пусть х1 – количество домашних компьютеров, х2 – количество игровых компьютеров, х3 – количество офисных компьютеров, х4 – количество экстрим компьютеров. Прибыль от продажи компьютеров составит 33х1 + 39х2 + 36х3 + 43х4, прибыль требуется максимизировать.

Ограничения задачи:

## 6.4 Соответствующая исходной двойственная задача

Найдем соответствующую двойственную задачу. Введем вектор двойственных переменных размерности семь . 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 у.е., 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 𝑥5, 𝑥6, 𝑥3, 𝑥8, 𝑥9, 𝑥10, 𝑥11. 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.

Прибавим к 6-ей строке строчку 3, умноженную на (-10/9);

прибавим к 1-ой и 2-ой строчке строчку 3, умноженную на (-1) и (-7/9) соответственно;

делим каждую строку матрицы на ведущий элемент соответствующей строки;

Запишем обратную матрицу.

Базисными переменными в симплекс-таблице являются , тогда

При этом минимальное значение целевой функции двойственной задачи

совпадает с максимальным значением 𝑓𝑚𝑎𝑥 = 1400 [у. 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: объём производства компьютеров типа «офисный» – 𝑥3 = 38.889; объём производства остальных компьютеров равен нулю; максимальный доход от продажи 𝑓𝑚𝑎𝑥 = 1400 [у. ед.]. Рассмотрим выполнение неравенств прямой задачи при подстановке 𝑥1, 𝑥2, *х*3, 𝑥4 в систему ограничений (Таблица 6.6.1).

Таблица 6.6.1 – Выполнение неравенств прямой задачи

|  |  |  |
| --- | --- | --- |
| Ограничение | Расчет | Вывод |
| 0.9x1 + 1.1x2 + 0.7x3 + 1.3x4 ≤ 70 | 0.9\*0+1.1\*0+0.7\*38.889+1.3\*0 < 360  27.2223 < 70 | Первое ограничение прямой задачи выполняется как строгое неравенство, остается спрос на продукцию компьютеров типа «домашний». Значит, этот ресурс не является дефицитным и его оценка в оптимальном плане равна нулю (𝑦1 = 0). |
| 1.2x1 + 1.5x2 + 0.9x3 + 1.1x4 ≤ 55 | 1.2\*0+1.5\*0+0.9\*38.889+1.1\*0 < 55  35 < 55 | Второе ограничение прямой задачи выполняется как строгое неравенство, остается спрос на продукцию компьютеров типа «игровой». Значит, этот ресурс не является дефицитным и его оценка в оптимальном плане равна нулю (𝑦2 = 0). |
| 1.3x1 + 1.5x2 + 0.9x3 + 1.2x4 ≤ 35 | 1.3\*0+1.5\*0+0.9\*38.889+1.2\*0 = 35  35 = 35 | Третье ограничение прямой задачи выполняется как равенство. Это означает что компьютеры типа «офисный» полностью используется в оптимальном плане, является дефицитным и его оценка согласно второй теоремы двойственности отлична от нуля (𝑦3 ≠ 0). |

Продолжение Таблицы 6.6.1

|  |  |  |
| --- | --- | --- |
| x1 ≤ 87 | 0 < 87 | Четвёртое ограничение прямой задачи выполняется как строгое неравенство, остается спрос на продукцию компьютеров типа «экстрим». Значит, этот ресурс не является дефицитным и его оценка в оптимальном плане равна нулю (𝑦4 = 0). |
| x2 ≤ 67 | 0 < 67 | Пятое ограничение прямой задачи выполняется как строгое неравенство, значит, оценка пятого ресурса равна нулю (y5=0) |
| x3 ≤ 110 | 38.889 < 110 | Шестое ограничение прямой задачи выполняется как строгое неравенство, значит, оценка шестого ресурса равна нулю (y6=0) |
| x4 ≤ 45 | 0 < 45 | Пятое ограничение прямой задачи выполняется как строгое неравенство, значит, оценка седьмого ресурса равна нулю (y7=0) |
| х1 ≥ 0 | 0 = 0 | Первое ограничение в двойственной задаче будет равенством y1 = 0, т.е. в процессе производства не используется является не дефицитном. |
| х2 ≥ 0 | 0 = 0 | Второе ограничение в двойственной задаче будет равенством y2 = 0 |
| х3 ≥ 0 | 38.889 > 0 | Третье ограничение в двойственной задаче будет равенством 0.7y1 + 0.9y2 + 0.9y3 + y6 = 36, т.е. весь его запас полностью используется в оптимальном плане, он является дефицитным. |
| х4 ≥ 0 | 0 = 0 | Четвёртое ограничение в двойственной задаче будет равенством y4 = 0 |

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.

## 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 (Ограничение по часам на подключение)*. Найдем нижнюю границу. В первом столбце обратной матрицы один положительный элемент (1), ему соответствует индекс базисной переменной оптимального плана (70).

Найдем верхнюю границу. Т.к. в первом столбце нет отрицательных элементов, то:

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 (Ограничение по часам на сборку)*. Найдем нижнюю границу. Во втором столбце обратной матрицы один положительный элемент (1), ему соответствует индекс базисной переменной оптимального плана (55).

Найдем верхнюю границу. Т.к. во втором столбце нет отрицательных элементов, то:

Таким образом, получаем

Тогда второй ресурс может изменяться в интервале:

При таком значении оптимальный план двойственной задачи остается неизменным.

*Ресурс 3 (Ограничение по часам на проверку)*. Найдем нижнюю границу. В третьем столбце обратной матрицы один положительный элемент (10/9), ему соответствует индекс базисной переменной оптимального плана (35).

Найдем верхнюю границу. В третьем столбце три отрицательных элемента (-7/9, -1, -10/9), которым соответствуют индексы базисных переменных оптимального плана (70, 55, 110).

Наибольшее значение равно 99.

Таким образом, получаем

Тогда третий ресурс может изменяться в интервале:

При таком значении оптимальный план двойственной задачи остается неизменным.

*Ресурс 4 (Ограничение на «домашние» компьютеры)*. Найдем нижнюю границу. В четвёртом столбце обратной матрицы один положительный элемент (1), ему соответствует индекс базисной переменной оптимального плана (87).

Найдем верхнюю границу. Т.к. во четвёртом столбце нет отрицательных элементов, то:

Таким образом, получаем

Тогда четвёртый ресурс может изменяться в интервале:

При таком значении оптимальный план двойственной задачи остается неизменным.

*Ресурс 5 (Ограничение на «игровые» компьютеры)*. Найдем нижнюю границу. В пятом столбце обратной матрицы один положительный элемент (1), ему соответствует индекс базисной переменной оптимального плана (67).

Найдем верхнюю границу. Т.к. во пятом столбце нет отрицательных элементов, то:

Таким образом, получаем

Тогда пятый ресурс может изменяться в интервале:

При таком значении оптимальный план двойственной задачи остается неизменным.

*Ресурс 6 (Ограничение на «офисные» компьютеры)*. Найдем нижнюю границу. В шестом столбце обратной матрицы один положительный элемент (1), ему соответствует индекс базисной переменной оптимального плана (110).

Найдем верхнюю границу. Т.к. во втором столбце нет отрицательных элементов, то:

Таким образом, получаем

Тогда шестой ресурс может изменяться в интервале:

При таком значении оптимальный план двойственной задачи остается неизменным.

*Ресурс 7 (Ограничение на «экстрим» компьютеры)*. Найдем нижнюю границу. В седьмом столбце обратной матрицы один положительный элемент (1), ему соответствует индекс базисной переменной оптимального плана (45).

Найдем верхнюю границу. Т.к. в седьмом столбце нет отрицательных элементов, то:

Таким образом, получаем

Тогда седьмой ресурс может изменяться в интервале:

При таком значении оптимальный план двойственной задачи остается неизменным.

Далее оценим влияние изменения объема ресурсов на величину максимальной стоимости продукции. Как известно, это дефицитные ресурсы Введем верхнюю границу 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 Рисунках 6.8.1 – 6.8.3.

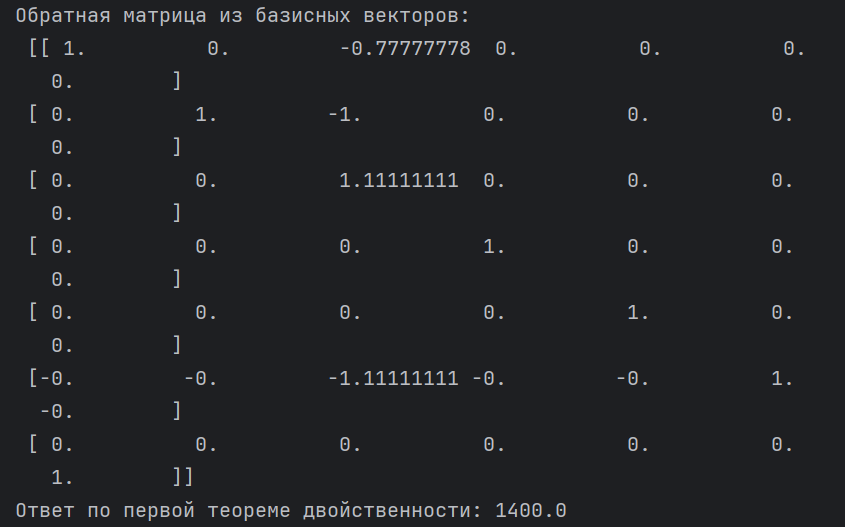


Рисунок 6.8.1 – Первая теорема двойственности

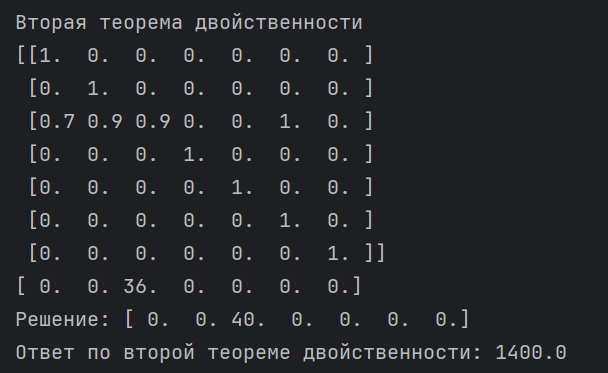


Рисунок 6.8.2 – Вторая теорема двойственности

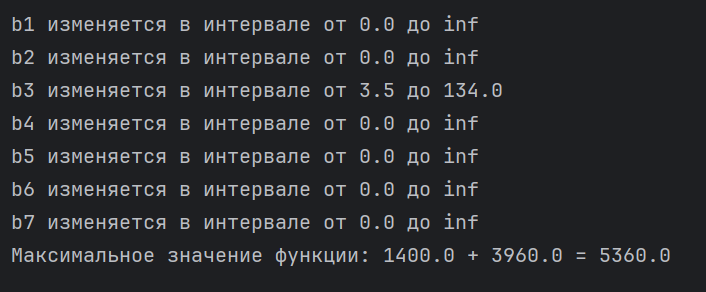


Рисунок 6.8.3 – Третья теорема двойственности

## 6.9 Выводы по разделу

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

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

Таблица 7.2.1. Исходные данные задачи.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Потребители  Поставщики | 8 | 9 | 13 | 8 | 12 |
| 19 | 5 | 15 | 3 | 6 | 10 |
| 17 | 23 | 8 | 13 | 27 | 12 |
| 14 | 30 | 1 | 5 | 24 | 25 |

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 (8 < 19), то . Запасы первого поставщика не исчерпаны, поэтому происходит переход к ячейке . Т.к. потребности меньше запасов (9 < 11), то Запасы первого поставщика не исчерпаны, поэтому происходит переход к ячейке . Т.к. потребности больше запасов (13 > 2), то . Запасы третьего потребителя не удовлетворены, поэтому происходит переход к ячейке . Т.к. потребности меньше запасов (11 < 17), то . Запасы второго поставщика не исчерпаны, поэтому происходит переход к ячейке . Т.к. потребности больше запасов (8 > 6), то . Запасы четвёртого потребителя не удовлетворены, поэтому происходит переход к ячейке . Т.к. потребности меньше запасов (2 < 14), то . Запасы третьего поставщика не исчерпаны, поэтому происходит переход к ячейке . Потребности равны запасам (12=12), значит . Все поставки распределены, количество базисных клеток равно 7 = m + n – 1, значит, план невырожденный (Таблица 7.4.1).

Таблица 7.4.1 – Метод северо-западного угла

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Пункты | B1 | B2 | B3 | B4 | B5 | Запасы |
| A1 | |  |  | | --- | --- | |  | 5 | | 8 |  | | |  |  | | --- | --- | |  | 15 | | 9 |  | | |  |  | | --- | --- | |  | 3 | | 2 |  | | |  |  | | --- | --- | |  | 6 | |  |  | | |  |  | | --- | --- | |  | 10 | |  |  | | 19 |
| A2 | |  |  | | --- | --- | |  | 23 | |  |  | | |  |  | | --- | --- | |  | 8 | |  |  | | |  |  | | --- | --- | |  | 13 | | 11 |  | | |  |  | | --- | --- | |  | 27 | | 6 |  | | |  |  | | --- | --- | |  | 12 | |  |  | | 17 |
| A3 | |  |  | | --- | --- | |  | 30 | |  |  | | |  |  | | --- | --- | |  | 1 | |  |  | | |  |  | | --- | --- | |  | 5 | |  |  | | |  |  | | --- | --- | |  | 24 | | 2 |  | | |  |  | | --- | --- | |  | 25 | | 12 |  | | 14 |
| Потребности | 8 | 9 | 13 | 8 | 12 | 50 |

## 7.5 Метод минимальной стоимости

Выбираем ячейку с минимальной стоимостью . Значение определяется как минимальное из остатков запасов и остатков запасов . Тогда . Запасы второго потребителя исчерпаны, значит, вычёркиваем второй столбец. Из оставшихся ячеек выбираем ячейку с минимальной стоимостью . Значение определяется как минимальное из остатков запасов и остатков запасов . Тогда . Запасы третьего потребителя исчерпаны, значит, вычёркиваем третий столбец. Из оставшихся ячеек выбираем ячейку с минимальной стоимостью . Значение определяется как минимальное из остатков запасов и остатков запасов . Тогда . Запасы первого поставщика исчерпаны, значит, вычёркиваем первую строку. Из оставшихся ячеек выбираем ячейку с минимальной стоимостью . Значение определяется как минимальное из остатков запасов и остатков запасов . Тогда . Запасы пятого потребителя исчерпаны, значит, вычёркиваем пятый столбец. Из оставшихся ячеек выбираем ячейку с минимальной стоимостью . Значение определяется как минимальное из остатков запасов и остатков запасов . Тогда . Запасы первого потребителя исчерпаны, значит, вычёркиваем первый столбец. Из оставшихся ячеек выбираем ячейку с минимальной стоимостью . Значение определяется как минимальное из остатков запасов и остатков запасов . Тогда . Запасы третьего поставщика исчерпаны, значит, вычёркиваем третью строку. Осталась ячейка , в которую записываем 3.

Получившиеся расчёты приведены в Таблице 7.5.1.

Таблица 7.5.1 – Метод минимальной стоимости

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Пункты | B1 | B2 | B3 | B4 | B5 | Запасы |
| A1 | |  |  | | --- | --- | |  | 5 | | 6 |  | | |  |  | | --- | --- | |  | 15 | |  |  | | |  |  | | --- | --- | |  | 3 | | 13 |  | | |  |  | | --- | --- | |  | 6 | |  |  | | |  |  | | --- | --- | |  | 10 | |  |  | | 19 |
| A2 | |  |  | | --- | --- | |  | 23 | | 2 |  | | |  |  | | --- | --- | |  | 8 | |  |  | | |  |  | | --- | --- | |  | 13 | |  |  | | |  |  | | --- | --- | |  | 27 | | 3 |  | | |  |  | | --- | --- | |  | 12 | | 12 |  | | 17 |
| A3 | |  |  | | --- | --- | |  | 30 | |  |  | | |  |  | | --- | --- | |  | 1 | | 9 |  | | |  |  | | --- | --- | |  | 5 | |  |  | | |  |  | | --- | --- | |  | 24 | | 5 |  | | |  |  | | --- | --- | |  | 25 | |  |  | | 14 |
| Потребности | 8 | 9 | 13 | 8 | 12 | 50 |

Расчёт стоимости: 5\*6 + 23\*2 + 1\*9 + 3\*13 + 27\*3 + 24\*5 + 12\*12 = 469

## 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 + 5 – 1 = 7

Таблица 7.6.1 – Метод северо-западного угла

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Пункты | B1 | B2 | B3 | B4 | B5 | Запасы |
| A1 | |  |  | | --- | --- | |  | 5 | | 8 |  | | |  |  | | --- | --- | |  | 15 | | 9 |  | | |  |  | | --- | --- | |  | 3 | | 2 |  | | |  |  | | --- | --- | |  | 6 | |  |  | | |  |  | | --- | --- | |  | 10 | |  |  | | 19 |
| A2 | |  |  | | --- | --- | |  | 23 | |  |  | | |  |  | | --- | --- | |  | 8 | |  |  | | |  |  | | --- | --- | |  | 13 | | 11 |  | | |  |  | | --- | --- | |  | 27 | | 6 |  | | |  |  | | --- | --- | |  | 12 | |  |  | | 17 |
| A3 | |  |  | | --- | --- | |  | 30 | |  |  | | |  |  | | --- | --- | |  | 1 | |  |  | | |  |  | | --- | --- | |  | 5 | |  |  | | |  |  | | --- | --- | |  | 24 | | 2 |  | | |  |  | | --- | --- | |  | 25 | | 12 |  | | 14 |
| Потребности | 8 | 9 | 13 | 8 | 12 | 50 |

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 (3,2).

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=5 | v2=15 | v3=3 | v4=17 | v5=18 |  |
|  | Пункты | B1 | B2 | B3 | B4 | B5 | Запасы |
| u1=0 | A1 | |  |  | | --- | --- | |  | 5 | | 8 |  | | |  |  | | --- | --- | | - | 15 | | 9 |  | | |  |  | | --- | --- | | + | 3 | | 2 |  | | |  |  | | --- | --- | |  | 6 | |  |  | | |  |  | | --- | --- | |  | 10 | |  |  | | 19 |
| u2=10 | A2 | |  |  | | --- | --- | |  | 23 | |  |  | | |  |  | | --- | --- | |  | 8 | |  |  | | |  |  | | --- | --- | | - | 13 | | 11 |  | | |  |  | | --- | --- | | + | 27 | | 6 |  | | |  |  | | --- | --- | |  | 12 | |  |  | | 17 |
| u3=7 | A3 | |  |  | | --- | --- | |  | 30 | |  |  | | |  |  | | --- | --- | | λ | 1 | | + |  | | |  |  | | --- | --- | |  | 5 | |  |  | | |  |  | | --- | --- | | - | 24 | | 2 |  | | |  |  | | --- | --- | |  | 25 | | 12 |  | | 14 |
|  | Потребности | 8 | 9 | 13 | 8 | 12 | 50 |

Таблица 7.6.3 – Таблица нового плана

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | v1=5 | v2=15 | v3=3 | v4=17 | v5=18 |  |
|  | Пункты | B1 | B2 | B3 | B4 | B5 | Запасы |
| u1=0 | A1 | |  |  | | --- | --- | |  | 5 | | 8 |  | | |  |  | | --- | --- | |  | 15 | | 7 |  | | |  |  | | --- | --- | |  | 3 | | 4 |  | | |  |  | | --- | --- | |  | 6 | |  |  | | |  |  | | --- | --- | |  | 10 | |  |  | | 19 |
| u2=10 | A2 | |  |  | | --- | --- | |  | 23 | |  |  | | |  |  | | --- | --- | |  | 8 | |  |  | | |  |  | | --- | --- | |  | 13 | | 9 |  | | |  |  | | --- | --- | |  | 27 | | 8 |  | | |  |  | | --- | --- | |  | 12 | |  |  | | 17 |
| u3=7 | A3 | |  |  | | --- | --- | |  | 30 | |  |  | | |  |  | | --- | --- | |  | 1 | | 2 |  | | |  |  | | --- | --- | |  | 5 | |  |  | | |  |  | | --- | --- | |  | 24 | |  |  | | |  |  | | --- | --- | |  | 25 | | 12 |  | | 14 |
|  | Потребности | 8 | 9 | 13 | 8 | 12 | 50 |

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 (2,5).

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=5 | v2=15 | v3=3 | v4=17 | v5=39 |  |
|  | Пункты | B1 | B2 | B3 | B4 | B5 | Запасы |
| u1=0 | A1 | |  |  | | --- | --- | |  | 5 | | 8 |  | | |  |  | | --- | --- | | - | 15 | | 7 |  | | |  |  | | --- | --- | | + | 3 | | 4 |  | | |  |  | | --- | --- | |  | 6 | |  |  | | |  |  | | --- | --- | |  | 10 | |  |  | | 19 |
| u2=10 | A2 | |  |  | | --- | --- | |  | 23 | |  |  | | |  |  | | --- | --- | |  | 8 | |  |  | | |  |  | | --- | --- | | - | 13 | | 9 |  | | |  |  | | --- | --- | |  | 27 | | 8 |  | | |  |  | | --- | --- | | λ | 12 | | + |  | | 17 |
| u3=-14 | A3 | |  |  | | --- | --- | |  | 30 | |  |  | | |  |  | | --- | --- | | + | 1 | | 2 |  | | |  |  | | --- | --- | |  | 5 | |  |  | | |  |  | | --- | --- | |  | 24 | |  |  | | |  |  | | --- | --- | | - | 25 | | 12 |  | | 14 |
|  | Потребности | 8 | 9 | 13 | 8 | 12 | 50 |

Таблица 7.6.5 – Таблица нового плана

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | v1=5 | v2=15 | v3=3 | v4=17 | v5=18 |  |
|  | Пункты | B1 | B2 | B3 | B4 | B5 | Запасы |
| u1=0 | A1 | |  |  | | --- | --- | |  | 5 | | 8 |  | | |  |  | | --- | --- | |  | 15 | |  |  | | |  |  | | --- | --- | |  | 3 | | 11 |  | | |  |  | | --- | --- | |  | 6 | |  |  | | |  |  | | --- | --- | |  | 10 | |  |  | | 19 |
| u2=10 | A2 | |  |  | | --- | --- | |  | 23 | |  |  | | |  |  | | --- | --- | |  | 8 | |  |  | | |  |  | | --- | --- | |  | 13 | | 2 |  | | |  |  | | --- | --- | |  | 27 | | 8 |  | | |  |  | | --- | --- | |  | 12 | | 7 |  | | 17 |
| u3=7 | A3 | |  |  | | --- | --- | |  | 30 | |  |  | | |  |  | | --- | --- | |  | 1 | | 9 |  | | |  |  | | --- | --- | |  | 5 | |  |  | | |  |  | | --- | --- | |  | 24 | |  |  | | |  |  | | --- | --- | |  | 25 | | 5 |  | | 14 |
|  | Потребности | 8 | 9 | 13 | 8 | 12 | 50 |

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 (3,3).

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=5 | v2=-22 | v3=3 | v4=17 | v5=2 |  |
|  | Пункты | B1 | B2 | B3 | B4 | B5 | Запасы |
| u1=0 | A1 | |  |  | | --- | --- | |  | 5 | | 8 |  | | |  |  | | --- | --- | |  | 15 | |  |  | | |  |  | | --- | --- | |  | 3 | | 11 |  | | |  |  | | --- | --- | |  | 6 | |  |  | | |  |  | | --- | --- | |  | 10 | |  |  | | 19 |
| u2=10 | A2 | |  |  | | --- | --- | |  | 23 | |  |  | | |  |  | | --- | --- | |  | 8 | |  |  | | |  |  | | --- | --- | | - | 13 | | 2 |  | | |  |  | | --- | --- | |  | 27 | | 8 |  | | |  |  | | --- | --- | | + | 12 | | 7 |  | | 17 |
| u3=23 | A3 | |  |  | | --- | --- | |  | 30 | |  |  | | |  |  | | --- | --- | |  | 1 | | 9 |  | | |  |  | | --- | --- | | λ | 5 | | + |  | | |  |  | | --- | --- | |  | 24 | |  |  | | |  |  | | --- | --- | | - | 25 | | 5 |  | | 14 |
|  | Потребности | 8 | 9 | 13 | 8 | 12 | 50 |

Таблица 7.6.7 – Таблица нового плана

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | v1=5 | v2=15 | v3=3 | v4=17 | v5=18 |  |
|  | Пункты | B1 | B2 | B3 | B4 | B5 | Запасы |
| u1=0 | A1 | |  |  | | --- | --- | |  | 5 | | 8 |  | | |  |  | | --- | --- | |  | 15 | |  |  | | |  |  | | --- | --- | |  | 3 | | 11 |  | | |  |  | | --- | --- | |  | 6 | |  |  | | |  |  | | --- | --- | |  | 10 | |  |  | | 19 |
| u2=10 | A2 | |  |  | | --- | --- | |  | 23 | |  |  | | |  |  | | --- | --- | |  | 8 | |  |  | | |  |  | | --- | --- | |  | 13 | |  |  | | |  |  | | --- | --- | |  | 27 | | 8 |  | | |  |  | | --- | --- | |  | 12 | | 9 |  | | 17 |
| u3=7 | A3 | |  |  | | --- | --- | |  | 30 | |  |  | | |  |  | | --- | --- | |  | 1 | | 9 |  | | |  |  | | --- | --- | |  | 5 | | 2 |  | | |  |  | | --- | --- | |  | 24 | |  |  | | |  |  | | --- | --- | |  | 25 | | 3 |  | | 14 |
|  | Потребности | 8 | 9 | 13 | 8 | 12 | 50 |

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 7.6.8). 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 7.6.9.

Таблица 7.6.8 – Таблица перерасчёта

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | v1=5 | v2=-1 | v3=3 | v4=38 | v5=23 |  |
|  | Пункты | B1 | B2 | B3 | B4 | B5 | Запасы |
| u1=0 | A1 | |  |  | | --- | --- | |  | 5 | | 8 |  | | |  |  | | --- | --- | |  | 15 | |  |  | | |  |  | | --- | --- | | - | 3 | | 11 |  | | |  |  | | --- | --- | | λ | 6 | | + |  | | |  |  | | --- | --- | |  | 10 | |  |  | | 19 |
| u2=-11 | A2 | |  |  | | --- | --- | |  | 23 | |  |  | | |  |  | | --- | --- | |  | 8 | |  |  | | |  |  | | --- | --- | |  | 13 | |  |  | | |  |  | | --- | --- | | - | 27 | | 8 |  | | |  |  | | --- | --- | | + | 12 | | 9 |  | | 17 |
| u3=2 | A3 | |  |  | | --- | --- | |  | 30 | |  |  | | |  |  | | --- | --- | |  | 1 | | 9 |  | | |  |  | | --- | --- | | + | 5 | | 2 |  | | |  |  | | --- | --- | |  | 24 | |  |  | | |  |  | | --- | --- | | - | 25 | | 3 |  | | 14 |
|  | Потребности | 8 | 9 | 13 | 8 | 12 | 50 |

Таблица 7.6.9 – Таблица нового плана

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | v1=5 | v2=-1 | v3=3 | v4=38 | v5=23 |  |
|  | Пункты | B1 | B2 | B3 | B4 | B5 | Запасы |
| u1=0 | A1 | |  |  | | --- | --- | |  | 5 | | 8 |  | | |  |  | | --- | --- | |  | 15 | |  |  | | |  |  | | --- | --- | |  | 3 | | 8 |  | | |  |  | | --- | --- | |  | 6 | | 3 |  | | |  |  | | --- | --- | |  | 10 | |  |  | | 19 |

Продолжение Таблицы 7.6.9

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| u2=-11 | A2 | |  |  | | --- | --- | |  | 23 | |  |  | | |  |  | | --- | --- | |  | 8 | |  |  | | |  |  | | --- | --- | |  | 13 | |  |  | | |  |  | | --- | --- | |  | 27 | | 5 |  | | |  |  | | --- | --- | |  | 12 | | 12 |  | | 17 |
| u3=2 | A3 | |  |  | | --- | --- | |  | 30 | |  |  | | |  |  | | --- | --- | |  | 1 | | 9 |  | | |  |  | | --- | --- | |  | 5 | | 5 |  | | |  |  | | --- | --- | |  | 24 | |  |  | | |  |  | | --- | --- | |  | 25 | |  |  | | 14 |
|  | Потребности | 8 | 9 | 13 | 8 | 12 | 50 |

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 (2,2).

Для определения количества груза подлежащего распределению, построим замкнутый цикл (указан стрелками) (Таблица 7.6.10). 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 7.6.11.

Таблица 7.6.10 – Таблица перерасчёта

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | v1=5 | v2=-1 | v3=3 | v4=6 | v5=-9 |  |
|  | Пункты | B1 | B2 | B3 | B4 | B5 | Запасы |
| u1=0 | A1 | |  |  | | --- | --- | |  | 5 | | 8 |  | | |  |  | | --- | --- | |  | 15 | |  |  | | |  |  | | --- | --- | | - | 3 | | 8 |  | | |  |  | | --- | --- | | + | 6 | | 3 |  | | |  |  | | --- | --- | |  | 10 | |  |  | | 19 |
| u2=21 | A2 | |  |  | | --- | --- | |  | 23 | |  |  | | |  |  | | --- | --- | | λ | 8 | | + |  | | |  |  | | --- | --- | |  | 13 | |  |  | | |  |  | | --- | --- | | - | 27 | | 5 |  | | |  |  | | --- | --- | |  | 12 | | 12 |  | | 17 |
| u3=2 | A3 | |  |  | | --- | --- | |  | 30 | |  |  | | |  |  | | --- | --- | | - | 1 | | 9 |  | | |  |  | | --- | --- | | + | 5 | | 5 |  | | |  |  | | --- | --- | |  | 24 | |  |  | | |  |  | | --- | --- | |  | 25 | |  |  | | 14 |
|  | Потребности | 8 | 9 | 13 | 8 | 12 | 50 |

Таблица 7.6.11 – Таблица нового плана

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | v1=5 | v2=-1 | v3=3 | v4=6 | v5=-9 |  |
|  | Пункты | B1 | B2 | B3 | B4 | B5 | Запасы |
| u1=0 | A1 | |  |  | | --- | --- | |  | 5 | | 8 |  | | |  |  | | --- | --- | |  | 15 | |  |  | | |  |  | | --- | --- | |  | 3 | | 3 |  | | |  |  | | --- | --- | |  | 6 | | 8 |  | | |  |  | | --- | --- | |  | 10 | |  |  | | 19 |
| u2=21 | A2 | |  |  | | --- | --- | |  | 23 | |  |  | | |  |  | | --- | --- | |  | 8 | | 5 |  | | |  |  | | --- | --- | |  | 13 | |  |  | | |  |  | | --- | --- | |  | 27 | |  |  | | |  |  | | --- | --- | |  | 12 | | 12 |  | | 17 |
| u3=2 | A3 | |  |  | | --- | --- | |  | 30 | |  |  | | |  |  | | --- | --- | |  | 1 | | 4 |  | | |  |  | | --- | --- | |  | 5 | | 10 |  | | |  |  | | --- | --- | |  | 24 | |  |  | | |  |  | | --- | --- | |  | 25 | |  |  | | 14 |
|  | Потребности | 8 | 9 | 13 | 8 | 12 | 50 |

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

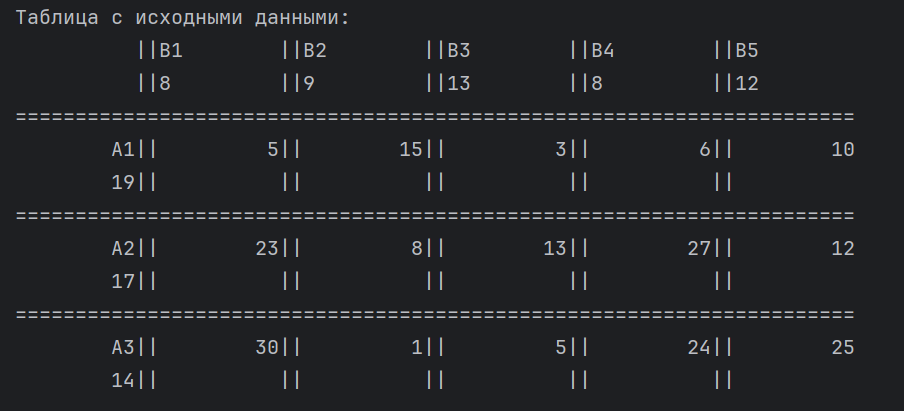


Рисунок 7.7.1 – Изначальная таблица



Рисунок 7.7.2 – Метод минимальной стоимости

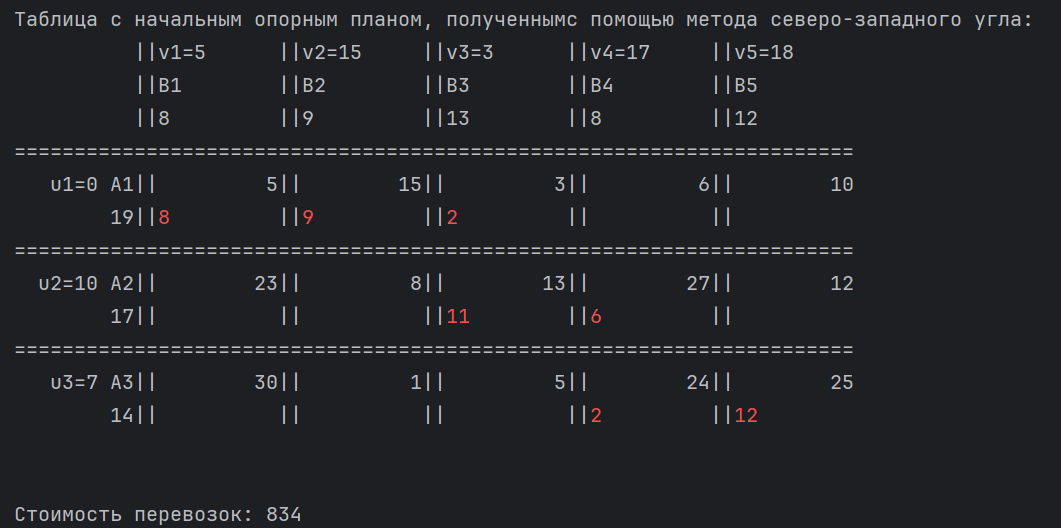


Рисунок 7.7.3 – Метод северо-западного угла

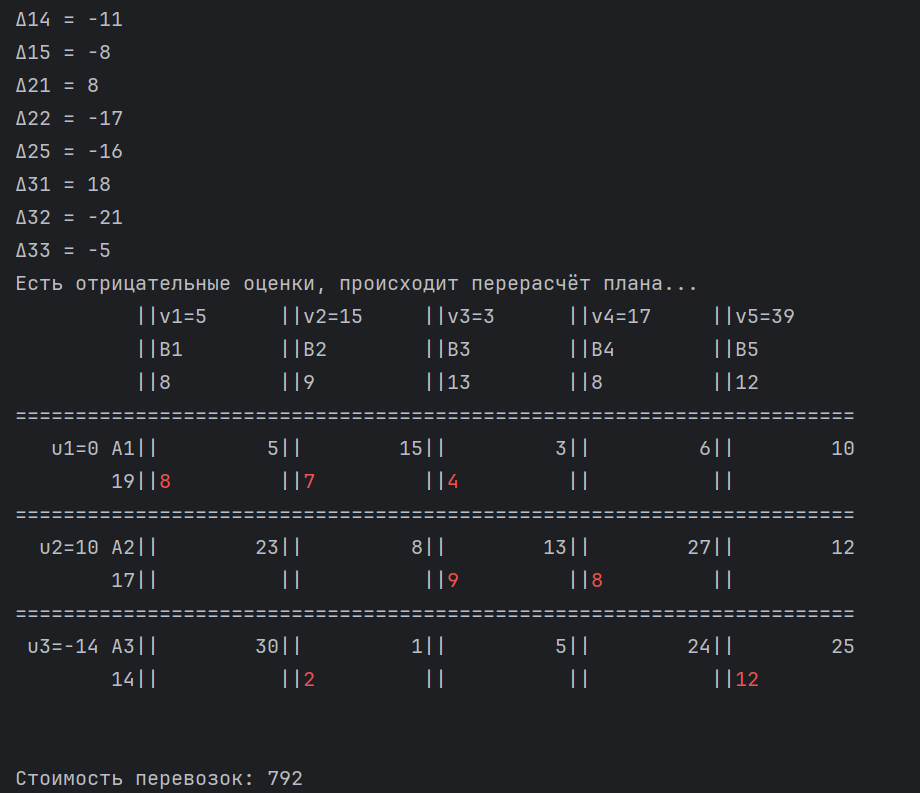


Рисунок 7.7.4 – Первая итерация метода потенциалов

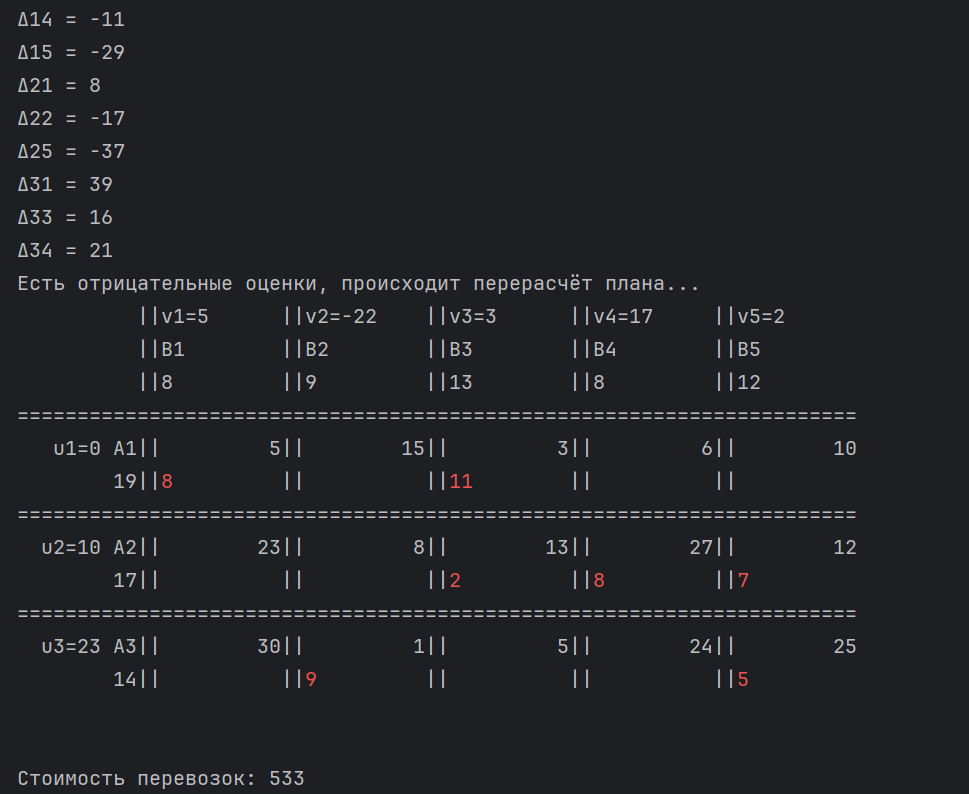


Рисунок 7.7.5 – Вторая итерация метода потенциалов

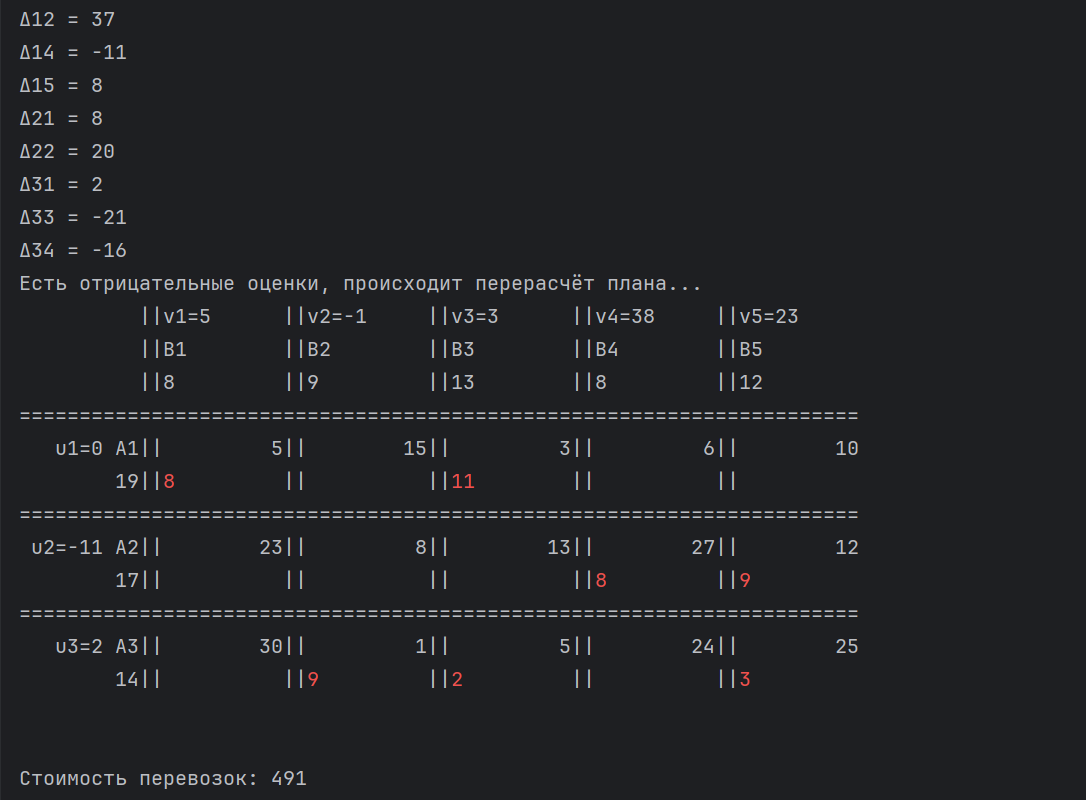


Рисунок 7.7.6 – Третья итерация метода потенциалов

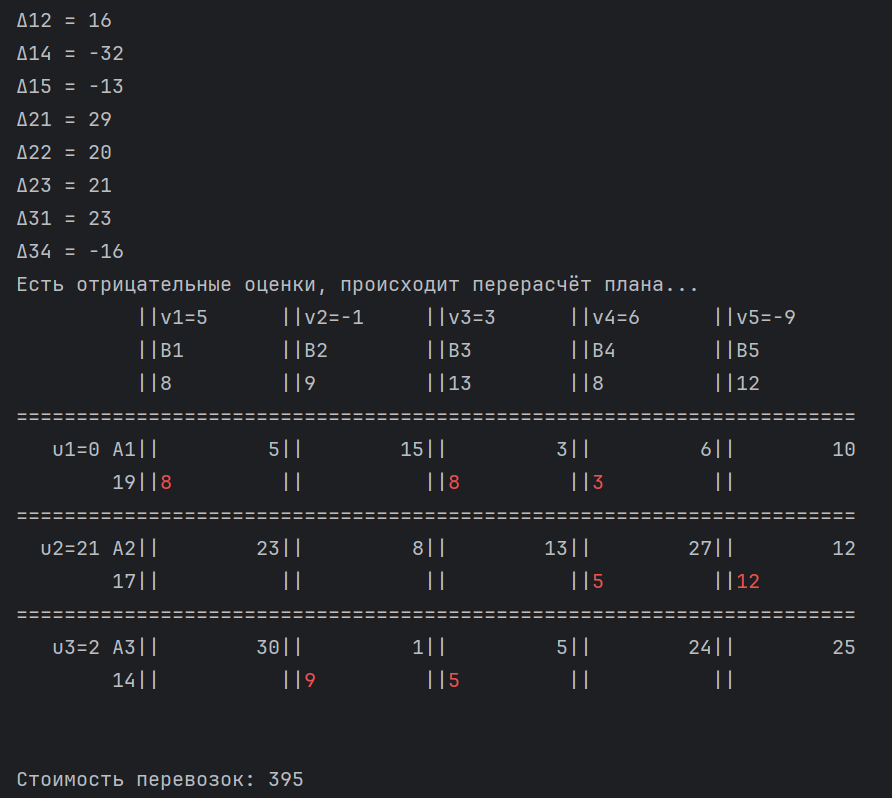


Рисунок 7.7.7 – Четвёртая итерация метода потенциалов

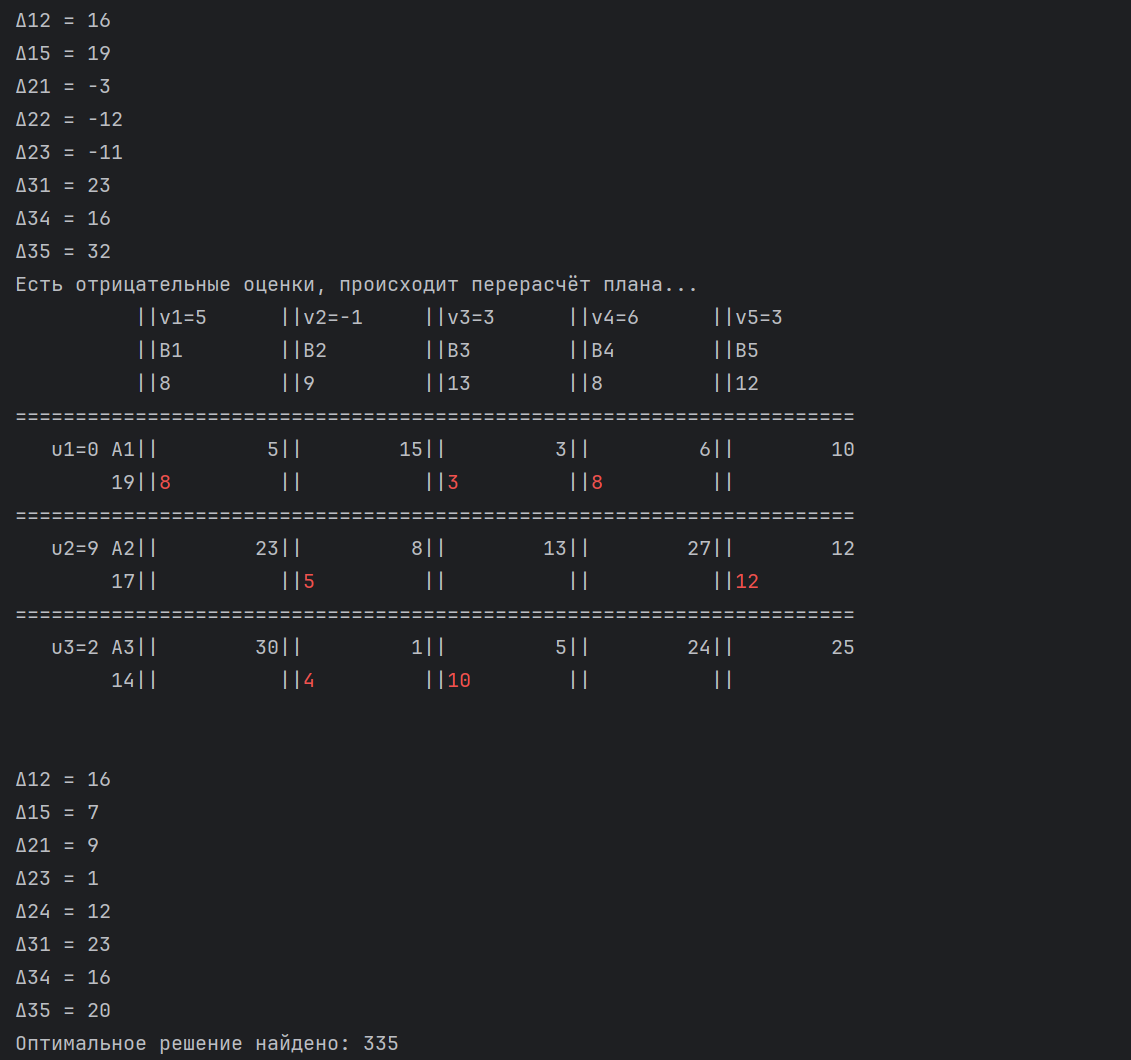


Рисунок 7.7.8 – Последняя итерация метода потенциалов

## 7.8 Выводы по разделу

В ходе выполнения данной работы 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.

# 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 графического метода на языке Python.

Приложение Д – Код реализации симплексного метода на языке Python.

Приложение Е – Код реализации двойственной задачи на языке Python.

Приложение Ж – Код реализации транспортной задачи на языке Python.

### Приложение А

Код реализации метода Парето на языке Python

Листинг А.1 — Реализация метода Парето

import csv  
  
  
def lexico\_graphic(table, criteria\_order):  
 winners = table  
 for key in criteria\_order:  
 winners = sorted(winners, key=lambda x: table[x][key], reverse=True)  
 cnt = 0  
 for w in winners:  
 if table[w][key] == table[winners[0]][key]:  
 cnt += 1  
 else:  
 break  
 if cnt == 1:  
 break  
 else:  
 winners = {w: table[w] for w in winners[:cnt]}  
  
 print(f'{winners[0]}: {table[winners[0]]}')  
  
  
def suboptimization(table, main\_criteria, borders):  
 winners = {}  
 for i, (key, value) in enumerate(table.items()):  
 flag = True  
 for criteria in borders.keys():  
 if criteria in ['Длина', 'Агрессивность']:  
 if value[criteria] > borders[criteria]:  
 flag = False  
 else:  
 if value[criteria] < borders[criteria]:  
 flag = False  
  
 if flag:  
 winners.update({key: value})  
  
 winner = max(winners, key=lambda k: winners[k][main\_criteria])  
 print(f'{winner}: {winners[winner]}')  
  
  
def criteria\_borders(table, borders):  
 winners = {}  
 for i, (key, value) in enumerate(table.items()):  
 flag = True  
 for criteria in borders.keys():  
 if criteria in ['Длина', 'Агрессивность']:  
 if value[criteria] > borders[criteria]:  
 flag = False  
 else:  
 if value[criteria] < borders[criteria]:  
 flag = False  
  
 if flag:  
 winners.update({key: value})  
  
 pareto\_optim(winners)

Продолжение Листинга А.1

def pareto\_optim(table):  
 losers = []  
 for key, value in table.items():  
 for key1, value1 in table.items():  
 if key == key1:  
 continue  
 if value['Популярность'] >= value1['Популярность'] and \

value['Рейтинг'] >= value1['Рейтинг'] and \

value['Темп'] >= value1['Темп'] and \

value['Длина'] <= value1['Длина'] and \

value['Агрессивность'] <= value1['Агрессивность'] and \

value['Громкость'] >= value1['Громкость'] and \

not all(value[k] == value1[k] for k in value1.keys()):  
 losers.append(value1)  
  
 winners = {}  
 for i, (key, value) in enumerate(table.items()):  
 if value not in losers:  
 winners.update({key: value})  
 print('\n'.join(f'{key}: {value}' for key, value in winners.items()))  
  
  
def main():  
 with open('table.csv', 'r', encoding='utf-8') as file:  
 array = []  
 reader = csv.reader(file, delimiter=',')  
 for row in reader:  
 array.append(row)  
  
 table = dict.fromkeys(elem[0] for elem in array[1:])  
 for i in range(1, len(array)):  
 values = [int(elem) if ':' not in elem else elem for elem in array[i][1:]]  
 table[array[i][0]] = dict.fromkeys(elem for elem in array[0][1:])  
 table[array[i][0]]['Популярность'] = values[0]  
 table[array[i][0]]['Рейтинг'] = values[1]  
 table[array[i][0]]['Темп'] = values[2]  
 table[array[i][0]]['Длина'] = values[3]  
 table[array[i][0]]['Агрессивность'] = values[4]  
 table[array[i][0]]['Громкость'] = values[5]  
  
 print('Парето-оптимальное множество:')  
 print('=============================')  
 pareto\_optim(table)  
 print('=============================\n')  
  
 print('Метод установки границ для критериев:')  
 print('==============================')  
 borders = {'Популярность': 1500,  
 'Рейтинг': 90,  
 'Длина': '4:00'}  
 criteria\_borders(table, borders)  
 print('==============================\n')  
  
 print('Метод субоптимизации:')  
 print('=====================')  
 borders = {'Рейтинг': 90,  
 'Длина': '4:00'}  
 main\_criteria = 'Популярность'

Продолжение Листинга А.1

suboptimization(table, main\_criteria, borders)

print('================\n')

print('Лексикографическая оптимизация:')  
 print('===============================')  
 criteria\_order = ['Рейтинг', 'Популярность', 'Длина',  
 'Темп', 'Агрессивность', 'Громкость']  
 lexico\_graphic(table, criteria\_order)  
 print('==============================\n')  
  
  
main()

### Приложение Б

Код реализации метода Электра II на языке Python.

Листинг Б.1. Реализация метода Электра II.

import csv

import graphviz

border = 2

def print\_matrix(matrix):

print(10 \* ' ' + '||', end='')

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

print(f'{i}'.rjust(10), end='')

print('\n' + '=' \* (10 \* (len(matrix) + 1) + 2))

for i in range(len(matrix)):

print(f'{i + 1}'.rjust(10), end='||')

for j in range(len(matrix[i])):

if matrix[i][j] == 1e6:

print('\u022e'.rjust(10), end='')

elif matrix[i][j] == 0:

print('-'.rjust(10), end='')

else:

print(f'{matrix[i][j]}'.rjust(10), end='')

print()

def get\_levels(matrix):

levels = []

visited = []

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] >= border:

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

if not flag:

break

if not flag:

continue

else:

level.append(i)

visited.append(i)

levels.append(level)

print(f'{len(levels)}-й уровень: ' + ', '.join(map(lambda x: str(x + 1), level)))

return levels

def csv\_write(matrix):

with open('result.csv', 'w', encoding='utf-8', newline='') as file:

writer = csv.writer(file, delimiter=',')

writer.writerows([[

elem if 0 < elem < 1e6 else '-' if elem == 0 else u'\u221E'

for elem in row] for row in matrix])

with open('result1.csv', 'w', encoding='utf-8', newline='') as file:

Продолжение Листинга Б.1.

writer = csv.writer(file, delimiter=',')

writer.writerows([[

elem if border <= elem < 1e6 else '-' if elem <= border else u'\u221E'

for elem in row] for row in matrix])

def main():

with open('table.csv', 'r', encoding='utf-8') as file:

array = []

reader = csv.reader(file, delimiter=',')

for row in reader:

array.append(row)

table = dict.fromkeys(elem[0] for elem in array[1:])

for i in range(1, len(array)):

values = [int(elem) if ':' not in elem else elem for elem in

array[i][1:]]

table[array[i][0]] = dict.fromkeys(elem for elem in array[0][1:])

table[array[i][0]]['Популярность'] = values[0]

table[array[i][0]]['Рейтинг'] = values[1]

table[array[i][0]]['Темп'] = values[2]

table[array[i][0]]['Длина'] = values[3]

table[array[i][0]]['Агрессивность'] = values[4]

table[array[i][0]]['Громкость'] = values[5]

weights = {'Популярность': 4,

'Рейтинг': 6,

'Темп': 2,

'Длина': 2,

'Агрессивность': 1,

'Громкость': 3}

marks = {'Популярность': lambda

x: 15 if x >= 2500 else 10 if 1500 <= x < 2500 else 5,

'Рейтинг': lambda x: 15 if x >= 95 else 10 if 80 <= x < 95 else 5,

'Темп': lambda x: 10 if x >= 120 else 5,

'Длина': lambda x: 5 if x >= '4:00' else 10,

'Агрессивность': lambda

x: 5 if x >= 7 else 10 if 4 <= x < 7 else 15,

'Громкость': lambda x: 10 if x > -6 else 5}

matrix = [[0 for \_ in range(len(table))] for \_ in range(len(table))]

for i, (key, value) in enumerate(table.items()):

for j, (key1, value1) in enumerate(table.items()):

if i >= j:

continue

else:

p = 0

n = 0

for k, criteria in enumerate(weights.keys()):

if marks[criteria](value[criteria]) > \

marks[criteria](value1[criteria]):

p += weights[criteria]

elif marks[criteria](value[criteria]) < \

marks[criteria](value1[criteria]):

n += weights[criteria]

if p == n:

matrix[i][j] = 0

matrix[j][i] = 0

Продолжение Листинга Б.1

elif n == 0:

matrix[i][j] = 1e+6

matrix[j][i] = 0

elif p == 0:

matrix[i][j] = 0

matrix[j][i] = 1e+6

elif p > n:

matrix[i][j] = round(p / n, 2)

matrix[j][i] = 0

elif p < n:

matrix[i][j] = 0

matrix[j][i] = round(n / p, 2)

csv\_write(matrix)

print('Матрица предпочтений:')

print\_matrix(matrix)

dot = graphviz.Digraph('Граф предпочтений', format='svg')

for i in range(len(matrix)):

dot.node(f'{i + 1}')

for i in range(len(matrix)):

for j in range(len(matrix[i])):

if matrix[i][j] != 0:

dot.edge(f'{i + 1}', f'{j + 1}')

dot.render(view=True)

matrix = [[elem if elem >= border else 0 for elem in row] for row in matrix]

print(f'\n\nМатрица предпочтений с порогом c = {border}:')

print\_matrix(matrix)

levels = get\_levels(matrix)

print('Оптимальное решение: ' + ', '.join(map(lambda x: str(x + 1), levels[0])))

dot1 = graphviz.Digraph('Граф предпочтений с порогом', format='svg')

for i in range(len(matrix)):

for j in range(len(matrix[i])):

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

dot1.edge(f'{i + 1}', f'{j + 1}')

for i in range(len(levels)):

sub = graphviz.Digraph(name=f'Подграф {i}')

sub.attr(rank='same')

sub.node(f'{i + 1}-й уровень')

for j in levels[i]:

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

dot1.subgraph(sub)

dot1.render(view=True)

main()

### Приложение В

Код реализации МАИ на языке Python.

Листинг В.1. Реализация метода анализа иерархий.

import csv

import math

const\_index\_s = [0, 0, 0.58, 0.9, 1.12, 1.24, 1.32, 1.41, 1.45, 1.49, 1.51,

1.48, 1.56, 1.57, 1.59]

def print\_matrix(matrix):

print(10 \* ' ' + '||', end='')

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

print(f'{i}'.rjust(10), end='')

print('\n' + '=' \* (10 \* (len(matrix) + 1) + 2))

for i in range(len(matrix)):

print(f'{i + 1}'.rjust(10), end='||')

for j in range(len(matrix[i])):

print(f'{round(matrix[i][j], 3)}'.rjust(10), end='')

print()

def compare(first, second, scale):

if first == second:

return 1

if first > second:

if math.ceil((first - second) / scale) > 9:

return 9

return math.ceil((first - second) / scale)

if math.ceil((second - first) / scale) > 9:

return 1 / 9

return 1 / math.ceil((second - first) / scale)

def create\_prior\_matrix(table, criteria, scale):

prior\_matrix = [[0 for \_ in range(len(table))] for \_ in range(len(table))]

for i, value in enumerate(table.values()):

for j, value1 in enumerate(table.values()):

if i == j:

prior\_matrix[i][j] = 1

else:

prior\_matrix[i][j] = compare(value[criteria], value1[criteria],

scale)

return prior\_matrix

def product(arr):

p = 1

for i in arr:

p \*= i

return p

def get\_weight(matrix):

v = [product(row) \*\* (1 / len(matrix)) for row in matrix]

s = sum(v)

Продолжение Листинга В.1.

w = [elem / s for elem in v]

s\_col = [sum(matrix[j][i] for j in range(len(matrix))) for i in

range(len(matrix))]

p = [s\_col[i] \* w[i] for i in range(len(s\_col))]

p\_sum = sum(p)

index\_s = (p\_sum - len(matrix)) / (len(matrix) - 1)

os = index\_s / const\_index\_s[len(matrix) - 1]

if os < 0.1:

return {'Код': 0, 'Веса': w}

else:

for i in range(len(p)):

print(f'P{i + 1} = {p[i]}')

return {'Код': 1, 'Веса': w}

def main():

with open('alts.csv', 'r', encoding='utf-8') as file:

array = []

reader = csv.reader(file, delimiter=',')

for row in reader:

array.append(row)

table = dict.fromkeys(elem[0] for elem in array[1:])

crits = ['Популярность', 'Рейтинг', 'Темп',

'Длина', 'Агрессивность', 'Громкость']

for i in range(1, len(array)):

values = [int(elem) if ':' not in elem else elem for elem in

array[i][1:]]

table[array[i][0]] = dict.fromkeys(elem for elem in array[0][1:])

table[array[i][0]]['Популярность'] = values[0]

table[array[i][0]]['Рейтинг'] = values[1]

table[array[i][0]]['Темп'] = values[2]

table[array[i][0]]['Длина'] = -(int(values[3].split(':')[0]) \* 60 +

int(values[3].split(':')[1]))

table[array[i][0]]['Агрессивность'] = -values[4]

table[array[i][0]]['Громкость'] = values[5]

criteria = [[1, 1 / 3, 5, 3, 7, 6],

[3, 1, 6, 4, 9, 8],

[1 / 5, 1 / 6, 1, 1 / 2, 3, 3],

[1 / 3, 1 / 4, 2, 1, 5, 4],

[1 / 7, 1 / 9, 1 / 3, 1 / 5, 1, 1 / 2],

[1 / 6, 1 / 8, 1 / 3, 1 / 4, 2, 1]]

print('Матрица парного сравнения критериев:')

print\_matrix(criteria)

prior\_matrixes = []

weights = []

for i in range(len(crits)):

scale = int(input(f'Введите шкалу для критерия {crits[i]}: '))

prior\_matrixes.append(create\_prior\_matrix(table, crits[i], scale))

print(f'Матрица для критерия {crits[i]}:')

print\_matrix(prior\_matrixes[i])

weight = get\_weight(prior\_matrixes[i])

while weight['Код'] != 0:

print('Матрица не согласована')

print('Что нужно изменить?')

print('1. Значение scale\n2. Значение в определённой ячейке')

choice = int(input())

match choice:

Продолжение Листинга В.1.

case 1:

scale = int(input('Введите новое значение шкалы: '))

prior\_matrixes[i] = create\_prior\_matrix(table, crits[i],

scale)

print\_matrix(prior\_matrixes[i])

weight = get\_weight(prior\_matrixes[i])

case 2:

j, k = map(int,

input(

'Введите номер строки и столбца, где нужно поменять значение приоритета: ').split())

value = float(

input('Введите значение, на которое нужно поменять: '))

prior\_matrixes[i][j - 1][k - 1] = value

prior\_matrixes[i][k - 1][j - 1] = 1 / value

print\_matrix(prior\_matrixes[i])

weight = get\_weight(prior\_matrixes[i])

else:

weights.append(weight['Веса'])

print()

criteria\_weight = get\_weight(criteria)['Веса']

result = []

for i in range(len(weights[0])):

s = 0

for j in range(len(weights)):

s += criteria\_weight[j] \* weights[j][i]

result.append(s)

maxi = max(result)

max\_index = 0

for i in range(len(result)):

print(f'W{i + 1} = {result[i]}')

if result[i] == maxi:

max\_index = i

print(f'Лучшая альтернатива: A{max\_index + 1}')

main()

### Приложение Г

Код реализации графического метода на языке Python.

Листинг Г.1. Реализация графического метода.

import numpy as np

import matplotlib.pyplot as plt

import sys

from sympy import Line, Point

x1 = np.linspace(-10, 10.1, 100)

def input\_line():

A = int(input('Введите коэффициент перед x1: '))

B = int(input('Введите коэффициент перед x2: '))

C = int(input('Введите свободный коэффициент: '))

sign = input('Введите знак неравенства: ')

return sign, A, B, C

def add\_lines\_to\_plot(lines, y\_min, y\_max, shade=False):

plt.plot(x1, np.zeros\_like(x1), color='black')

plt.plot(np.zeros\_like(np.linspace(y\_min, y\_max, 100)),

np.linspace(y\_min, y\_max, 100), color='black')

for i, line in enumerate(lines):

if i == len(lines) - 1:

plt.plot(x1, line[0], label=line[1], linewidth=4)

else:

plt.plot(x1, line[0], label=line[1])

if shade:

if i == len(lines) - 1:

continue

if '<' in line[2]:

plt.fill\_between(x1, line[0], line[0] - 0.2,

color='black', alpha=0.3)

else:

plt.fill\_between(x1, line[0], line[0] + 0.2,

color='black', alpha=0.3)

plt.legend(loc='upper right')

def generate\_legend(A, B, C, sign):

s = ''

if A == 1:

s += 'x1'

elif A == -1:

s += '-x1'

elif A != 0:

s += f'{A}x1'

if B < 0:

s += ' - '

if B == -1:

s += 'x2'

else:

s += f'{-B}x2'

elif B > 0:

s += ' + '

if B == 1:

s += 'x2'

Продолжение Листинга Г.1.

else:

s += f'{B}x2'

match sign:

case '<=':

s += ' \u2264 '

case '>=':

s += ' \u2265 '

case '=':

return s

case \_:

s += f' {sign} '

s += f'{C}'

return s

def scale\_plot(lines, y\_min, y\_max):

global x1

choice = input('Нужно отмасштабировать? [y/n] ')

x\_min, x\_max = -10, 10

while choice != 'n':

x\_min = int(input('Введите минимальный х: '))

x\_max = int(input('Введите максимальный х: '))

x1 = np.linspace(x\_min, x\_max + 0.1, 100)

for line in lines:

A, B, C = line[3]

line[0] = (C - A \* x1) / B

y\_min = min(np.min(elem[0]) for elem in lines) - 1

y\_max = max(np.max(elem[0]) for elem in lines) + 1

plt.figure(figsize=(6, 6))

plt.xlim(x\_min - 1, x\_max + 1)

plt.ylim(y\_min, y\_max)

add\_lines\_to\_plot(lines, y\_min, y\_max, shade=True)

plt.show()

choice = input('Нужно отмасштабировать? [y/n] ')

return x\_min, x\_max, y\_min, y\_max

def fill\_area(mini, maxi):

global x1

res = np.array([True for \_ in range(x1.size)])

for elem1 in maxi:

for elem2 in mini:

res = (elem1 > elem2) & res

if len(maxi) != 0:

upper = maxi[0]

for i in range(1, len(maxi)):

upper = np.minimum(maxi[i], upper)

else:

upper = np.linspace(10, 10, 100)

if len(mini) != 0:

lower = mini[0]

for i in range(1, len(mini)):

lower = np.maximum(mini[i], lower)

else:

lower = np.zeros\_like(x1)

plt.fill\_between(x1, upper, lower,

where=res, color='red', alpha=0.3)

def find\_intersection(lines, msg):

global x1

Продолжение Листинга Г.1.

line1, line2 = tuple(input('Введите номера пересекающихся линий: ').split())

if line1 == 'x':

p11 = Point(0, 0)

p12 = Point(1, 0)

elif line1 == 'y':

p11 = Point(0, 0)

p12 = Point(0, 1)

else:

A1, B1, C1 = lines[int(line1) - 1][3]

if B1 != 0:

p11 = Point(0, C1 / B1)

p12 = Point(C1 / A1, 0)

else:

p11 = Point(0, C1)

p12 = Point(1, C1)

if line2 == 'x':

p21 = Point(0, 0)

p22 = Point(1, 0)

elif line2 == 'y':

p21 = Point(0, 0)

p22 = Point(0, 1)

else:

A2, B2, C2 = lines[int(line2) - 1][3]

if B2 != 0:

p21 = Point(0, C2 / B2)

p22 = Point(C2 / A2, 0)

else:

p21 = Point(0, C2)

p22 = Point(1, C2)

line1 = Line(p11, p12)

line2 = Line(p21, p22)

p = line1.intersection(line2)

A\_func, B\_func, C\_func = lines[len(lines) - 1][3]

plt.scatter(p[0].args[0], p[0].args[1])

print(f'{msg}: {p}; Значение функции равно {A\_func \* p[0].args[0] +

B\_func \* p[0].args[1]}')

def move(lines, y\_min, y\_max, mini, maxi):

global x1

msgs = ['Максимум', 'Минимум']

for msg in msgs:

delta = input('Введите смещение для целевой функции[число/n]: ')

while delta != 'n':

delta = float(delta)

last = len(lines) - 1

A, B, C = lines[last][3]

C += delta

lines[last][0] = (C - A \* x1) / B

lines[last][3] = (A, B, C)

plt.figure(figsize=(6, 6))

add\_lines\_to\_plot(lines, y\_min, y\_max)

fill\_area(mini, maxi)

plt.show()

delta = input('Введите смещение для целевой функции[число/n]: ')

plt.figure(figsize=(6, 6))

find\_intersection(lines, msg)

add\_lines\_to\_plot(lines, y\_min, y\_max)

fill\_area(mini, maxi)

Продолжение Листинга Г.1.

plt.show()

def main():

global x1

sys.stdin = open('input.txt', 'r')

n = int(input("Введите количество ограничений, накладываемых на х1 и х2: "))

lines = []

for i in range(n):

print(f'Введите {i + 1}-ое ограничение:')

sign, A, B, C = input\_line()

if B != 0:

new\_line = (C - A \* x1) / B

legend = f'{i + 1}: ' + generate\_legend(A, B, C, sign)

if B < 0 and '>' in sign:

sign = sign.replace('>', '<')

elif B < 0 and '<' in sign:

sign = sign.replace('<', '>')

lines.append([new\_line, legend, sign, (A, B, C)])

else:

if '>' in sign:

plt.xlim(xmin=C / A)

else:

plt.xlim(xmax=C / A)

print('Введите целевую функцию')

A\_func, B\_func = map(int, input('Введите коэффициенты при х1 и х2: ')

.split())

C\_func = 0

new\_line = (C\_func - A\_func \* x1) / B\_func

legend = f'Целевая функция: ' + generate\_legend(A\_func, B\_func, C\_func, '=')

lines.append([new\_line, legend, '=', (A\_func, B\_func, C\_func)])

y\_min = min(np.min(elem[0]) for elem in lines) - 1

y\_max = max(np.max(elem[0]) for elem in lines) + 1

plt.figure(figsize=(6, 6))

add\_lines\_to\_plot(lines, y\_min, y\_max, shade=True)

plt.show()

x\_min, x\_max, y\_min, y\_max = scale\_plot(lines, y\_min, y\_max)

mini = [line[0] for line in lines if '>' in line[2]]

maxi = [line[0] for line in lines if '<' in line[2]]

plt.figure(figsize=(6, 6))

add\_lines\_to\_plot(lines, y\_min, y\_max)

fill\_area(mini, maxi)

plt.show()

print(f'\nГрадиент: ({A\_func},{B\_func}); антиградиент: ({-A\_func}{-B\_func})')

plt.figure(figsize=(6, 6))

plt.arrow(0, 0, A\_func / 2, B\_func / 2,

color='black', head\_width=0.15, label='Градиент')

plt.arrow(0, 0, -A\_func / 2, -B\_func / 2,

color='blue', head\_width=0.15, label='Антиградиент')

add\_lines\_to\_plot(lines, y\_min, y\_max)

fill\_area(mini, maxi)

plt.show()

move(lines, y\_min, y\_max, mini, maxi)

main()

### Приложение Д

Код реализации симплексного метода на языке Python.

Листинг Д.1. Реализация симплексного метода.

import numpy as np

simplex\_table: np.ndarray[np.float64]

base\_coefs: np.ndarray[np.float64]

base\_vars: list[int]

func\_coefs: np.ndarray[np.float64]

main\_vars: list[int]

def print\_table():

print(' ' \* 10, end='')

for coef in func\_coefs:

print(f'{coef}'.ljust(8), end='')

print()

print(' ' \* 10, end='')

for var in main\_vars:

print(f'X{var + 1}'.ljust(8), end='')

print('A0'.ljust(8))

for i in range(len(simplex\_table)):

if i != len(simplex\_table) - 1:

print(f'{base\_coefs[i]}'.ljust(5), end='')

print(f'X{base\_vars[i] + 1}'.ljust(5), end='')

else:

print(' ' \* 10, end='')

for j in range(len(simplex\_table[i])):

print(f'{simplex\_table[i][j]:.3f}'.ljust(8), end='')

print()

def solution\_step():

global simplex\_table, base\_coefs, func\_coefs

last\_row = len(simplex\_table) - 1

last\_col = len(simplex\_table[0]) - 1

col = np.argmin(simplex\_table[last\_row])

if np.max(simplex\_table[:, col]) <= 0:

print('Оптимальное решение найти невозможно')

exit(1)

row, mini, main\_elem = 0, 1e7, 0

for i in range(len(simplex\_table) - 1):

if simplex\_table[i][col] > 0 and \

simplex\_table[i][last\_col] / simplex\_table[i][col] < mini:

main\_elem = simplex\_table[i][col]

mini = simplex\_table[i][last\_col] / simplex\_table[i][col]

row = i

for i in range(len(simplex\_table)):

for j in range(len(simplex\_table[0])):

if i == row or j == col:

continue

Продолжение Листинга Д.1.

simplex\_table[i][j] = (simplex\_table[i][j] \* main\_elem -

simplex\_table[row][j] \* simplex\_table[i][col]) / main\_elem

for i in range(len(simplex\_table)):

if i != row:

simplex\_table[i][col] = -simplex\_table[i][col] / main\_elem

for j in range(len(simplex\_table[0])):

if j != col:

simplex\_table[row][j] = simplex\_table[row][j] / main\_elem

simplex\_table[row][col] = 1 / simplex\_table[row][col]

base\_coefs[row], func\_coefs[col] = func\_coefs[col], base\_coefs[row]

base\_vars[row], main\_vars[col] = main\_vars[col], base\_vars[row]

def input\_coefs(n, k, mode):

global simplex\_table, base\_coefs, func\_coefs, main\_vars, base\_vars

func\_coefs = list(map(float,

input(

'Введите коэффициенты целевой функции через '

'пробел: ')

.split()))

func\_coefs = np.array(func\_coefs)

if mode == 2:

func\_coefs = -func\_coefs

main\_vars = [i for i in range(n)]

base\_coefs = np.zeros(shape=k, dtype=np.int32)

base\_vars = [i + n for i in range(k)]

simplex\_table = np.zeros(shape=(k + 1, n + 1), dtype=np.float64)

for i in range(len(simplex\_table) - 1):

print(f'Введите коэффициенты в {i + 1}-м ограничении')

for j in range(len(simplex\_table[i]) - 1):

simplex\_table[i][j] = float(

input(f'Введите коэффициент при х{j + 1}: '))

simplex\_table[i][len(simplex\_table[i]) - 1] = float(

input('Введите свободный коэффициент: ')

)

last = len(simplex\_table) - 1

for j in range(len(simplex\_table[0]) - 1):

simplex\_table[last][j] = -func\_coefs[j]

def main():

n = int(input('Введите количество переменных: '))

k = int(input('Введите количество ограничений: '))

mode = int(input('1. Максимизация\n2. Минимизация\n'))

input\_coefs(n, k, mode)

print('Исходная симплекс-таблица:')

print\_table()

flag = True

i = 1

while flag:

solution\_step()

last\_row = len(simplex\_table) - 1

print('\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_')

print(f'Итерация {i}\n\n')

Продолжение Листинга Д.1.

i += 1

print\_table()

if np.min(simplex\_table[last\_row]) > 0:

print('Решение найдено')

for i in range(len(base\_coefs)):

if base\_coefs[i] != 0:

print(f'X{base\_vars[i] + 1} -- {simplex\_table[i][n]: .3f} шт.')

print(f'Значение целевой функции: {simplex\_table[k][n]: .3f} [ден.ед.]')

flag = False

main()

### Приложение Е

Код реализации двойственной задачи на языке Python.

Листинг Е.1. Реализация двойственной задачи.

import numpy as np

start\_table: np.ndarray[np.float64]

start\_func\_coefs: np.ndarray[np.float64]

vectors: np.ndarray[np.float64]

simplex\_table: np.ndarray[np.float64]

base\_coefs: np.ndarray[np.float64]

base\_vars: list[int]

func\_coefs: np.ndarray[np.float64]

main\_vars: list[int]

def print\_table():

print(' ' \* 10, end='')

for coef in func\_coefs:

print(f'{coef}'.ljust(8), end='')

print()

print(' ' \* 10, end='')

for var in main\_vars:

print(f'X{var + 1}'.ljust(8), end='')

print('A0'.ljust(8))

for i in range(len(simplex\_table)):

if i != len(simplex\_table) - 1:

print(f'{base\_coefs[i]}'.ljust(5), end='')

print(f'X{base\_vars[i] + 1}'.ljust(5), end='')

else:

print(' ' \* 10, end='')

for j in range(len(simplex\_table[i])):

print(f'{simplex\_table[i][j]:.3f}'.ljust(8), end='')

print()

def solution\_step():

global simplex\_table, base\_coefs, func\_coefs

last\_row = len(simplex\_table) - 1

last\_col = len(simplex\_table[0]) - 1

col = np.argmin(simplex\_table[last\_row])

if np.max(simplex\_table[:, col]) <= 0:

print('Оптимальное решение найти невозможно')

exit(1)

temp = (simplex\_table[:last\_row, last\_col] /

np.maximum(0, simplex\_table[:last\_row, col]))

row = np.argmin(temp)

main\_elem = simplex\_table[row][col]

for i in range(len(simplex\_table)):

for j in range(len(simplex\_table[0])):

if i == row or j == col:

continue

Продолжение Листинга Е.1.

simplex\_table[i][j] = (simplex\_table[i][j] \* main\_elem -

simplex\_table[row][j] \* simplex\_table[i][

col]) / main\_elem

for i in range(len(simplex\_table)):

if i != row:

simplex\_table[i][col] = -simplex\_table[i][col] / main\_elem

for j in range(len(simplex\_table[0])):

if j != col:

simplex\_table[row][j] = simplex\_table[row][j] / main\_elem

simplex\_table[row][col] = 1 / simplex\_table[row][col]

base\_coefs[row], func\_coefs[col] = func\_coefs[col], base\_coefs[row]

base\_vars[row], main\_vars[col] = main\_vars[col], base\_vars[row]

def input\_coefs(n, k, mode):

global start\_table, start\_func\_coefs, vectors,\

simplex\_table, base\_coefs, func\_coefs, main\_vars, base\_vars

func\_coefs = list(map(float,

input(

'Введите коэффициенты целевой функции через '

'пробел: ')

.split()))

func\_coefs = np.array(func\_coefs)

if mode == 2:

func\_coefs = -func\_coefs

start\_func\_coefs = func\_coefs.copy()

main\_vars = [i for i in range(n)]

base\_coefs = np.zeros(shape=k, dtype=np.int32)

base\_vars = [i + n for i in range(k)]

simplex\_table = np.zeros(shape=(k + 1, n + 1), dtype=np.float64)

for i in range(len(simplex\_table) - 1):

print(f'Введите коэффициенты в {i + 1}-м ограничении')

for j in range(len(simplex\_table[i]) - 1):

simplex\_table[i][j] = float(

input(f'Введите коэффициент при х{j + 1}: '))

simplex\_table[i][len(simplex\_table[i]) - 1] = float(

input('Введите свободный коэффициент: ')

)

last = len(simplex\_table) - 1

for j in range(len(simplex\_table[0]) - 1):

simplex\_table[last][j] = -func\_coefs[j]

start\_table = simplex\_table.copy()

vectors = np.zeros(shape=(k, n + k), dtype=np.float64)

for i in range(n):

vectors[:, i] = simplex\_table[:-1, i]

for i in range(n, n + k):

vectors[i - n, i] = 1

def twin\_task():

global simplex\_table, base\_coefs, func\_coefs, main\_vars, base\_vars

# 1-я теорема

print()

Продолжение Листинга Е.1.

D = vectors[:, base\_vars[0]].reshape((len(start\_table) - 1, 1))

for i in range(1, len(start\_table) - 1):

D = np.hstack((D, vectors[:, base\_vars[i]].

reshape((len(start\_table) - 1, 1))))

D = np.linalg.inv(D)

print(f'Обратная матрица из базисных векторов:\n {D}')

twin\_coefs = np.matmul(base\_coefs, D)

answer = np.dot(twin\_coefs, start\_table[:-1, -1])

print(f'Ответ по первой теореме двойственности: {answer}')

# 2-я теорема

print('Вторая теорема двойственности')

A = start\_table[:-1, :-1].T

B = start\_func\_coefs.T

for i in range(len(A)):

if twin\_coefs[i] == 0:

A[i] = np.zeros(A.shape[1])

A[i][i] = 1

B[i] = 0

for i in range(len(A), len(twin\_coefs)):

A = np.vstack([A, np.zeros(A.shape[1])])

A[i][i] = 1

B = np.hstack([B, [0]]).T

print(A)

print(B)

solution = np.linalg.solve(A, B)

print(f'Решение: {solution}')

answer\_2 = np.dot(solution, start\_table[:-1, -1])

print(f'Ответ по второй теореме двойственности: {answer\_2}')

print()

# 3-я теорема

deltas = []

for j in range(D.shape[1]):

mini, maxi = 1000000, 0

for i in range(D.shape[0]):

if D[i][j] > 0:

mini = min(mini, start\_table[i, -1] / D[i][j])

elif D[i][j] < 0:

maxi = max(maxi, abs(start\_table[i, -1] / D[i][j]))

deltas.append((mini, maxi))

for i, (b\_min, b\_max) in enumerate(deltas):

print(f'b{i + 1} изменяется в интервале от'

f' {(start\_table[i, -1] - b\_min) if b\_min != 1000000 else

-np.infty} до {(start\_table[i, -1] + b\_max) if b\_max != 0

else np.infty}')

delta\_f = []

for i in range(len(twin\_coefs)):

if twin\_coefs[i] == 0:

continue

delta\_f.append(twin\_coefs[i] \* deltas[i][1])

print(f'Максимальное значение функции: {answer} +'

f' {' + '.join(map(str, delta\_f))}'

f' = {answer + sum(delta\_f)}')

def main():

n = int(input('Введите количество переменных: '))

k = int(input('Введите количество ограничений: '))

Продолжение Листинга Е.1.

mode = int(input('1. Максимизация\n2. Минимизация\n'))

input\_coefs(n, k, mode)

print('Исходная симплекс-таблица:')

print\_table()

flag = True

i = 1

while flag:

solution\_step()

last\_row = len(simplex\_table) - 1

print('\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_')

print(f'Итерация {i}\n\n')

i += 1

print\_table()

if np.min(simplex\_table[last\_row]) > 0:

print('Решение найдено')

for i in range(len(base\_coefs)):

if base\_coefs[i] != 0:

print(

f'X{base\_vars[i] + 1} -- {simplex\_table[i][n]: .3f} шт.')

print(

f'Значение целевой функции: {simplex\_table[k][n]: .3f} [ден.ед.]')

flag = False

twin\_task()

main()

### Приложение Ж

Код реализации транспортной задачи на языке Python.

Листинг Ж.1. Реализация транспортной задачи.

import numpy as np  
  
  
class Solution:  
 def \_\_init\_\_(self):  
 self.A = np.array(list(map(int, input('Введите кол-во грузов у '  
 'каждого поставщика через '  
 'пробел: ').split())))  
 self.B = np.array(list(map(int, input('Введите кол-во грузов у '  
 'каждого потребителя через '  
 'пробел: ').split())))  
 c = []  
 for i in range(len(self.A)):  
 c.append(np.array(  
 list(map(int,  
 input(f'Введите стоимость перевозок от {i + 1}-го '  
 f'поставщика').split()))))  
 self.C = np.vstack(c)  
 self.X = np.zeros\_like(self.C)  
 self.basis = []  
 self.U = np.zeros\_like(self.A)  
 self.V = np.zeros\_like(self.B)  
 self.marks = np.zeros\_like(self.C)  
  
 def print\_table(self, potential=True):  
  
 print(10 \* ' ', end='')  
 if potential:  
 for i in range(len(self.V)):  
 print(f'||v{i + 1}={self.V[i]}'.ljust(12), end='')  
 print('\n' + 10 \* ' ', end='')  
 for i in range(len(self.B)):  
 print(f'||B{i + 1}'.ljust(12), end='')  
 print('\n' + 10 \* ' ', end='')  
 for i in range(len(self.B)):  
 print(f'||{self.B[i]}'.ljust(12), end='')  
  
 for i in range(len(self.A)):  
 print('\n' + '=' \* (10 + 12 \* len(self.B)))  
 if potential:  
 print(f'u{i + 1}={self.U[i]} A{i + 1}'.rjust(10), end='')  
 else:  
 print(f'A{i + 1}'.rjust(10), end='')  
 for j in range(len(self.B)):  
 print('||' + f'{self.C[i][j]}'.rjust(10), end='')  
 print('\n' + f'{self.A[i]}'.rjust(10), end='')  
 for j in range(len(self.B)):  
 if (i, j) not in self.basis:  
 print('||' + f''.ljust(10), end='')  
 else:  
 print(  
 '||' + f'\033[31m{self.X[i][j]}'.ljust(15) + '\033[0m',  
 end='')  
 print('\n\n')

Продолжение Листинга Ж.1.

def print\_marks(self):  
 for i in range(len(self.A)):  
 for j in range(len(self.B)):  
 if (i, j) not in self.basis:  
 print(f'\u0394{i + 1}{j + 1} = {self.marks[i][j]}')  
  
 def min\_price(self):  
 visited = {'row': set(), 'col': set()}  
 while (len(visited['row']) + len(visited['col']) <  
 len(self.A) + len(self.B) - 1):  
 possible = [tuple(set(range(len(self.A))) - visited['row']),  
 tuple(set(range(len(self.B))) - visited['col'])]  
 min\_, min\_i, min\_j = 1000000, 0, 0  
 for i in possible[0]:  
 for j in possible[1]:  
 if self.C[i][j] < min\_:  
 min\_, min\_i, min\_j = self.C[i][j], i, j  
 self.basis.append((min\_i, min\_j))  
 if self.A[min\_i] - np.sum(self.X[min\_i, :]) < \  
 self.B[min\_j] - np.sum(self.X[:, min\_j]):  
 visited['row'].add(min\_i)  
 self.X[min\_i][min\_j] = self.A[min\_i] - np.sum(self.X[min\_i, :])  
 else:  
 visited['col'].add(min\_j)  
 self.X[min\_i][min\_j] = self.B[min\_j] - np.sum(self.X[:, min\_j])  
  
 def make\_first\_plan(self):  
 for i in range(len(self.A)):  
 for j in range(len(self.B)):  
 self.X[i][j] = min(self.A[i] - np.sum(self.X[i, :]),  
 self.B[j] - np.sum(self.X[:, j]))  
 if self.X[i][j] != 0:  
 self.basis.append((i, j))  
 if len(self.basis) != len(self.A) + len(self.B) - 1:  
 for i in range(len(self.basis) - 1):  
 if abs(self.basis[i][0] - self.basis[i + 1][0]) + \  
 abs(self.basis[i][1] - self.basis[i + 1][1]) > 1:  
 self.basis.append((self.basis[i + 1][0] - 1,  
 self.basis[i + 1][1]))  
  
 def solve\_potential(self):  
 order = [[0]]  
 visited = [[], []]  
 i = 0  
 while i < len(order):  
 temp = []  
 for k in order[i]:  
 if i % 2 == 0:  
 visited[0].append(k)  
 for j in range(len(self.B)):  
 if (k, j) in self.basis and j not in visited[1]:  
 self.V[j] = self.C[k][j] - self.U[k]  
 temp.append(j)  
 else:  
 visited[1].append(k)  
 for j in range(len(self.A)):  
 if (j, k) in self.basis and j not in visited[0]:  
 self.U[j] = self.C[j][k] - self.V[k]

Продолжение Листинга Ж.1.

temp.append(j)  
 if len(temp) != 0:  
 order.append(temp)  
 i += 1  
  
 def count\_marks(self):  
 U = self.U.reshape((len(self.U), 1))  
 V = self.V  
 for \_ in range(len(self.V) - 1):  
 U = np.hstack((U, self.U.reshape((len(self.U), 1))))  
 for \_ in range(len(self.U) - 1):  
 V = np.vstack((V, self.V))  
 self.marks = self.C - (U + V)  
  
 def dfs(self, i, j, mode, cycle):  
 if len(cycle) != 1 and cycle[-1] == cycle[0]:  
 return True  
 result = False  
 if mode == 'v':  
 for base\_i, base\_j in self.basis:  
 if base\_j == j and base\_i != i:  
 cycle.append((base\_i, base\_j))  
 result = result or self.dfs(base\_i, base\_j,  
 'h', cycle)  
 if result:  
 return True  
 elif mode == 'h':  
 for base\_i, base\_j in self.basis:  
 if base\_i == i and base\_j != j:  
 cycle.append((base\_i, base\_j))  
 result = result or self.dfs(base\_i, base\_j,  
 'v', cycle)  
 if result:  
 return True  
 if not result:  
 cycle.pop()  
 return result  
  
 def recount(self, main\_i, main\_j):  
 self.basis.append((main\_i, main\_j))  
 cycle = [(main\_i, main\_j)]  
 self.dfs(main\_i, main\_j, 'v', cycle)  
 cycle.pop()  
 diff = min(self.X[i][j] for i, j in cycle[1::2])  
 cnt = 0  
 for index, (i, j) in enumerate(cycle):  
 if index % 2 == 0:  
 self.X[i][j] += diff  
 else:  
 self.X[i][j] -= diff  
 if self.X[i][j] == 0 and cnt != 1:  
 self.basis.remove((i, j))  
 cnt += 1  
  
 def count\_function(self):  
 return np.dot(self.C.reshape(self.A.size \* self.B.size),  
 self.X.reshape(self.A.size \* self.B.size))  
  
 def solve(self):  
 print('Таблица с исходными данными:')

Продолжение Листинга Ж.1.

self.print\_table(False)  
  
 self.min\_price()  
 print('Таблица, полученная методом минимальной стоимости')  
 self.print\_table()  
 print(f'Посчитанная стоимость перевозок: {self.count\_function()}')  
 self.basis = []  
 self.X = np.zeros\_like(self.C)  
  
 self.make\_first\_plan()  
 print('Таблица с начальным опорным планом, полученным'  
 'с помощью метода северо-западного угла:')  
 self.solve\_potential()  
 self.count\_marks()  
 self.print\_table()  
 while np.min(self.marks) < 0:  
 print(f'Стоимость перевозок: {self.count\_function()}')  
 self.print\_marks()  
 print('Есть отрицательные оценки, происходит перерасчёт плана...')  
 i, j = np.where(self.marks == np.min(self.marks))  
 i, j = i[0], j[0]  
 self.recount(i, j)  
 self.solve\_potential()  
 self.count\_marks()  
 self.print\_table()  
 self.print\_marks()  
 print(f'Оптимальное решение найдено: {self.count\_function()}')  
  
  
def main():  
 solution = Solution()  
 solution.solve()  
  
  
main()