МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования

«МИРЭА – Российский технологический университет»

РТУ МИРЭА

Институт информационных технологий

Кафедра вычислительной техники

КУРСОВАЯ РАБОТА

По дисциплине «Теория принятия решений»
(наименование дисциплины)

Тема курсовой работы Методы многокритериальной оптимизации и линейного
(наименование темы)
программирования «Выбор оптимального технического высшего учебного заведения»

Студент группы ИКБО-04-22 Кликушин Владислав Игоревич
(учебная группа) *(Фамилия Имя Отчество)* *(подпись студента)*

Руководитель курсовой работы доцент каф.ВТ Сорокин А.Б.
(Должность, звание, ученая степень) *(подпись руководителя)*

Консультант Жемерин А.И.
(Должность, звание, ученая степень) *(подпись консультанта)*

Работа представлена к защите «20» 08 2024 г.

Допущен к защите «20» 08 2024г.

отл.

Москва 2024 г.

МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«МИРЭА – Российский технологический университет»

РТУ МИРЭА

Институт информационных технологий

Кафедра вычислительной техники

Утверждаю

Заведующий кафедрой ______________

Подпись

Платонова О.В.

ФИО

«_19_» февраля 2024г.

ЗАДАНИЕ

На выполнение курсовой работы

по дисциплине **«Теория принятия решений»**

Студент Кликушин Владислав Игоревич Группа ИКБО-04-22

Тема Методы многокритериальной оптимизации и линейного программирования «Выбор оптимального технического высшего учебного заведения»

Исходные данные:

1. Описания исходных данных для многокритериальной оптимизации: 10 альтернатив и 4-6 критериев.
2. Линейное программирование по вариантам.

Перечень вопросов, подлежащих разработке, и обязательного графического материала:

1. Реализовать расчет и консольное приложение:

1.1. Для многокритериальной оптимизации: Парето множество и его сужение, метод Электра II и метод анализ иерархий;

1.2. Для линейного программирования: графический метод, симплекс метод, двойственную (3 теоремы) и транспортную задачу.

Срок представления к защите курсовой работы: до «_31_» мая 2024 г.

Задание на курсовую работу выдал ______________ (Сорокин А.Б.)

Подпись *ФИО консультанта*

«_19_» февраля 2024 г.

Задание на курсовую работу получил ______________ (Кликушин В.И.)

Подпись *ФИО исполнителя*

«_19_» февраля 2024 г.

Москва 2024г.

ОТЗЫВ

на курсовую работу

по дисциплине «Теория принятия решений»

Студент Кликушин Владислав Игоревич группа ИКБО-04-22
(ФИО студента) *(Группа)*

Характеристика курсовой работы

Критерий	Да	Нет	Не полностью
1. Соответствие содержания курсовой работы указанной теме	+		
2. Соответствие курсовой работы заданию	+		
3. Соответствие рекомендациям по оформлению текста, таблиц, рисунков и пр.	+		
4. Полнота выполнения всех пунктов задания	+		
5. Логичность и системность содержания курсовой работы	+		
6. Отсутствие фактических грубых ошибок	+		

Замечаний: нет

Рекомендуемая оценка: отлично

_______________ доцент каф.ВТ Сорокин А.Б.
(Подпись руководителя) *(ФИО руководителя)*

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

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

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

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

Решения могут делиться на два типа:

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

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

Традиционно в математическом программировании выделяют следующие основные разделы.

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

Нелинейное программирование – целевая функция и ограничения нелинейны.

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

Математическая модель любой задачи линейного программирования включает в себя:

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

1 МЕТОД ПАРЕТО

1.1 Введение

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

Предметная область: выбор оптимального высшего учебного заведения.

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

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

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

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

1.2 Выбор Парето-оптимального множества

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

- Проходной балл на бюджет, взятый за 2023 год;
- Количество бюджетных мест, взятое за 2023 год;

- Минимальная стоимость обучения за год на 2023 год;
- Размер государственной академической стипендии для студентов, сдавших сессию на «хорошо» и «отлично» на 2023 год;
- Национальный рейтинг университета на основе информации с сайта academia.interfax.ru;
- Расстояние до общежития (километры).

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

Таблица 1.1.1 – Альтернативы и критерии

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

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

Таблица 1.1.2 – Сравнения альтернатив

	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10
A1	х	х	х	х	х	х	х	х	х	х
A2	A2	х	х	х	х	х	х	х	х	х
A3	A3	н	х	х	х	х	х	х	х	х
A4	н	н	н	х	х	х	х	х	х	х
A5	н	н	н	н	х	х	х	х	х	х
A6	A6	н	н	н	н	х	х	х	х	х
A7	н	н	н	н	A5	A6	х	х	х	х
A8	н	н	н	н	н	н	н	х	х	х
A9	н	н	н	н	н	н	н	н	х	х
A10	A10	н	н	н	н	A10	A10	A10	н	х

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

Таблица 1.1.3 – Парето-оптимальное множество

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

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

1.3 Указание верхних/нижних границ критериев

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

Таблица 1.3.1 – Результат указания границ критериев

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

Заметим, что альтернативы А2, А3, А6, А10 доминируют над А1 в то время, как А10 доминирует над А6. Следовательно, Парето-оптимальное множество состоит из альтернатив {2, 3, 10}. Единственную альтернативу выбрать не удалось в силу не слишком «жестких границ» для критериев, однако указание границ помогло уменьшить размер Парето-множества.

1.4 Субоптимизация

Главный критерий: рейтинг университета. Нижняя граница: проходной балл должен быть не ниже 290 и рейтинг университета должен быть не ниже 840 баллов. Верхняя граница: расстояние до общежития должно быть менее 14 км.

Отбросим варианты, которые не удовлетворяют данным ограничениям и составим таблицу 1.4.1 результата субоптимизации.

Таблица 1.4.1 – Результат субоптимизации

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

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

1.5 Лексикографическая оптимизация

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

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

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

Таблица 1.5.1 – Результат лексикографической оптимизации

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

Лексикографическая оптимизация позволяет выделить единственное наилучшее решение.

1.6 Результаты работы программы

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

Оптимальное-множество Парето:

	Альтернатива	Проходной балл (+)	Кол-во бюджетных мест (-)	Стоимость обучения (+)	Размер стипендии (+)	Рейтинг университета (+)	Расстояние до общежития (-)
0	МГТУ имени Н.Э. Баумана	291	84	363040	1854	964	4.2
1	ВШЭ	295	135	700000	1994	878	7.9
2	МАИ	247	25	277770	1765	761	12.4
3	ИТМО	304	45	349000	2252	838	3.4
4	НИЯУ МИФИ	300	37	287000	1875	975	15
5	МФТИ	290	30	432000	1900	965	4.8

Рисунок 1.6.1 – Парето-оптимальное множество

Установка верхних и нижних границ:

	Альтернатива	Проходной балл (+)	Кол-во бюджетных мест (-)	Стоимость обучения (+)	Размер стипендии (+)	Рейтинг университета (+)	Расстояние до общежития (-)
0	МГТУ имени Н.Э. Баумана	291	84	363040	1854	964	4.2
1	ВШЭ	295	135	700000	1994	878	7.9
2	МФТИ	290	30	432000	1900	965	4.8

Рисунок 1.6.2 – Установки верхних/нижних границ для критериев

Субоптимизация:

	Альтернатива	Проходной балл (+)	Кол-во бюджетных мест (-)	Стоимость обучения (+)	Размер стипендии (+)	Рейтинг университета (+)	Расстояние до общежития (-)
0	МФТИ	290	30	432000	1900	965	4.8

Рисунок 1.6.3 – Субоптимизация

Лексикографическая оптимизация:

	Альтернатива	Проходной балл (+)	Кол-во бюджетных мест (-)	Стоимость обучения (+)	Размер стипендии (+)	Рейтинг университета (+)	Расстояние до общежития (-)
0	НИЯУ МИФИ	300	37	287000	1875	975	15

```
PS C:\python_projects>
```

Рисунок 1.6.4 – Лексикографическая оптимизация

```python
print("Исходная таблица с альтернативами и критериями:".center(201))
print_table(data)

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

print("Установка верхних и нижних границ:".center(201))
branches = [{"Проходной балл (+)": 270}, {"Рейтинг университета (+)": 840},
            {"Расстояние до общежития (-)": 14}]
print_table(branches_and_boundaries(data, branches))

print("Субоптимизация:".center(201))
branches = [{"Проходной балл (+)": 290}, {"Расстояние до общежития (-)": 14}]
main_criteria = "Рейтинг университета (+)"
print_table(suboptimization(data, branches, main_criteria))

print("Лексикографическая оптимизация:".center(201))
priority = ("Рейтинг университета (+)", "Проходной балл (+)", "Стоимость обучения (+)",
            "Кол-во бюджетных мест (-)", "Расстояние до общежития (-)",
            "Размер стипендии (+)")
print_table(lexical_optimization(data, priority))
```

Рисунок 1.6.5 – Границы для оптимизации и приоритеты критериев

1.7 Заключение

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

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

2 МЕТОД ЭЛЕКТРА II

2.1 Введение

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

Предметная область: выбор оптимального высшего учебного заведения.

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

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

2.2 Выбор лучшего варианта

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

Таблица 2.2.1 – Таблица критериев для оценки альтернатив

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

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

Таблица 2.2.2 – Таблица оценок по критериям

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

Продолжение Таблицы 2.2.2

8	СГУ им. Чернышевского	5	5	5	5	10	5
9	НИЯУ МИФИ	15	5	10	10	15	15
10	МФТИ	15	5	15	5	15	15
Вес		4	5	2	5	2	4
Стремление		max	min	max	max	max	min

2.3 Веса предпочтений

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

Таблица 2.3.1 – Полная матрица предпочтений альтернатив

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

Продолжение Таблицы 2.3.1

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

По матрице построен граф предпочтений (Рисунок 2.3.1).

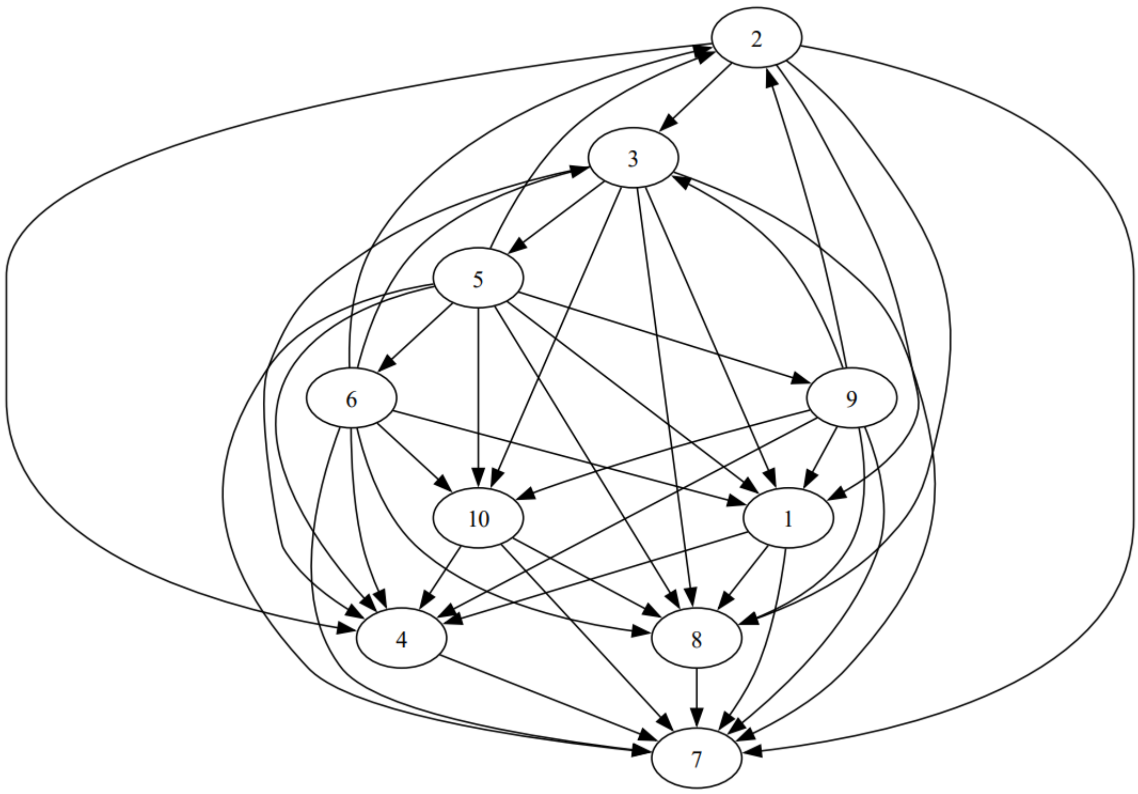

Рисунок 2.3.1 – Вид графа предпочтений

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

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

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

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

По этой матрице построен граф предпочтений (Рисунок 2.3.2).

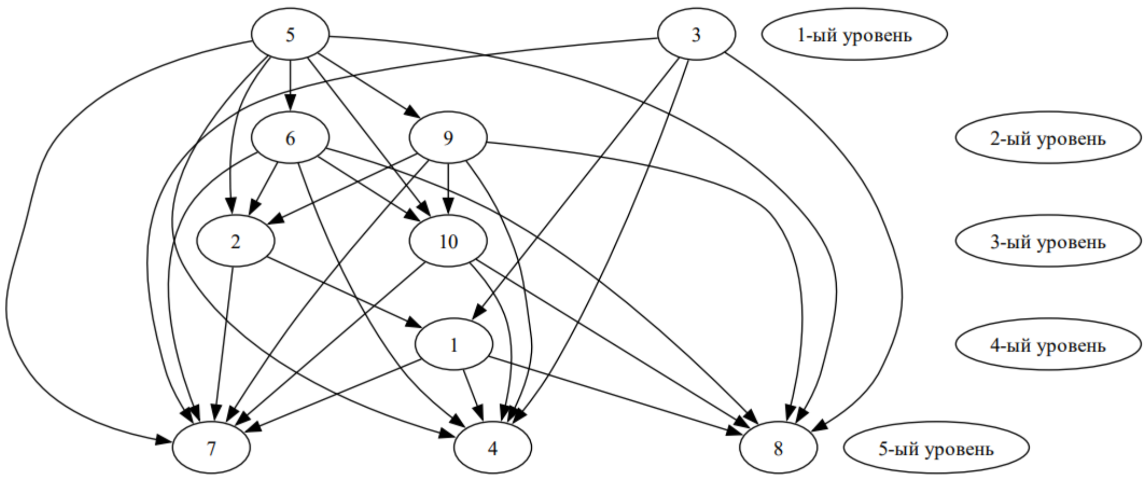

Рисунок 2.3.2 – Вид графа предпочтений для случая порога принятия решений С = 1.76

Циклов в графе нет, при этом граф остался целостным. Оптимальным решением является альтернатива А5 и А3.

2.4 Вывод

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

2.5 Результат работы программы

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

Матрица предпочтений:

	1	2	3	4	5	6	7	8	9	10
1	0	0	0	2.2	0	0	2.2	2.2	0	0
2	2.25	0	1.75	1.44	0	0	3.25	1.44	0	0
3	inf	0	0	2.6	1.2	0	2.6	2.2	0	1.29
4	0	0	0	0	0	0	1.25	0	0	0
5	1.25	2.25	0	2.75	0	2	4	2.75	2	2.25
6	1.75	2.5	1.17	2.75	0	0	4	3.25	0	2.5
7	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	1.17	0	0	0
9	1.75	2.5	1.17	2.75	0	0	4	3.25	0	2.5
10	0	0	0	2	0	0	3.25	2	0	0

Рисунок 2.5.1 – Вывод матрицы предпочтений

Матрица предпочтений:

	1	2	3	4	5	6	7	8	9	10
1	0	0	0	2.2	0	0	2.2	2.2	0	0
2	2.25	0	0	0	0	0	3.25	0	0	0
3	inf	0	0	2.6	0	0	2.6	2.2	0	0
4	0	0	0	0	0	0	0	0	0	0
5	0	2.25	0	2.75	0	2	4	2.75	2	2.25
6	0	2.5	0	2.75	0	0	4	3.25	0	2.5
7	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0
9	0	2.5	0	2.75	0	0	4	3.25	0	2.5
10	0	0	0	2	0	0	3.25	2	0	0

Рисунок 2.5.2 – Вывод матрицы предпочтений с порогом = 1.76

2.6 Заключение

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

3 МЕТОД АНАЛИЗА ИЕРАРХИЙ

3.1 Введение

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

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

3.2 Постановка задачи

Задача практической работы: выбрать лучшее техническое высшее учебное заведение.

3.3 Представление проблемы в виде иерархии

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

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

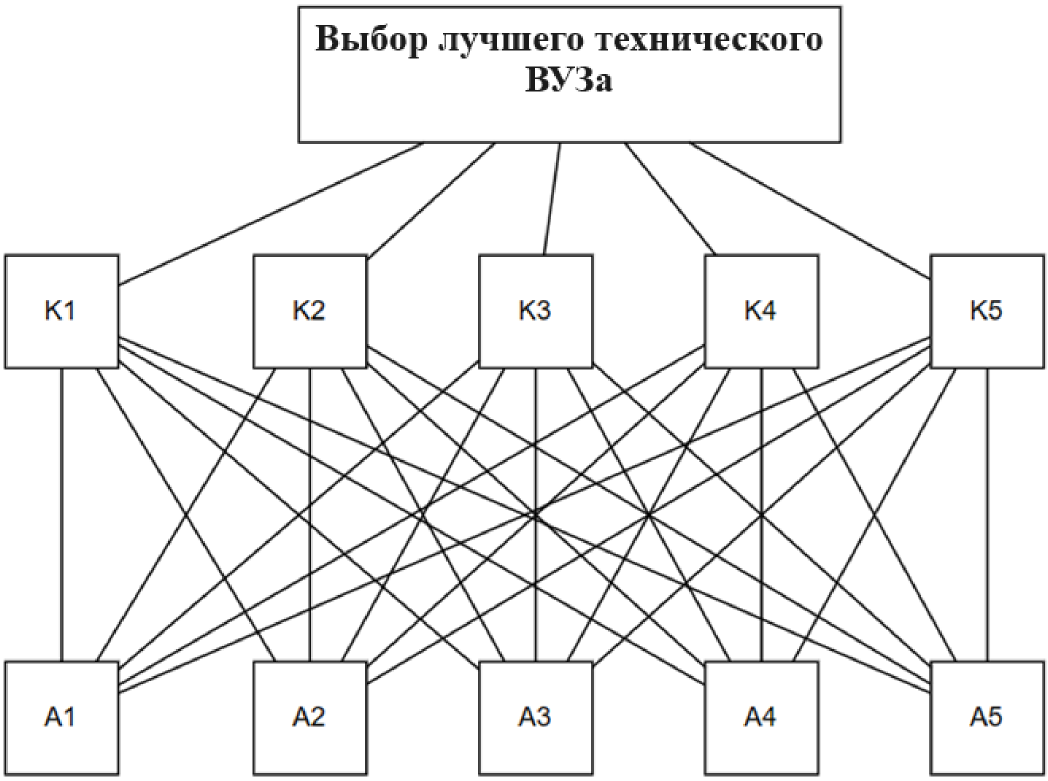

Рисунок 3.3.1 – Полная доминантная иерархия

Критерии:

К 1 – проходной балл (+);

К 2 – количество бюджетных мест (-);

К 3 – стоимость обучения (+);

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

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

Альтернативы:

А 1 – МГТУ имени Н.Э. Баумана;

А 2 – ВШЭ;

А 3 – МАИ;

А 4 – НИЯУ МИФИ;

А 5 – МФТИ.

3.4 Установка приоритетов критериев

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

Таблица 3.4.1 – Шкала относительной важности.

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

Шкала содержит соответствующие обратные значения.

3.5 Синтез приоритетов

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

Таблица 3.5.1 – Матрица парного сравнения критериев

Цель	К 1	К 2	К 3	К 4	К 5	Vi	W_{2i}
К 1	1	2	2	1/2	4	1.516	0.245
К 2	1/2	1	2	1/5	2	0.833	0.135
К 3	1/2	1/2	1	1/8	2	0.574	0.093
К 4	2	5	8	1	2	2.759	0.446
К 5	1/4	1/2	1/2	1/2	1	0.5	0.081
∑Vi						6.182	

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

Строка № 1

$V_1=(1*2*2*1/2*4)^{1/5}=1.516$;

Строка № 2

$V_2=(1/2*1*2*1/5*2)^{1/5}= 0.833$;

Строка № 3

$V_3=(1/2*1/2*1*1/8*2)^{1/5}= 0.574$;

Строка № 4

$V_4=(2*5*8*1*2)^{1/5}=2.759$;

Строка № 5

$V_5=(1/4*1/2*1/2*1/2*1)^{1/5}=0.5$.

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

$\sum Vi = V_1 + V_2 + V_3 + V_4 + V_5 = 1.516 + 0.833 + 0.574 + 2.759 + 0.5 = 6.182$.

Найдена важность приоритетов W_{2i}, для этого каждое из чисел Vi разделено на ∑Vi.

Строка № 1

$W_{21}= 1.516/ \sum Vi = 0.245 = Y_{21}$;

Строка № 2

W_{22}= 0.833 / $\sum Vi$ = 0.135 = Y_{22};

Строка № 3

W_{23}= 0.574 / $\sum Vi$ = 0.093 = Y_{23};

Строка № 4

W_{24}= 2.759 / $\sum Vi$ = 0.446 = Y_{24};

Строка № 5

W_{25}= 0.5 / $\sum Vi$ = 0.081 = Y_{25}.

В результате получен вектор приоритетов:

W_{2i} = (Y_{21}=0.245; Y_{22}=0.135; Y_{23}=0.093; Y_{24}=0.446; Y_{25}=0.081), где индекс 2 означает, что вектор приоритетов относится ко второму уровню иерархии.

К 1 – проходной балл (Таблица 3.5.2);

Таблица 3.5.2 – Матрица сравнения по критерию 1

К1	А1	А2	А3	А4	А5	VK1Y	W3K1Y
А1	1	1/2	8	1/4	2	1.149	0.188
А2	2	1	6	1/2	2	1.644	0.269
А3	1/8	1/6	1	1/5	1/6	0.213	0.035
А4	4	2	5	1	2	2.639	0.432
А5	1/2	1/2	1/6	1/2	1	0.461	0.075
$\sum V_{K1Y}$						6.053	

Определена относительная ценность каждого элемента.

Строка № 1

$V_{K11}=(1*1/2*8*1/4*2)^{1/5}=1.149$;

Строка № 2

$V_{K12}=(2*1*6*1/2*2)^{1/5}=1.644$;

Строка № 3

$V_{K13}=(1/8*1/6*1*1/5*1/6)^{1/5}=0.213$;

Строка № 4

$V_{K14}=(4*2*8*1*2)^{1/5}=2.639$;

Строка № 5

$V_{K15}=(1/2*1/2*1/6*1/2*1)^{1/5}=0.461$.

Проведена нормализация полученных чисел. Для этого определен нормирующий коэффициент $\sum V_{K1Y}$.

$\sum V_{K1Y} = V_{K11} + V_{K12} + V_{K13} + V_{K14} + V_{K15} = 1.149 + 1.644 + 0.213 + 2.639 + 0.461 = 6.106$.

Найдена важность приоритетов W_{3K1Y}, для этого каждое из чисел V_{K1Y} разделено на $\sum V_{K1Y}$.

Строка № 1

$W_{3K11} = 1.149 / \sum Vi = 1.149 / 6.106 = 0.188$;

Строка № 2

$W_{3K12} = 1.644 / \sum Vi = 1.644 / 6.106 = 0.269$;

Строка № 3

$W_{3K13} = 0.213 / \sum Vi = 0.213 / 6.106 = 0.035$;

Строка № 4

$W_{3K14} = 2.639 / \sum Vi = 2.639 / 6.106 = 0.432$;

Строка № 5

$W_{3K15} = 0.461 / \sum Vi = 0.461 / 6.106 = 0.075$;

В результате получаем вектор приоритетов:

$W_{3K1Y} = (Y_{311}=0.188;\ Y_{312}=0.269;\ Y_{313}=0.035;\ Y_{314}=0.432;\ Y_{315}=0.075)$,

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

К 2 – количество бюджетных мест (Таблица 3.5.3):

Таблица 3.5.3 – Матрица сравнения по критерию 2

К2	А1	А2	А3	А4	А5	V_{K2Y}	W_{3K2Y}
А1	1	5	1/6	1/5	1/5	0.507	0.074
А2	1/5	1	1/8	1/5	1/4	0.263	0.038
А3	6	8	1	2	2	2.862	0.415
А4	5	5	1/2	1	1/2	1.443	0.209
А5	5	4	1/2	2	1	1.821	0.264
$\sum V_{K2Y}$						6.896	

Определена относительная ценность каждого элемента.

Строка № 1

$V_{K21}=(1*5*1/6*1/5*1/5)^{1/5}=0.507$;

Строка № 2

$V_{K22}=(1/5*1*1/8*1/5*1/4)^{1/5}=0.263$;

Строка № 3

$V_{K23}=(6*8*1*2*2)^{1/5}=2.862$;

Строка № 4

$V_{K24}=(5*5*1/2*1*1/2)^{1/5}=1.443$;

Строка № 5

$V_{K25}=(5*4*1/2*2*1)^{1/5}=1.821$.

Проведена нормализация полученных чисел. Для этого определен нормирующий коэффициент $\sum V_{K2Y}$.

$\sum V_{K2Y} = V_{K21} + V_{K22} + V_{K23} + V_{K24} + V_{K25} = 0.507 + 0.263 + 2.862 + 1.443 + 1.821 = 6.896$.

Найдена важность приоритетов W_{3K2Y}, для этого каждое из чисел V_{K2Y} разделено на $\sum V_{K2Y}$.

Строка № 1

$W_{3K21}= 0.507 / \sum Vi = 0.507 / 6.896 = 0.074$;

Строка № 2

$W_{3K22}= 0.263 / \sum Vi = 0.263 / 6.896 = 0.038$;

Строка № 3

$W_{3K23}= 2.862 / \sum Vi = 2.862 / 6.896 = 0.415$;

Строка № 4

$W_{3K24}= 1.443 / \sum Vi = 1.443 / 6.896 = 0.209$;

Строка № 5

$W_{3K25}= 1.821 / \sum Vi = 1.821 / 6.896 = 0.264$.

В результате получаем вектор приоритетов:

$W_{3K2Y} = (Y_{321}=0.074; Y_{322}=0.038; Y_{323}=0.415; Y_{324}=0.209; Y_{325}=0.264)$,

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

К 3 – стоимость обучения (Таблица 3.5.4):

Таблица 3.5.4 – Матрица сравнения по критерию 3

К3	А1	А2	А3	А4	А5	V_{K3Y}	W_{3K3Y}
А1	1	1/8	4	4	1/4	0.871	0.107
А2	8	1	9	8	4	4.704	0.578
А3	1/4	1/9	1	1/2	1/5	0.308	0.038
А4	1/4	1/8	2	1	1/4	0.435	0.053
А5	4	1/4	5	4	1	1.821	0.224
V_{K35}						8.139	

Определена относительная ценность каждого элемента.

Строка № 1

$V_{K31}=(1*1/8*4*4*1/4)^{1/5}= 0.871$;

Строка № 2

$V_{K32}=(8*1*9*8*4)^{1/5}= 4.704$;

Строка № 3

$V_{K33}=(1/4*1/9*1*1/2*1/5)^{1/5}= 0.308$;

Строка № 4

$V_{K34}=(1/4*1/8*2*1*1/4)^{1/5}= 0.435$;

Строка № 5

$V_{K35}=(4*1/4*5*4*1)^{1/5}=1.821$.

Проведена нормализация полученных чисел. Для этого определен нормирующий коэффициент $\sum V_{K3Y}$.

$\sum V_{K3Y} = V_{K31} + V_{K32} + V_{K33} + V_{K34} + V_{K35} = 0.871 + 4.704 + 0.308 + 0.435 + 1.821 = 8.139$.

Найдена важность приоритетов W_{3K2Y}, для этого каждое из чисел V_{K2Y} разделено на $\sum V_{K2Y}$.

Строка № 1

$W_{3K31}= 0.871 / \sum Vi = 0.871 / 8.139 = 0.107$;

Строка № 2

$W_{3K32}= 4.704/ \sum Vi = 4.704 / 8.139 = 0.578$;

Строка № 3

$W_{3K33}= 0.308 / \sum Vi = 0.308 / 8.139 = 0.038$;

Строка № 4

W_{3K34}= 0.435 / ∑Vi = 0.435 / 8.139 = 0.053;

Строка № 5

W_{3K35}= 1.821 / ∑Vi = 1.821 / 8.139 = 0.224.

В результате получаем вектор приоритетов:

W_{3K3Y} = (Y_{331}=0.107; Y_{332}=0.578; Y_{333}=0.038; Y_{334}=0.053; Y_{335}=0.224),

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

К 4 – рейтинг университета (Таблица 3.5.5);

Таблица 3.5.5 – Матрица сравнения по критерию 4

К4	А1	А2	А3	А4	А5	V_{K4Y}	W_{3K4Y}
А1	1	4	8	1/2	1	1.741	0.261
А2	1/4	1	4	1/3	1/3	0.644	0.097
А3	1/8	1/4	1	1/8	1/8	0.218	0.033
А4	2	3	8	1	1	2.169	0.326
А5	1	3	8	1	1	1.888	0.283
$\sum V_{K4Y}$						6.66	

Определена относительная ценность каждого элемента.

Строка № 1

$V_{K41}=(1*4*8*1/2*1)^{1/5} = 1.741$;

Строка № 2

$V_{K42}=(1/4*1*4*1/3*1/3)^{1/5} = 0.644$;

Строка № 3

$V_{K43}=(1/8*1/4*1*1/8*1/8)^{1/5} = 0.218$;

Строка № 4

$V_{K44}=(2*3*8*1*1)^{1/5} = 2.169$;

Строка № 5

$V_{K45}=(1*3*8*1*1)^{1/5} = 1.888$.

Проведена нормализация полученных чисел. Для этого определен нормирующий коэффициент $\sum V_{K4Y}$.

$\sum V_{K4Y} = V_{K41} + V_{K42} + V_{K43} + V_{K44} + V_{K45}$ = 1.741 + 0.644 + 0.218 + 2.169 + 1.888 = 6.66.

Найдена важность приоритетов W_{3K4Y}, для этого каждое из чисел V_{K4Y} разделено на $\sum V_{K4Y}$.

Строка № 1

W_{3K41}= 1.741/ $\sum$Vi = 1.741 / 6.66 = 0.261;

Строка № 2

W_{3K42}= 0.644/ $\sum$Vi = 0.644 / 6.66 = 0.097;

Строка № 3

W_{3K43}= 0.218/ $\sum$Vi = 0.218 / 6.66 = 0.033;

Строка № 4

W_{3K44}= 2.169/ $\sum$Vi = 2.169 / 6.66 = 0.326;

Строка № 5

W_{3K45}= 1.888/ $\sum$Vi = 1.888 / 6.66 = 0.283.

В результате получаем вектор приоритетов:

W_{3K4Y} = (Y_{341}=0.261; Y_{342}=0.097; Y_{343}=0.033; Y_{344}=0.326; Y_{345}=0.283),

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

К 5 – Расстояние до общежития (Таблица 3.5.6).

Таблица 3.5.6 – Матрица сравнения по критерию 5.

К5	A1	A2	A3	A4	A5	V_{K5Y}	W_{3K5Y}
A1	1	5	8	9	2	3.728	0.47
A2	1/5	1	4	5	1/3	1.059	0.134
A3	1/8	1/4	1	2	1/8	0.379	0.048
A4	1/9	1/5	1/2	1	1/8	0.268	0.034
A5	1/2	3	8	8	1	2.491	0.314
$\sum V_{K5Y}$						7.925	

Определена относительная ценность каждого элемента.

Строка № 1

$V_{K51}=(1*5*8*9*2)^{1/5}$ =3.728;

Строка № 2

$V_{K52}=(1/5*1*4*5*1/3)^{1/5}$ =1.059;

Строка № 3

V_{K53}=(1/8*1/4*1*2*1/8)$^{1/5}$ =0.379;

Строка № 4

V_{K54}=(1/9*1/5*1/2*1*1/8)$^{1/5}$ =0.268;

Строка № 5

V_{K55}=(1/2*3*8*8*1)$^{1/5}$ =2.491.

Проведена нормализация полученных чисел. Для этого определен нормирующий коэффициент $\sum V_{K5Y}$.

$\sum V_{K5Y} = V_{K51} + V_{K52} + V_{K53} + V_{K54} + V_{K55} = 3.728 + 1.059 + 0.379 + 0.268 + 2.491 = 7.925$.

Найдена важность приоритетов W_{3K5Y}, для этого каждое из чисел V_{K5Y} разделено на $\sum V_{K5Y}$.

Строка № 1

W_{3K51}= 3.728/ $\sum$Vi = 0.47;

Строка № 2

W_{3K52}= 1.059/ $\sum$Vi = 0.134;

Строка № 3

W_{3K53}= 0.379/ $\sum$Vi = 0.048;

Строка № 4

W_{3K54}= 0.268/ $\sum$Vi = 0.034;

Строка № 5

W_{3K55}= 2.491/ $\sum$Vi = 0.314.

В результате получаем вектор приоритетов:

W_{3K5Y} = (Y_{351}=0.47; Y_{352}=0.134; Y_{353}=0.048; Y_{354}=0.034; Y_{355}=0.314),

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

3.6 Согласованность локальных приоритетов

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

Метод анализа иерархий дает возможность провести такую оценку.

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

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

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

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

Таблица 3.6.1 – Матрица «Выбор лучшего технического вуза»

Цель	К 1	К 2	К 3	К 4	К 5	W_{2i}
К 1	1	2	2	1/2	4	0.245
К 2	1/2	1	2	1/5	2	0.135
К 3	1/2	1/2	1	1/8	2	0.093
К 4	2	5	8	1	2	0.446
К 5	1/4	1/2	1/2	1/2	1	0.081

Определена сумма каждого столбца матрицы суждений.

$S_1 = 1 + 1/2 + 1/2 + 2 + 1/4 = 4.25;$

$S_2 = 2 + 1 + 1/2 + 5 + 1/2 = 9;$

$S_3 = 2 + 2 + 1 + 8 + 1/2 = 13.5;$

$S_4 = 1/2 + 1/5 + 1/8 + 1 + 1/2 = 2.325;$

$S_5 = 4 + 2 + 2 + 2 +1 = 11.$

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

$P_1 = S_1 x W_{21} = 1.041$;

$P_2 = S_2 x W_{22} = 1.215$;

$P_3 = S_3 x W_{23} = 1.256$;

$P_4 = S_4 x W_{24} = 1.037$;

$P_5 = S_5 x W_{25} = 0.891$.

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

$\lambda_{max} = P_1 + P_2 + P_3 + P_4 + P_5 = 5.44$.

Отклонение от согласованности выражается индексом согласованности.

$ИС = (\lambda_{max} - n)/(n - 1) = (5.222-5)/(5-1) = 0.11$.

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

ОС = ИС/СИ = 0.098.

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

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

Таблица 3.6.2 – Матрица сравнения по критерию 1.

К1	А1	А2	А3	А4	А5	W3K1Y
А1	1	1/2	8	1/4	2	0.188
А2	2	1	6	1/2	2	0.269
А3	1/8	1/6	1	1/5	1/6	0.035
А4	4	2	5	1	2	0.432
А5	1/2	1/2	1/6	1/2	1	0.075

Определяется сумма каждого столбца матрицы суждений.

$S_{1К1} = 1 + 2 + 1/8 + 4 + 1/2 = 7.625$;

$S_{2\,К1} = 1/2 + 1 + 1/6 + 2 + 1/2 = 4.167$;

$S_{3\,К1} = 8 + 6 + 1 + 5 + 1/6 = 20.167$;

$S_{4\,К1} = 1/4 + 1/2 + 1/5 + 1 + 1/2 = 2.45$;

$S_{5\,К1} = 2 + 2 + 1/6 + 2 + 1 = 7.167$.

Затем полученный результат умножен на компоненту нормализованного вектора приоритетов.

$P_{1\ K1} = S_1 \text{ x } W_{3K11} = 1.487;$

$P_{2\ K1} = S_2 \text{ x } W_{3K12} = 1.163;$

$P_{3\ K1} = S_3 \text{ x } W_{3K13} = 0.807;$

$P_{4\ K1} = S_1 \text{ x } W_{3K14} = 1;$

$P_{5\ K1} = S_1 \text{ x } W_{3K15} = 0.559.$

Найдена пропорциональность предпочтений.

$\lambda_{max\ K1} = P_{1K1} + P_{2K1} + P_{3K1} + P_{4K1} + P_{5K1} = 5.016.$

Отклонение от согласованности выражается индексом согласованности.

$ИС_{K1} = (\lambda_{max\ K1} - n)/(n - 1) = 0.004.$

Найдено отношение согласованности ОС.

$ОС_{K1} = ИС/СИ = 0.004.$

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

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

Таблица 3.6.3 – Матрица сравнения по критерию 2.

К2	А1	А2	А3	А4	А5	W_{3K2Y}
А1	1	5	1/6	1/5	1/5	0.074
А2	1/5	1	1/8	1/5	1/4	0.038
А3	6	8	1	2	2	0.415
А4	5	5	1/2	1	1/2	0.209
А5	5	4	1/2	2	1	0.264

Определена сумма каждого столбца матрицы суждений.

$S_{1K2} = 1 + 1/5 + 6 + 5 + 5 = 17.2;$

$S_{2\ K2} = 5 + 1 + 8 + 5 + 4 = 23;$

$S_{3\ K2} = 1/6 + 1/8 + 1 + 1/2 + 1/2 = 2.292;$

$S_{4\ K2} = 1/5 + 1/5 + 2 + 1 + 2 = 5.4;$

$S_{5\ K2} = 1/5 + 1/4 + 2 + 1/2 + 1 = 3.95.$

Затем полученный результат умножен на компоненту нормализованного вектора приоритетов.

$P_{1\ К2} = S_1 \times W_{3\ К21} = 1.273$;

$P_{2\ К2} = S_2 \times W_{3\ К22} = 0.874$;

$P_{3\ К2} = S_3 \times W_{3\ К23} = 0.951$;

$P_{4\ К2} = S_4 \times W_{3\ К24} = 1.129$;

$P_{5\ К2} = S_5 \times W_{3\ К25} = 1.043$.

Найдена пропорциональность предпочтений.

$$\lambda_{max\ К2} = P_{1К2} + P_{2К2} + P_{3К2} + P_{4К2} + P_{5К2} = 5.27.$$

Отклонение от согласованности выражается индексом согласованности.

$$ИС_{\ К2} = (\lambda_{max\ К2} - n)/(n - 1) = (5.27 – 5) / (5 – 1) = 0.067.$$

Найдено отношение согласованности ОС.

$$ОС_{\ К2} = ИС/СИ = 0.06.$$

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

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

Таблица 3.6.4 – Матрица сравнения по критерию 3.

К3	А1	А2	А3	А4	А5	$W_{3К3Y}$
А1	1	1/8	4	4	1/4	0.107
А2	8	1	9	8	4	0.578
А3	1/4	1/9	1	1/2	1/5	0.038
А4	1/4	1/8	2	1	1/4	0.053
А5	4	1/4	5	4	1	0.224

Определена сумма каждого столбца матрицы суждений.

$S_{1К3} = 1 + 8 + 1/4 + 1/4 + 4 = 13.5$;

$S_{2\ К3} = 1/8 + 1 + 1/9 + 1/8 + 1/4 = 1.611$;

$S_{3\ К3} = 4 + 9 + 1 + 2 + 5 = 21$;

$S_{4\ К3} = 4 + 8 + 1/2 + 1 + 4 = 17.5$;

$S_{5\ К3} = 1/4 + 4 + 1/5 + 1/4 + 1 = 5.7$.

Затем полученный результат умножен на компоненту нормализованного вектора приоритетов.

$P_{1\ K3} = S_1 \times W_{3\ K31} = 1.444$;

$P_{2\ K3} = S_2 \times W_{3\ K32} = 0.931$;

$P_{3\ K3} = S_3 \times W_{3\ K33} = 0.798$;

$P_{4\ K3} = S_4 \times W_{3\ K34} = 0.927$;

$P_{5\ K3} = S_5 \times W_{3\ K35} = 1.277$.

Найдем пропорциональность предпочтений.

$\lambda_{max\ K3} = P_{1K3} + P_{2K3} + P_{3K3} + P_{4K3} + P_{5K3} = 5.377$.

Отклонение от согласованности выражается индексом согласованности.

$ИС_{\ K3} = (\lambda_{max\ K3} - n)/(n - 1) = 0.094$.

Найдено отношение согласованности ОС.

$ОС_{\ K3} = ИС/СИ = 0.084$.

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

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

Таблица 3.6.5 – Матрица сравнения по критерию 4.

К4	А1	А2	А3	А4	А5	W_{3K4Y}
А1	1	4	8	1/2	1	0.261
А2	1/4	1	4	1/3	1/3	0.097
А3	1/8	1/4	1	1/8	1/8	0.033
А4	2	3	8	1	1	0.326
А5	1	3	8	1	1	0.283

Определена сумма каждого столбца матрицы суждений.

$S_{1K4} = 1 + 1/4 + 1/8 + 2 + 1 = 4.375$;

$S_{2K4} = 4 + 1 + 1/4 + 3 + 3 = 11.25$;

$S_{3K4} = 8 + 4 + 1 + 8 + 8 = 29$;

$S_{4K4} = 1/2 + 1/3 + 1/8 + 1 + 1 = 2.958$;

$S_{5K4} = 1 + 1/3 + 1/8 + 1 + 1 = 3.458$.

Затем полученный результат умножен на компоненту нормализованного вектора приоритетов.

$P_{1K4} = S_1 \text{ x } W_{3\ K41} = 1.142$;

$P_{2K4} = S_2 \text{ x } W_{3\ K42} = 1.091$;

$P_{3K4} = S_3 \text{ x } W_{3\ K43} = 0.957$;

$P_{4K4} = S_4 \text{ x } W_{3\ K44} = 0.964$;

$P_{5K4} = S_5 \text{ x } W_{3\ K45} = 0.979$.

Найдена пропорциональность предпочтений.

$\lambda_{max\ K4} = P_{1K4} + P_{2K4} + P_{3K4} + P_{4K4} + P_{5K4} = 5.133$.

Отклонение от согласованности выражается индексом согласованности.

$ИС_{\ K4} = (\lambda_{max\ K4} - n)/(n - 1) = 0.033$.

Найдено отношение согласованности ОС.

$ОС_{\ K4} = ИС/СИ = 0.029$.

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

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

Таблица 3.6.6 – Матрица сравнения по критерию 5.

К5	А1	А2	А3	А4	А5	W_{3K5Y}
А1	1	5	8	9	2	0.47
А2	1/5	1	4	5	1/3	0.134
А3	1/8	1/4	1	2	1/8	0.048
А4	1/9	1/5	1/2	1	1/8	0.034
А5	1/2	3	8	8	1	0.314

Определена сумма каждого столбца матрицы суждений.

$S_{1K5} = 1 + 1/5 + 1/8 + 1/9 + 1/2 = 1.936$;

$S_{2K5} = 5 + 1 + 1/4 + 1/5 + 3 = 9.45$;

$S_{3K5} = 8 + 4 + 1 + 1/2 + 8 = 21.5$;

$S_{4K5} = 9 + 5 + 2 + 1 + 8 = 25$;

$S_{5K5} = 2 + 1/3 + 1/8 + 1/8 + 1 = 3.583$.

Затем полученный результат умножен на компоненту нормализованного вектора приоритетов.

$P_{1K5} = S_1 \text{ x } W_{3\ K41} = 0.91;$

$P_{2K5} = S_2 \text{ x } W_{3\ K42} = 1.266;$

$P_{3K5} = S_3 \text{ x } W_{3\ K43} = 1.032;$

$P_{4K5} = S_1 \text{ x } W_{3\ K44} = 0.85;$

$P_{5K5} = S_1 \text{ x } W_{3\ K45} = 1.125.$

Найдена пропорциональность предпочтений.

$\lambda_{max\ K5} = P_{1K5} + P_{2K5} + P_{3K5} + P_{4K5} + P_{5K5} = 5.183.$

Отклонение от согласованности выражается индексом согласованности.

$ИС_{K5} = (\lambda_{max\ K5} - n)/(n - 1) = 0.046.$

Найдено отношение согласованности ОС.

$ОС_{K5} = ИС/СИ = 0.041.$

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

3.7 Синтез альтернатив

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

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

$W_{2i} = (Y_{21}=0.245; Y_{22}=0.135; Y_{23}=0.093; Y_{24}=0.446; Y_{25}=0.081);$

$W_{3K1Y} = (Y_{311}=0.195; Y_{312}=0.279; Y_{313}=0.04; Y_{314}=0.408; Y_{315}=0.078);$

$W_{3K2Y} = (Y_{321}=0.074; Y_{322}=0.038; Y_{323}=0.415; Y_{324}=0.209; Y_{325}=0.264);$

$W_{3K3Y} = (Y_{331}=0.107; Y_{332}=0.578; Y_{333}=0.038; Y_{334}=0.053; Y_{335}=0.224);$

$W_{3K4Y} = (Y_{341}=0.261; Y_{342}=0.097; Y_{343}=0.033; Y_{344}=0.326; Y_{345}=0.283);$

$W_{3K5Y} = (Y_{351}=0.47; Y_{352}=0.134; Y_{353}=0.048; Y_{354}=0.034; Y_{355}=0.314).$

Приоритеты альтернатив получены следующим образом:

$W_1 = W_{21} \times W_{3K11} + W_{22} \times W_{3K21} + W_{23} \times W_{3K31} + W_{24} \times W_{3K41} + W_{25} \times W_{3K51} = 0.222$.

$W_2 = W_{21} \times W_{3K12} + W_{22} \times W_{3K22} + W_{23} \times W_{3K32} + W_{24} \times W_{3K42} + W_{25} \times W_{3K52} = 0.181$.

$W_3 = W_{21} \times W_{3K13} + W_{22} \times W_{3K23} + W_{23} \times W_{3K33} + W_{24} \times W_{3K43} + W_{25} \times W_{3K53} = 0.088$.

$W_4 = W_{21} \times W_{3K14} + W_{22} \times W_{3K24} + W_{23} \times W_{3K34} + W_{24} \times W_{3K44} + W_{25} \times W_{3K54} = 0.281$.

$W_5 = W_{21} \times W_{3K15} + W_{22} \times W_{3K25} + W_{23} \times W_{3K35} + W_{24} \times W_{3K45} + W_{25} \times W_{3K55} = 0.227$.

Таким образом, приоритеты альтернатив равны:

альтернатива А1 (МГТУ имени Н.Э. Баумана) - W_1 приоритет равен = 0.222;

альтернатива А2 (ВШЭ)- W_2 приоритет равен = 0.181;

альтернатива А3 (МАИ) - W_3 приоритет равен = 0.088;

альтернатива А4 (НИЯУ МИФИ) – W_4 приоритет равен = 0.281;

альтернатива А5 (МФТИ) - W_5 приоритет равен = 0.227.

3.8 Вывод

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

3.9 Результаты работы программы

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

Матрица парного сравнения критериев

	0	1	2	3	4
0	1	2	2	0.5	4
1	0.5	1	2	0.2	2
2	0.5	0.5	1	0.125	2
3	2	5	8	1	2
4	0.25	0.5	0.5	0.5	1

Рисунок 3.9.1 – Матрица парного сравнения критериев

Матрица сравнения по критерию 1 (проходной балл)

	0	1	2	3	4
0	1	0.5	8	0.25	2
1	2	1	6	0.5	2
2	0.125	0.167	1	0.2	0.167
3	4	2	5	1	2
4	0.5	0.5	0.167	0.5	1

Рисунок 3.9.2 – Сгенерированная матрица для первого критерия

Матрица сравнения по критерию 2 (количество бюджетных мест)

	0	1	2	3	4
0	1	5	0.167	0.2	0.2
1	0.2	1	0.125	0.2	0.25
2	6	8	1	2	2
3	5	5	0.5	1	0.5
4	5	4	0.5	2	1

Рисунок 3.9.3 – Сгенерированная матрица для второго критерия

Матрица сравнения по критерию 3 (стоимость обучения)

	0	1	2	3	4
0	1	0.125	4	4	0.25
1	8	1	9	8	4
2	0.25	0.111	1	0.5	0.2
3	0.25	0.125	2	1	0.25
4	4	0.25	5	4	1

Рисунок 3.9.4 – Сгенерированная матрица для третьего критерия

Матрица сравнения по критерию 4 (рейтинг университета)

	0	1	2	3	4
0	1	4	8	0.5	1
1	0.25	1	4	0.333	0.333
2	0.125	0.25	1	0.125	0.125
3	2	3	8	1	1
4	1	3	8	1	1

Рисунок 3.9.5 – Сгенерированная матрица для четвёртого критерия

Матрица сравнения по критерию 5 (расстояние до общежития)

	0	1	2	3	4
0	1	5	8	9	2
1	0.2	1	4	5	0.333
2	0.125	0.25	1	2	0.125
3	0.111	0.2	0.5	1	0.125
4	0.5	3	8	8	1

Рисунок 3.9.6 – Сгенерированная матрица для пятого критерия

```
Приоритеты альтернатив получены следующим образом:
W1 = W21 * W3K11 + W22 * W3K21 + W23 * W3K31 + W24 * W3K41 + W25 * W3K51 = 0.222
W2 = W21 * W3K12 + W22 * W3K22 + W23 * W3K32 + W24 * W3K42 + W25 * W3K52 = 0.181
W3 = W21 * W3K13 + W22 * W3K23 + W23 * W3K33 + W24 * W3K43 + W25 * W3K53 = 0.088
W4 = W21 * W3K14 + W22 * W3K24 + W23 * W3K34 + W24 * W3K44 + W25 * W3K54 = 0.281
W5 = W21 * W3K15 + W22 * W3K25 + W23 * W3K35 + W24 * W3K45 + W25 * W3K55 = 0.227
Таким образом, приоритеты альтернатив равны:
альтернатива A1 - W1 приоритет равен 0.222
альтернатива A2 - W2 приоритет равен 0.181
альтернатива A3 - W3 приоритет равен 0.088
альтернатива A4 - W4 приоритет равен 0.281
альтернатива A5 - W5 приоритет равен 0.227
PS C:\python_projects>
```

Рисунок 3.9.7 – Результат работы программы

3.10 Заключение

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

4 ГРАФИЧЕСКИЙ МЕТОД

4.1 Введение

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

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

4.2 Постановка задачи

Решить задачу линейного программирования с двумя переменными графическим методом.

4.3 Данные индивидуального варианта

$$f(x) = -x_1 + 2x_2 \to min/max$$

$$\begin{cases} 5x_1 - 2x_2 \le 7 \\ -x_1 + 2x_2 \le 5 \\ x_1 + x_2 \ge 6 \\ x_1, x_2 \ge 0 \end{cases}$$

4.4 Подготовка данных

В среде Microsoft Excel добавим 4 столбца:

1. x_1 – значения от 0 до 10 с шагом 0,5;
2. $x_2 = (\dots)$ – значения ограничения $(5x_1 - 2x_2 \leq 7)$;
3. $x_2 = (\dots)$ – значения ограничения $(-x_1 + 2x_2 \leq 5)$;
4. $x_2 = (\dots)$ – значения ограничения $(x_1 + x_2 \geq 6)$;
5. $x_2 = (\dots)$ – значения целевой функции при условии $f(x) = 0$.

Таблица 4.4.1 – Данные для графика

x_1	$x_2 = \frac{5x_1 - 7}{2}$	$x_2 = \frac{x_1 + 5}{2}$	$x_2 = -x_1 + 6$	$x_2 = \frac{x_1}{2}$
0	-3,50	2,50	6,00	0,00
0,5	-2,25	2,75	5,50	0,00
1	-1,00	3,00	5,00	0,00
1,5	0,25	3,25	4,50	0,00
2	1,50	3,50	4,00	0,00
2,5	2,75	3,75	3,50	0,00
3	4,00	4,00	3,00	0,00
3,5	5,25	4,25	2,50	0,00
4	6,50	4,50	2,00	0,00
4,5	7,75	4,75	1,50	0,00
5	9,00	5,00	1,00	0,00
5,50	10,25	5,25	0,50	0,00
6,00	11,50	5,50	0,00	0,00
6,50	12,75	5,75	-0,50	0,00
7,00	14,00	6,00	-1,00	0,00
7,50	15,25	6,25	-1,50	0,00
8,00	16,50	6,50	-2,00	0,00
8,50	17,75	6,75	-2,50	0,00
9,00	19,00	7,00	-3,00	0,00
9,50	20,25	7,25	-3,50	0,00
10,00	21,50	7,50	-4,00	0,00

4.5 Построение графика

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

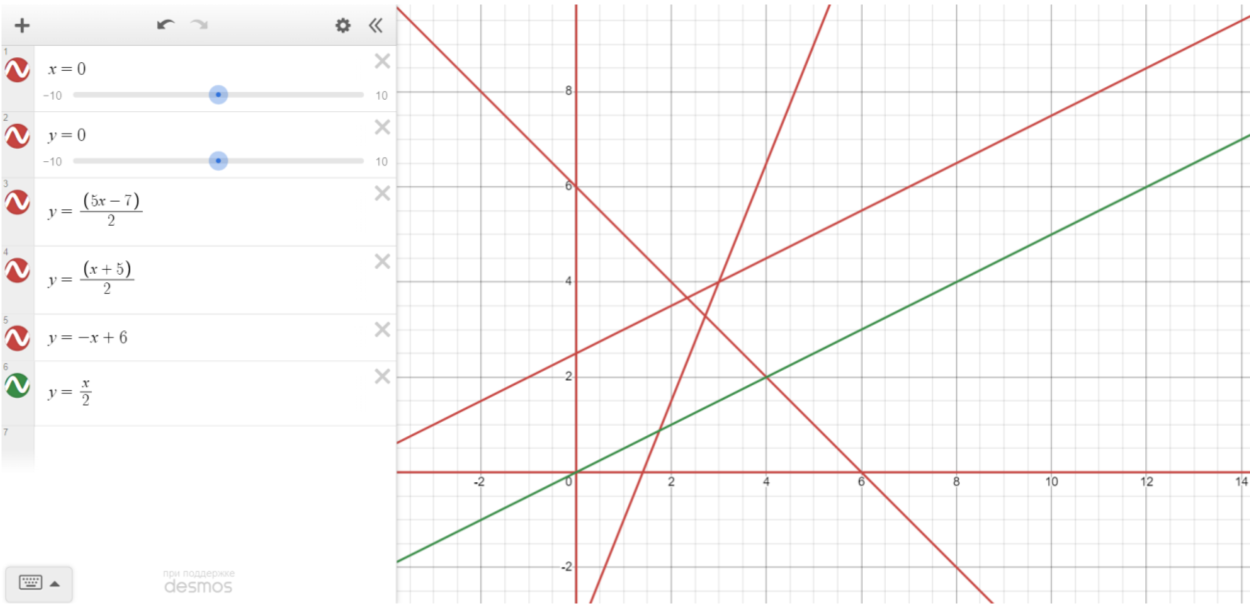

Рисунок 4.5.1 – Построение графиков по данным

4.6 Выделение области допустимых решений

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

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

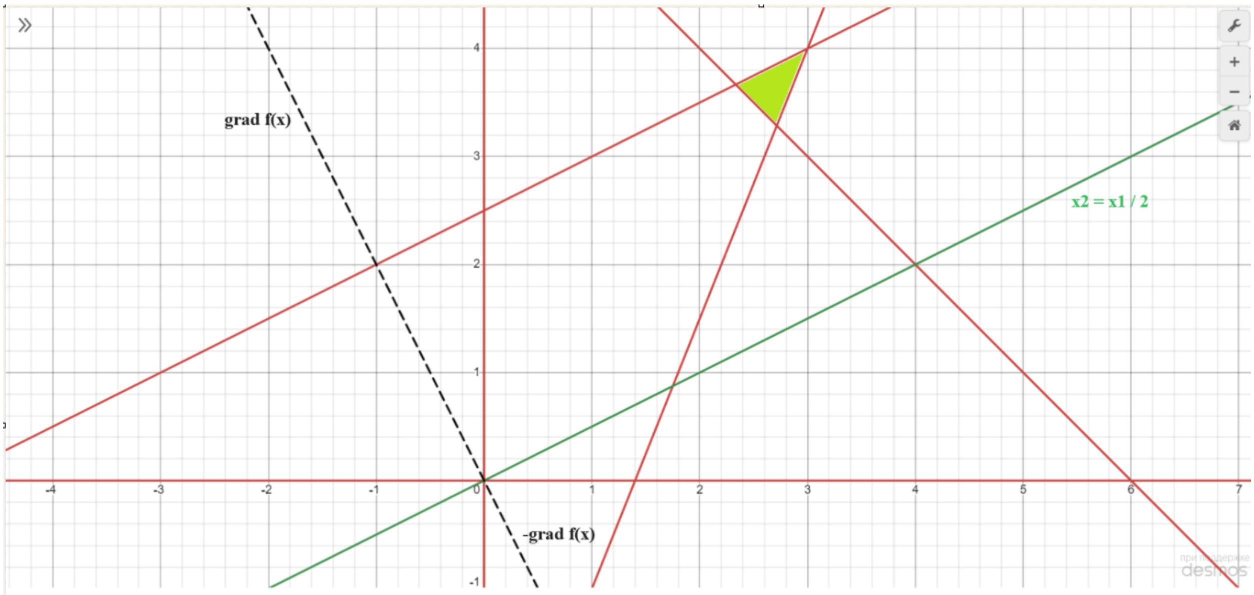

Рисунок 4.6.1 – Выделение области допустимых решений

4.7 Максимум функции

Для нахождения максимума функции найдем её градиент по формуле 4.1:

$$\overline{gradf(x)} = \left\{\frac{df(x)}{dx_1}, \frac{df(x)}{dx_2}\right\} \tag{4.1}$$

Для нахождения минимума функции найдем её градиент по формуле 4.1:

$$-\overline{gradf(x)} = \left\{-\frac{df(x)}{dx_1}, -\frac{df(x)}{dx_2}\right\} \tag{4.2}$$

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

Теперь начинаем мысленно сдвигать прямую целевой функции в направлении градиента, и определяем последнюю точку ОДР, которая лежит на пути прямой. На пути прямой лежит отрезок с началом в точке $(\frac{7}{3};\frac{11}{3})$ и конце в точке (3;4).

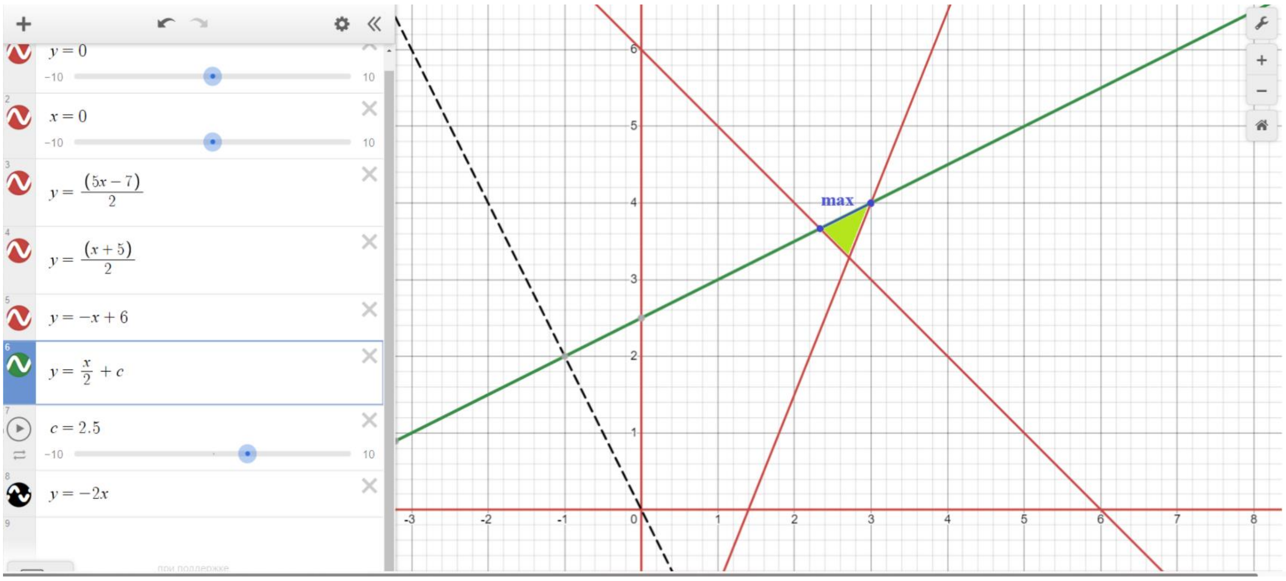

Рисунок 4.7.1 – Точка максимума функции

Найдем значение функции в точке максимума.

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

$$\begin{cases} 5*3-2*4 \le 7 \\ -3+2*4 \le 5 \\ 3+4 \ge 6 \\ 3{,}4 \ge 0 \end{cases}$$

Получим значение равное F(x)max = -3 + 2*4 = 5. Убедимся, что для любой точки, принадлежащей отрезку, максимальное значение функции остаётся неизменным. Подставим координаты начала отрезка: F(x)max = $-\frac{7}{3}+2*\frac{11}{3}=$ 5.

4.8 Минимум функции

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

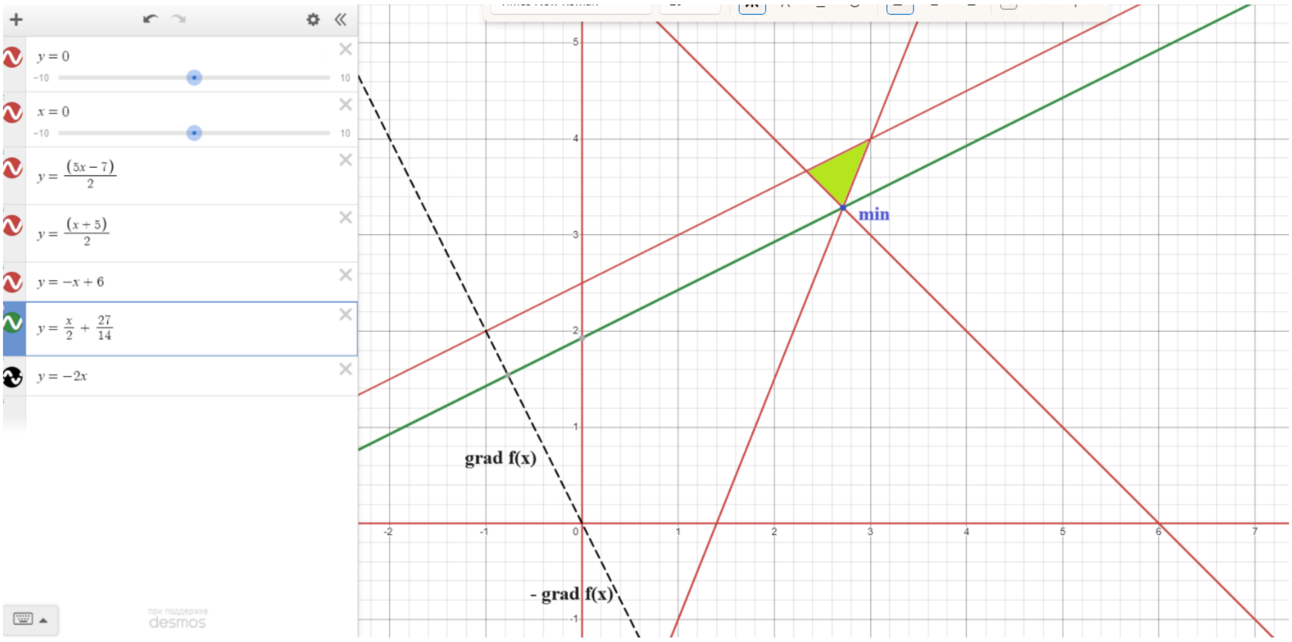

Рисунок 4.8.1 – Точка минимума функции

Найдем координаты точки минимума:

$$\frac{5x_1-7}{2} = -x_1 + 6 => x_1 = \frac{19}{7};\ x_2 = -\frac{19}{7} + 6 = \frac{23}{7}$$

В результате получим точку с координатами $(\frac{19}{7}; \frac{23}{7})$. Найдем значение функции в этой точке.

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

$$\begin{cases} 5*\frac{19}{7} - 2*\frac{23}{7} \le 7 \\ -\frac{19}{7} + 2*\frac{23}{7} \le 5 \\ \frac{19}{7} + \frac{23}{7} \ge 6 \\ \frac{19}{7}, \frac{23}{7} \ge 0 \end{cases} => \begin{cases} 7 \le 7 \\ 3\frac{6}{7} \le 5 \\ 6 \ge 6 \\ \frac{19}{7}, \frac{23}{7} \ge 0 \end{cases}$$

Получим результат F(x)min = $-\frac{19}{7} + 2*\frac{23}{7} = 3\frac{6}{7}$

Ответ:

F(x)max = 5.

F(x)min = $3\frac{6}{7}$.

4.9 Заключение

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

5 СИМПЛЕКСНЫЙ МЕТОД

5.1 Введение

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

5.2 Постановка задачи

Вариант №13

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

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

Таблица 5.2.1. Исходные данные задачи

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

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

5.3 Математическая модель задачи

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

Ограничения задачи:

$$\begin{cases} 0.2x1 + 0.1x2 \leq 100 \\ 0.75x1 + 0.2x2 \leq 150 \\ 0.15x1 + 0.2x2 \leq 100 \\ 0.15x1 + 0.25x2 \leq 150 \end{cases}$$

Таким образом, переходим к задаче линейного программирования:

$$f(x) = 1.4x1 + 0.9x2 \to max$$

$$\begin{cases} 0.2x1 + 0.1x2 \leq 100 \\ 0.75x1 + 0.2x2 \leq 150 \\ 0.15x1 + 0.2x2 \leq 100 \\ 0.15x1 + 0.25x2 \leq 150 \\ x_i \geq 0, i \in [1, 2] \end{cases}$$

Приведем задачу к канонической форме. Для этого в левые части ограничений вводим дополнительные переменные: x3 ≥ 0. Эти переменные выбираются так, чтобы они обращали неравенства в равенства.

$$\begin{cases} 0.2x1 + 0.1x2 + x3 = 100 \\ 0.75x1 + 0.2x2 + x4 = 150 \\ 0.15x1 + 0.2x2 + x5 = 100 \\ 0.15x1 + 0.25x2 + x6 = 150 \\ x_i \geq 0, i \in [1, 6] \end{cases}$$

$$f(x) = 1.4x1 + 0.9x2 + 0x3 + 0x4 + 0x5 + 0x6$$

5.4 Решение задачи

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

$$\mathrm{A1x1\ +\ A2x2\ + A3x3\ +\ A4x4\ +\ A5x5\ +\ A6x6 =\ A0,}$$

$$A1 = \begin{pmatrix} 0.2 \\ 0.75 \\ 0.15 \\ 0.15 \end{pmatrix}, A2 = \begin{pmatrix} 0.1 \\ 0.2 \\ 0.2 \\ 0.25 \end{pmatrix}, A3 = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}, A4 = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}, A5 = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix}, A6 = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}$$

$$A0 = \begin{pmatrix} 100 \\ 150 \\ 100 \\ 150 \end{pmatrix}$$

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

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

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

$$\mathrm{A3x3} + \mathrm{A4x4} + A5x5 + A6x6 = \mathrm{A0},$$

Которому соответствует первоначальный опорный план

$$\mathrm{x}^{(0)} = (\mathrm{x1}, \mathrm{x2}, \mathrm{x3}, \mathrm{x4}, \mathrm{x5}, \mathrm{x6}) == (0,0,100,150,100,150),$$

$$f\left(x^{(0)}\right) = 0.$$

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

$$\overline{C_B} = (c3, c4, c5, c6)^T = (0,0,0,0)^T.$$

В левый столбец Таблицы 5.4.1 запишем переменные $x3$, $x4$, x5, x6, образующие базис, в верхней строке – небазисные переменные $x1$, $x2$. В строке c_j запишем коэффициенты целевой функции, соответствующие небазисным переменным c1 = 1.4, c2 = 0.9. В столбце $\overline{C_B}$ запишем коэффициенты целевой функции, соответствующие базисным переменным Столбец, определяемый переменной $x1$, состоит из коэффициентов вектора $\overline{A_1}$. Аналогично, столбец, определяемый переменной $x2$, состоит из коэффициентов вектора $\overline{A_2}$. Крайний правый столбец заполняется элементами столбца $\overline{A_0}$, в нем же в результате вычислений получаем оптимальный план.

Заполнение f-строки (Таблица 5.4.2). Найдем относительные оценки Δ1, Δ2 и значение целевой функции Q.

$$\Delta_1 = (\overline{C_B} * \overline{A_1}) - c_1 = 0 * 0.2 + 0 * 0.75 + 0 * 0.15 + 0 * 0.15 - 1.4 = -1.4;$$

$$\Delta_2 = (\overline{C_B} * \overline{A_2}) - c_2 = 0 * 0.1 + 0 * 0.2 + 0 * 0.2 + 0 * 0.25 - 0.9 = -0.9;$$

$$\mathrm{Q} = (\overline{C_B} * \overline{A_0}) = 0 * 100 + 0 * 150 + 0 * 100 + 0 * 150 = 0.$$

Таблица 5.4.1 – Начальная симплекс-таблица задачи о максимальном доходе

	c_j	1.4	0.9	
$\overline{C_B}$		X1	X2	$\overline{A_0}$
0	X3	0.2	0.1	100
0	X4	0.75	0.2	150
0	X5	0.15	0.2	100
0	X6	0.15	0.25	150
	f			
		Δ_1	Δ_2	Q

Таблица 5.4.2 – Заполнение f-строки

	c_j	1.4	0.9		
$\overline{C_B}$		X1	X2	$\overline{A_0}$	
0	X3	0.2	0.1	100	100 / 0.2 = 500
0	X4	0.75	0.2	150	150 / 0.75 = 200 min
0	X5	0.15	0.2	100	100 / 0.15 = 666.66
0	X6	0.15	0.25	150	150 / 0.15 = 1000
	f	-1.4	-0.9	0	
		Δ_1	Δ_2	Q	

Для оптимальности опорного решения в задаче на максимум требуется выполнение неотрицательности всех относительных оценок Δi ≥ 0. Так как оценки Δ1= −1.4, Δ2= −0.9 в f-строке отрицательны, то это свидетельствуют о возможности улучшения полученного решения.

Наибольшая по модулю отрицательная оценка Δ1 = −1.4. В базис будет включена соответствующая ей небазисная переменная $x1$. Составим отношения свободных членов к положительным элементам разрешающего столбца. Данные отношения приведены справа от таблицы. Наименьшему частному соответствует строка с переменной $x4$. Эта переменная исключается из базиса. В Таблице 5.4.2

разрешающий столбец и разрешающая строка выделены. Разрешающим элементом является число $a21 = 0.75$.

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

Таблица 5.4.3 – Новая симплекс-таблица

	c_j	0	0.9	
$\overline{C_B}$		X4	X2	$\overline{A_0}$
0	X3			
1.4	X1	4/3		
0	X5			
0	X6			
	f			
		Δ_1	Δ_2	Q

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

Таблица 5.4.4 – Симплекс преобразования

	c_j	0	0.9	
$\overline{C_B}$		X4	X2	$\overline{A_0}$
0	X3	-4/15		
1.4	X1	4/3	4/15	200
0	X5	-0.2		
0	X6	-0.2		
	f	28/15		
		Δ_1	Δ_2	Q

Таблица 5.4.5 – Итерация 0

	c_j	0	0.9		
$\overline{C_B}$		X4	X2	$\overline{A_0}$	
0	X3	-4/15	7/150	60	60 / (7/150) = 1285.7
1.4	X1	4/3	4/15	200	200 / (4/15) = 750
0	X5	-0.2	4/25	70	70 / (4/25) = 437.5 min
0	X6	-0.2	21/100	120	120 / (21/100) = 571.0 = 571.4
	f	28/15	-79/150	280	

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

$$a_{12} = \frac{(0.1 * 0.75) - (0.2 * 0.2)}{0.75} = \frac{7}{150};\ a_{13} = \frac{(100 * 0.75) - (150 * 0.2)}{0.75} = 60;$$

$$a_{32} = \frac{(0.2 * 0.75) - (0.15 * 0.2)}{0.75} = \frac{4}{25};\ a_{33} = \frac{(100 * 0.75) - (0.15 * 150)}{1.3} = 70;$$

$$a_{42} = \frac{(0.25 * 0.75) - (0.15 * 0.2)}{0.75} = \frac{21}{100};\ a_{43} = \frac{(150 * 0.75) - (150 * 0.15)}{0.75}$$
$$= 120;$$

$$\Delta_2 = \frac{(-0.9 * 0.75) - (-1.4\ *\ 0.2)}{0.75} = -\frac{79}{150};$$

Базисное решение, которое дает последняя таблица

$$x^{(1)} = (x1, x2, x3, x4, x5, x6) = (200, 0,60,0,70,120),$$

$$f\left(x^{(1)}\right) = (\overline{C_B} * \overline{A_0}) = 0 * 60 + 1.4 * 200 + 0 * 70 + 0 * 120 = 280.$$

Это решение не является оптимальным, так как в f-строке имеется отрицательная оценка Δ2. Наибольшая по модулю отрицательная оценка Δ2 = $-\frac{79}{150}$. В базис будет включена соответствующая ей небазисная переменная $x2$. Составим отношения свободных членов к положительным элементам разрешающего столбца. Наименьшему частному соответствует строка с переменной $x5$, которая исключается из базиса. В Таблице 5.4.5 разрешающий столбец и разрешающая строка выделены. Разрешающим элементом является число $a32 = \frac{4}{25}$.

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

Таблица 5.4.6 – Новая симплекс-таблица

	c_j	0	0	
$\overline{C_B}$		X4	X5	$\overline{A_0}$
0	X3			
1.4	X1			
0.9	X2		25/4	
0	X6			
	f			
		Δ_1	Δ_2	Q

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

Таблица 5.4.7 – Симплекс преобразования

	c_j	0	0	
$\overline{C_B}$		X4	X5	$\overline{A_0}$
0	X3		-7/24	
1.4	X1		-5/3	
0.9	X2	-5/4	25/4	437.5
0	X6		-21/16	
	f		79/24	
		Δ_1	Δ_2	Q

Таблица 5.4.8 – Итерация 1

	c_j	0	0	
$\overline{C_B}$		X4	X5	$\overline{A_0}$
0	X3	-5/24	-7/24	475/12
1.4	X1	5/3	-5/3	250/3
0.9	X2	-5/4	25/4	437.5
0	X6	1/16	-21/16	225/8
	f	29/24	79/24	6125/12
		Δ_1	Δ_2	Q

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

$$a_{11} = \frac{\left(-\frac{4}{15} * \frac{4}{25}\right) - \left(-0.2 * \frac{7}{150}\right)}{\frac{4}{25}} = -\frac{5}{24};$$

$$a_{13} = \frac{\left(60 * \frac{4}{25}\right) - \left(70 * \frac{7}{150}\right)}{\frac{4}{25}} = \frac{475}{12};$$

$$a_{21} = \frac{\left(\frac{4}{3} * \frac{4}{25}\right) - \left(\frac{4}{15} * -0.2\right)}{\frac{4}{25}} = \frac{5}{3};$$

$$a_{23} = \frac{\left(200 * \frac{4}{25}\right) - \left(\frac{4}{15} * 70\right)}{\frac{4}{25}} = \frac{250}{3};$$

$$a_{41} = \frac{\left(-0.2 * \frac{4}{25}\right) - \left(\frac{21}{100} * -0.2\right)}{\frac{4}{25}} = \frac{1}{16};$$

$$a_{43} = \frac{\left(120 * \frac{4}{25}\right) - \left(\frac{21}{100} * 70\right)}{\frac{4}{25}} = \frac{225}{8};$$

$$\Delta_1 = \frac{\left(\frac{28}{15} * \frac{4}{25}\right) - \left(-\frac{79}{150} * -0.2\right)}{\frac{4}{25}} = \frac{29}{24};$$

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

Базисное решение, которое дает последняя таблица

$$x^{(2)} = (x1, x2, x3, x4, x5, x6) = \left(\frac{250}{3}, 437.5, \frac{475}{12}, 0, 0, \frac{225}{8}\right),$$

$$f\left(x^{(2)}\right) = (\overline{C_B} * \overline{A_0}) = 0 * \frac{475}{12} + 1.4 * \frac{250}{3} + 0.9 * 437.5 + 0 * \frac{225}{8} = \frac{6125}{12} = 510.417.$$

Проверим решение по «правилу прямоугольника».

$$f_{max} = Q = (\overline{C_B} * \overline{A_0}) = \frac{(280 * \frac{4}{25}) - (-\frac{79}{150} * 70)}{\frac{4}{25}} = \frac{6125}{12} = 510.417.$$

Таким образом, кондитерский цех должен выпекать $x1 = \frac{250}{3}$ кг печенья первого сорта и 437.5 кг печенья второго сорта. Тогда общая стоимость будет наибольшей и кондитерская получит прибыль от продажи 510.417 [ден.ед].

5.5 Пример работы программы

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

```
TPR_PRACT5.csv
1   f(x) = 1.4x1 + 0.9x2
2   0.2x1 + 0.1x2 <= 100
3   0.75x1 + 0.2x2 <= 150
4   0.15x1 + 0.2x2 <= 100
5   0.15x1 + 0.25x2 <= 150
```

Рисунок 5.5.1 – Условия задачи в csv файле

```
Переходим к задаче линейного программирования:
f(x) = 1.4x1 + 0.9x2
{ 0.2x1 + 0.1x2 <= 100
{ 0.75x1 + 0.2x2 <= 150
{ 0.15x1 + 0.2x2 <= 100
{ 0.15x1 + 0.25x2 <= 150
```

Рисунок 5.5.2 – Обработка входных данных

```
                                                                Итерация №0
                    Cj                  0                   0.9
Cv                                      x4                  x2                  A0
0                   x3                  -0.2667             0.0467              60.0
1.4                 x1                  1.3333              0.2667              200.0
0                   x5                  -0.2                0.16                70.0
0                   x6                  -0.2                0.21                120.0
                    f                   1.8667              -0.5267             280.0
                                                                Итерация №1
                    Cj                  0                   0
Cv                                      x4                  x5                  A0
0                   x3                  -0.2083             -0.2919             39.5687
1.4                 x1                  1.6667              -1.6669             83.3187
0.9                 x2                  -1.25               6.25                437.5
0                   x6                  0.0625              -1.3125             28.125
                    f                   1.2083              3.2919              510.4313
Решение найдено! Общая прибыль составила 510.4313 денежных единиц
```

Рисунок 5.5.3 – Выполнение двух итераций и найденное решение

5.6 Заключение

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

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

6 ДВОЙСТВЕННАЯ ЗАДАЧА

6.1 Введение

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

6.2 Постановка задачи

Вариант №13

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

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

Таблица 5.2.1. Исходные данные задачи

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

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

6.3 Математическая модель

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

Ограничения задачи:

$$\begin{cases} 0.2x1 + 0.1x2 \le 100 \\ 0.75x1 + 0.2x2 \le 150 \\ 0.15x1 + 0.2x2 \le 100 \\ 0.15x1 + 0.25x2 \le 150 \end{cases}$$

Таким образом, переходим к задаче линейного программирования:

$$f(x) = 1.4x1 + 0.9x2 \to max$$

$$\begin{cases} 0.2x1 + 0.1x2 \le 100 \\ 0.75x1 + 0.2x2 \le 150 \\ 0.15x1 + 0.2x2 \le 100 \\ 0.15x1 + 0.25x2 \le 150 \\ x_i \ge 0, i \in [1, 2] \end{cases}$$

6.4 Соответствующая исходной двойственная задача

Найдем соответствующую двойственную задачу. Введем вектор двойственных переменных размерности два $\overline{\text{y}} = (\text{y}_1, \text{y}_2)^{\text{T}}$. Соответствующие векторы и матрица ограничений имеет вид:

$$\overline{\text{c}} = (1.4, 0.9), \overline{b} = (100,150,100,150), A = \begin{pmatrix} 0.2 & 0.1 \\ 0.75 & 0.2 \\ 0.15 & 0.2 \\ 0.15 & 0.25 \end{pmatrix}, A^T = \begin{pmatrix} 0.2 & 0.75 & 0.15 & 0.15 \\ 0.1 & 0.2 & 0.2 & 0.25 \end{pmatrix}.$$

Запишем двойственную задачу. Найти минимум функции.

$$g(\overline{\text{y}}) = (\overline{b}, \overline{\text{y}}) = 100\text{y}_1 + 150\text{y}_2 + 100\text{y}_3 + 150\text{y}_4 \to min$$

При ограничениях:

$$\begin{pmatrix} 0.2 & 0.75 & 0.15 & 0.15 \\ 0.1 & 0.2 & 0.2 & 0.25 \end{pmatrix} \cdot \begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{pmatrix} \geq \begin{pmatrix} 1.4 \\ 0.9 \end{pmatrix}, \text{следовательно}$$

$$\begin{cases} 0.2y_1 + 0.75y_2 + 0.15y_3 + 0.15y_4 \geq 1.4, \\ 0.1y_1 + 0.2y_2 + 0.2y_3 + 0.25y_4 \geq 0.9, \\ yi \geq 0, \quad 1 \leq i \leq 4. \end{cases}$$

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 $f_{max} = 510.417$ тыс. ден. ед., оптимальный план $\overline{x^*} = (x1, x2, x3, x4, x5, x6) = \left(\frac{250}{3}, 437.5, \frac{475}{12}, 0, 0, \frac{225}{8}\right)$.

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:

$$\overline{x^*} = \overline{C_B} \cdot D^{-1},$$

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

В последней симплекс-таблице базисными переменными являются $x3$, $x1$, $x2$, $x6$. Соответствующие этим переменным векторы $\overline{A_3}, \overline{A_1}, \overline{A_2}, \overline{A_6}$ в разложении используются для формирования столбцов матрицы D.

$$\overline{A_3} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \overline{A_1} = \begin{pmatrix} 0.2 \\ 0.75 \\ 0.15 \\ 0.15 \end{pmatrix}, \overline{A_2} = \begin{pmatrix} 0.1 \\ 0.2 \\ 0.2 \\ 0.25 \end{pmatrix}, \overline{A_6} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix},$$

Тогда,

$$D = (\overline{A_3}, \overline{A_1}, \overline{A_2}, \overline{A_6}) = \begin{pmatrix} 1 & 0.2 & 0.1 & 0 \\ 0 & 0.75 & 0.2 & 0 \\ 0 & 0.15 & 0.2 & 0 \\ 0 & 0.15 & 0.25 & 1 \end{pmatrix}$$

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

$$\left(\begin{array}{cccc|cccc} 1 & 0.2 & 0.1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0.75 & 0.2 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0.15 & 0.2 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0.15 & 0.25 & 1 & 0 & 0 & 0 & 1 \end{array}\right)$$

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

Разделим вторую строку на 0.75;

$$\left(\begin{array}{cccc|cccc} 1 & 0.2 & 0.1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & \frac{4}{15} & 0 & 0 & \frac{4}{3} & 0 & 0 \\ 0 & 0.15 & 0.2 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0.15 & 0.25 & 1 & 0 & 0 & 0 & 1 \end{array}\right)$$

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

$$\left(\begin{array}{cccc|cccc} 1 & 0 & \frac{7}{150} & 0 & 1 & -\frac{4}{15} & 0 & 0 \\ 0 & 1 & \frac{4}{15} & 0 & 0 & \frac{4}{3} & 0 & 0 \\ 0 & 0 & 0.16 & 0 & 0 & -0.2 & 1 & 0 \\ 0 & 0 & 0.21 & 1 & 0 & -0.2 & 0 & 1 \end{array}\right)$$

разделим третью строку на 0.16;

$$\left(\begin{array}{cccc|cccc} 1 & 0 & \frac{7}{150} & 0 & 1 & -\frac{4}{15} & 0 & 0 \\ 0 & 1 & \frac{4}{15} & 0 & 0 & \frac{4}{3} & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & -1.25 & 6.25 & 0 \\ 0 & 0 & 0.21 & 1 & 0 & -0.2 & 0 & 1 \end{array}\right)$$

от первой строки отнимем третью строку, умноженную на $\frac{7}{150}$; от второй строки отнимем третью, умноженную на $\frac{4}{15}$; от четвертой строки отнимем третью строку, умноженную на 0.21;

$$\left(\begin{array}{cccc|cccc} 1 & 0 & 0 & 0 & 1 & -\frac{5}{24} & -\frac{7}{24} & 0 \\ 0 & 1 & 0 & 0 & 0 & \frac{5}{3} & -\frac{5}{3} & 0 \\ 0 & 0 & 1 & 0 & 0 & -1.25 & 6.25 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0.0625 & -1.3125 & 1 \end{array}\right)$$

Запишем обратную матрицу.

$$D^{-1} = (y_3^*, y_1^*, y_2^*, y_6^*) = \begin{pmatrix} 1 & -\frac{5}{24} & -\frac{7}{24} & 0 \\ 0 & \frac{5}{3} & -\frac{5}{3} & 0 \\ 0 & -1.25 & 6.25 & 0 \\ 0 & 0.0625 & -1.3125 & 1 \end{pmatrix}$$

Базисными переменными в симплекс-таблице являются $\overline{\text{C}_\text{Б}} = (0, 1.4, 0.9, 0)$, тогда

$$\overline{y^*} = (y_3^*, y_1^*, y_2^*, y_6^*) = \overline{\text{C}_\text{Б}} \cdot D^{-1} =$$

$$= \left(0; \left(0 * -\frac{5}{24} + 1.4 * \frac{5}{3} + 0.9 * -1.25 + 0 * 0.0625\right); \left(0 * -\frac{7}{24} + 1.4 * -\frac{5}{3} + 0.9 * 6.25 + 0 * -1.3125\right); 0\right) =$$

$$= (0; \frac{29}{24}; \frac{79}{24}; 0)$$

При этом минимальное значение целевой функции двойственной задачи

$$g_{min} = g\overline{(y^*)} = \left(\bar{b}, \overline{y^*}\right) = 100 * 0 + 150 * \frac{29}{24} + 100 * \frac{79}{24} + 150 * 0 = 510.417 \text{ тыс. ден. ед}$$

совпадает с максимальным значением f_{max} = 510.417 [тыс. ден.ед.] прямой задачи, что является результатом взаимодвойственности. Таким образом,

$$max f(\bar{x}) = min g(\bar{y}) = 510.417 \text{ [тыс. ден. ед.]}.$$

6.6 Вторая теорема двойственности

Для того, чтобы планы $\overline{\mathrm{x}^*} = (x_1^*, x_2^*, \dots, x_n^*)$ и $\overline{y^*} = (y_1^*, y_2^*, \dots, y_m^*)$ ЗЛП двойственной пары были оптимальными, необходимо и достаточно, чтобы эти планы удовлетворяли условиям дополняющей нежесткости.

$$\left\{x_j^*\left(\sum_{i=1}^{m} a_{ij}y_i^* - c_j\right) = 0, j = \overline{1,n}\right.$$

$$\left\{y_i^*\left(\sum_{i=1}^{n} a_{ij}x_j^* - b_i\right) = 0, i = \overline{1,m}\right.$$

Итак, имеем оптимальное решение прямой задачи: объем производства печенья первого сорта – $x1 = \frac{250}{3}$; объем производства печенья второго сорта – $x2 = 437.5$; максимальный доход от продажи $f_{max} = 510.417$ [тыс. ден.ед.]. Рассмотрим выполнение неравенств прямой задачи при подстановке $x1$, $x2$ в систему ограничений (Таблица 6.6.1).

Согласно Таблице 6.6.1 имеем следующую систему уравнений:

$$\begin{cases} 0.2y1 + 0.75y2 + 0.15y3 + 0.15y4 = 1.4 \\ 0.1y1 + 0.2y2 + 0.2y3 + 0.25y4 = 0.9 \\ y1 = 0 \\ y4 = 0 \end{cases}$$

Решим данную систему уравнений

$$(y1, y2, y3, y4) = (0, \frac{29}{24}, \frac{79}{24}, 0)$$

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

$$g\overline{(y^*)} = \left(\bar{b}, \overline{y^*}\right) = 100 * 0 + 150 * \frac{29}{24} + 100 * \frac{79}{24} + 150 * 0$$

$$= 510.417 \text{ тыс. ден. ед}$$

$$min\ g(\bar{y}) = 510.417\ [\text{тыс. ден. ед.}]$$

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

Таблица 6.6.1 – Выполнение неравенств прямой задачи

Ограничение	Расчет	Вывод
0.2x1 + 0.1x2 ≤ 100	$0.2*\frac{250}{3}$ + 0.1*437.5 <100 $60\frac{5}{12}$ < 100	Первое ограничение прямой задачи выполняется как строгое неравенство, остается спрос на печенье первого сорта. Значит, этот ресурс не является дефицитным и его оценка в оптимальном плане равна нулю ($y_1 = 0$).
0.75x1 + 0.2x2 ≤ 150	$0.75*\frac{250}{3}$+0.2*437.5 = 150 150=150	Второе ограничение прямой задачи выполняется как равенство. Это означает, что печенье первого сорта полностью используется в оптимальном плане, является дефицитным и его оценка согласно второй теоремы двойственности отлична от нуля ($y_2 \neq 0$).
0.15x1 + 0.2x2 ≤ 100	$0.15*\frac{250}{3}$+0.2*437.5 = 100 100=100	Третье ограничение прямой задачи выполняется как равенство. Это означает что печенье второго сорта полностью используется в оптимальном плане, является дефицитным и его оценка согласно второй теоремы двойственности отлична от нуля ($y_3 \neq 0$).
0.15x1 + 0.25x2 ≤ 150	$0.15*\frac{250}{3}$+0.25*437.5<150 $121\frac{7}{8}$ < 150	Четвёртое ограничение прямой задачи выполняется как строгое неравенство, остается спрос на печенье второго сорта. Значит, этот ресурс не является дефицитным и его оценка в оптимальном плане равна нулю ($y4 = 0$).
x1 ≥ 0	$\frac{250}{3} > 0$	Первое ограничение в двойственной задаче будет равенством 0.2y1 + 0.75y2 + 0.15y3 + 0.15y4 = 1.4
x2 ≥ 0	437.5 > 0	Второе ограничение в двойственной задаче будет равенством 0.1y1 + 0.2y2 + 0.2y3 + 0.25y4 = 0.9

6.7 Третья теорема двойственности

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

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

$$D^{-1} = (y_3^*, y_1^*, y_2^*, y_6^*) = \begin{pmatrix} 1 & -\frac{5}{24} & -\frac{7}{24} & 0 \\ 0 & \frac{5}{3} & -\frac{5}{3} & 0 \\ 0 & -1.25 & 6.25 & 0 \\ 0 & 0.0625 & -1.3125 & 1 \end{pmatrix}$$

Индексы базисных переменных оптимального плана:

$$\overline{A_0^*} = (x_3^*, x_1^*, x_2^*, x_6^*) = \begin{pmatrix} \frac{475}{12} \\ \frac{250}{3} \\ 437.5 \\ \frac{225}{8} \end{pmatrix}$$

Свободные члены неравенств (ограничений) прямой задачи:

$$\overline{\mathrm{A}_0} = (b_1, b_2, b_3, b_4) = \begin{pmatrix} 100 \\ 150 \\ 100 \\ 150 \end{pmatrix}$$

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

Ресурс 1 (Масло). Найдем нижнюю границу. В четвёртом столбце обратной матрицы один положительный элемент (1), ему соответствует индекс базисной переменной оптимального плана (150).

$$\Delta b_4^H = min\{150/1\} = 150$$

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

$$\Delta b_4^{\mathrm{B}} = +\infty$$

Таким образом, получаем $\Delta b_1 \in (150; +\infty)$.

Тогда первый ресурс может изменяться в интервале:

$$(b_4 - \Delta b_4^H, b_4 + \Delta b_4^B) = (150 - 150; 150 + \infty) = (0; +\infty)\text{ед.}$$

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

Ресурс 2 (Яйца). Рассматриваем третий столбец обратной матрицы, в котором один положительный элемент (6.25) и три отрицательных ($-\frac{7}{24}$, $-\frac{5}{3}$, -1.3125). Данным элементам соответствуют следующие индексы базисных переменных оптимального плана: положительного элемента – 100; для отрицательных – 100, 150,150.

Тогда находим нижнюю границу.

$$\Delta b_3^H = min\{100/6.25\} = 16$$

Найдем верхнюю границу.

$$\Delta b_3^{\text{В}} = \begin{cases} |max\{100\ x\ (24/7)\}| = \left|\frac{2400}{7}\right| = 342\frac{6}{7} \\ \left|\max\left\{150\ x\ \left(\frac{3}{5}\right)\right\}\right| = |90| = 90 \\ \left|\max\left\{150\ x\ \left(\frac{16}{21}\right)\right\}\right| = \left|\frac{800}{7}\right| = 114\frac{2}{7} \end{cases}$$

Выбираем наибольшее значение, равное $342\frac{6}{7}$.

Получаем $\Delta b_3 \in \left(16; 342\frac{6}{7}\right)$.

Тогда второй ресурс может изменяться в интервале:

$$(b_3 - \Delta b_3^H, b_3 + \Delta b_3^B) = \left(100 - 16; 100 + 342\frac{6}{7}\right) == \left(84; 442\frac{6}{7}\right) \text{ ед.}$$

Ресурс 3 (Сахар). Рассматриваем второй столбец обратной матрицы, в котором два положительных элемента ($\frac{5}{3}$, 0.0625). Данным элементам соответствуют индексы соответствующего базисного переменного оптимального плана – 150, 150.

Находим нижнюю границу.

$$\Delta b_2^H = \begin{cases} |min\{150\ x\ (3/5)\}| = |90| = 90 \\ |\min\{150\ x\ 16\}| = |2400| = 2400 \end{cases}$$

Выбираем наименьшее значение, равное 90.

Найдем верхнюю границу.

$$\Delta b_2^{\text{В}} = \begin{cases} |max\{100\ x\ (24/5)\}| = |480| = 480 \\ |\max\{100\ x\ 0.8\}| = |80| = 80 \end{cases}$$

Выбираем наибольшее значение, равное 480.

Тогда, получаем что $\Delta b_2 \in (90; 480)$.

Получаем интервал устойчивости оценок по отношению к третьему ограничению:

$$(b_2 - \Delta b_2^H, b_2 + \Delta b_2^B) = (150 - 90; 150 + 480) = (60; 630) \text{ ед.}$$

Ресурс 4 (Молоко). Найдем нижнюю границу. В первом столбце обратной матрицы один положительный элемент (1), ему соответствует индекс базисной переменной оптимального плана (100).

$$\Delta b_1^H = min\{100/1\} = 100$$

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

$$\Delta b_1^{\text{В}} = +\infty$$

Таким образом, получаем $\Delta b_1 \in (100; +\infty)$.

Тогда первый ресурс может изменяться в интервале:

$$(b_1 - \Delta b_1^H, b_1 + \Delta b_1^B) = (100 - 100; 100 + \infty) = (0; +\infty)\text{ед.}$$

Далее оценим влияние изменения объема ресурсов на величину максимальной стоимости продукции. Как известно, это дефицитные ресурсы $(y2, y3) = (\frac{29}{24}, \frac{79}{24})$. Введем верхние границы $\Delta b_2^{\text{В}}$ и $\Delta b_3^{\text{В}}$ в формулу:

$$\Delta G_{max}^i \approx y_i^* \times \Delta b_i$$

$$\Delta G_{max_2} = y_2 \times \Delta b_2^B = \frac{29}{24} \times 342\frac{6}{7} = 414$$

$$\Delta G_{max_3} = y_3 \times \Delta b_3^B = \frac{79}{24} \times 480 = 1580$$

Совместное влияние изменений этих ресурсов приводит к изменению максимальной стоимости продукции G_{max} на величину:

$$\Delta G_{max} = \Delta G_{max_1} + \Delta G_{max_2} = 414 + 1580 = 1994.28$$

Следовательно, оптимальное значение целевой функции при максимальном изменении ресурсов:

$$G_{max} \approx 1994.28 + 510.417 = 2504.70 \text{ [тыс. ден. ед./неделю]}$$

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

6.8 Результаты работы программы

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

```
Gmin is 510.4166666666667 by first_duality_theorem
Gmin is 510.4166666666667 by second_duality_theorem
Ресурс №1
b1 ∈ (150.0; inf)
1-й ресурс изменяется в интервале: (0.0; inf)
Ресурс №2
b2 ∈ (16.0; 342.85714285714283)
2-й ресурс изменяется в интервале: (84.0; 442.85714285714283)
Ресурс №3
b3 ∈ (90.00000000000001; 480.0)
3-й ресурс изменяется в интервале: (59.999999999999986; 630.0)
Ресурс №4
b4 ∈ (100.0; inf)
4-й ресурс изменяется в интервале: (0.0; inf)
ΔGmax2 = y2 * bB2 = 414.2857142857142
ΔGmax3 = y3 * bB3 = 1580.0000000000002
Совместное влияние изменений этих ресурсов приводит к изменению максимальной стоимости продукции Gmax на величину: 19
94.2857142857144
Следовательно, оптимальное значение целевой функции при максимальном изменении ресурсов: 2504.7023824057146
PS C:\python_projects>
```

Рисунок 6.8.1 –Теоремы двойственности

6.9 Заключение

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

7 ТРАНСПОРТНАЯ ЗАДАЧА

7.1 Введение

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

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

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

7.2 Постановка задачи

Вариант 13.

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

Таблица 7.2.1. Исходные данные задачи.

Потребители / Поставщики	**40**	**30**	**30**	**50**
60	2	3	5	1
70	3	4	9	4
20	2	5	2	5

Определить оптимальный план перевозок.

7.3 Математическая модель транспортной задачи

Математически задачу можно сформулировать следующим образом. Определить переменные x_{ij}, которые минимизируют суммарную стоимость перевозок.

$$f(x) = \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}x_{ij}$$

и удовлетворяют системе ограничений

a) $x_{i1} + x_{i2} + \cdots + x_{in} = a_i,\ i = 1,2,\ldots,m$ – с каждого пункта отправления груз должен быть вывезен полностью;

b) $x_{1j} + x_{2j} + \cdots + x_{mj} = b_j,\ j = 1,2,\ldots,n$ – потребитель должен получить ровно столько, сколько ему требуется;

c) $x_{ij} \geq 0,\ i = 1,2,\ldots,m; j = 1,2,\ldots,n$.

Транспортная задача является закрытой, т.к. $\sum_{i=0}^{3} a_i = 60 + 70 + 20 = 150$ и $\sum_{i=0}^{4} b_i = 40 + 30 + 30 + 50 = 150$.

7.4 Метод северо-западного угла

Заполним ячейку a_{11}. Т.к. потребности B_1 меньше запасов A_1 (40 < 60), то $x_{11} = 40$. Запасы первого поставщика не исчерпаны, поэтому происходит переход к ячейке a_{12}. Т.к. потребности B_2 больше запасов A_1 (30 > 20), то $x_{12} = 20$. Запасы первого поставщика исчерпаны, происходит переход к ячейки a_{22}. В составе заявки пункта B_2 остались неудовлетворёнными 10 единиц. Эти 10 единиц покроем за счёт пункта A_2. Далее перейдём к ячейки a_{23}. Запасов второго поставщика достаточно, значит $x_{23} = 30$. Запасы третьего потребителя удовлетворены, значит переходим к ячейки a_{24}. Так как потребность B_4 больше запасов A_2, (50 > 30) $x_{24} = 30$. Из запасов пункта A_3 выделим все доступные 20 единиц, чтобы удовлетворить запрос пункта B_4.

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

$f_0 = 2 * 40 + 3 * 20 + 4 * 10 + 9 * 30 + 4 * 30 + 5 * 20 = 670$ (единиц).

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

Таблица 7.4.1 – Метод северо-западного угла

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

7.5 Метод минимальной стоимости

Выбираем ячейку с минимальной стоимостью $c_{14} = 1$. Значение x_{14} определяется как минимальное из остатков запасов A_1 и потребностей потребителя B_4. Тогда $x_{14} = 50$. Мысленно вычёркиваем из таблицы столбец B_4, так как его запрос удовлетворён. Перейдём к ячейки a_{11} со стоимостью перевозки, равной 2. В рассматриваемую клетку запишем минимальное из остатков запасов A_1 и потребностей потребителя B_1. В данном случае запишем 10 – остаток запасов в пункте A_1, после чего мысленно вычеркнем строку A_1. Далее в ячейку a_{31} запишем минимум из потребностей B_1 (30) и запасов A_3 (20) - 20 и вычеркнем строку A_3. Следующая минимальная стоимость находится в ячейке a_{21}. Сюда запишем 10 и вычеркнем столбец B_1. В ячейку a_{22} запишем 30, так как потребность B_2 (30) меньше, чем запас A_2 (60). В последнюю оставшуюся ячейку a_{23} запишем 30 и зачеркнём соответствующий столбец B_3 и строку A_2. Все ресурсы израсходованы, а потребности удовлетворены.

Общая стоимость перевозок груза составляет:

$f_0 = 2 * 10 + 1 * 50 + 3 * 10 + 4 * 30 + 9 * 30 + 2 * 20 = 530$ (единиц).

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

Таблица 7.5.1 – Метод минимальной стоимости

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

7.6 Метод потенциалов

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

Таблица 7.6.1 – Метод северо-западного угла

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

Стоимость перевозок по этому плану:

$f_0 = 2 * 40 + 3 * 20 + 4 * 10 + 9 * 30 + 4 * 30 + 5 * 20 = 670$ (единиц).

Вычислим потенциалы u_i и v_j исходя из базисных переменных. Для их нахождения используем условие $u_i + v_j = c_{ij}$.

$$u_1 + v_1 = 2; u_1 + v_2 = 3; u_2 + v_2 = 4; u_2 + v_3 = 9;$$

$$u_2 + v_4 = 4; u_3 + v_4 = 5;$$

Считая $u_1 = 0$, имеем $u_1 = 0; v_1 = 2; v_2 = 3; u_2 = 1; v_3 = 8; v_4 = 3; u_3 = 2$.

Для каждой свободной клетки вычислим относительные оценки:

$$\Delta_{13} = c_{13} - (u_1 + v_3) = 5 - 0 - 8 = -3;$$

$$\Delta_{14} = c_{14} - (u_1 + v_4) = 1 - 0 - 3 = -2;$$

$$\Delta_{21} = c_{21} - (u_2 + v_1) = 3 - 1 - 2 = 0;$$

$$\Delta_{31} = c_{31} - (u_3 + v_1) = 2 - 2 - 2 = -2;$$

$$\Delta_{32} = c_{32} - (u_3 + v_2) = 5 - 2 - 3 = 0;$$

$$\Delta_{33} = c_{33} - (u_3 + v_3) = 2 - 2 - 8 = -8;$$

Условие оптимальности плана перевозок $\Delta_{ij} \geq 0$ не выполняется, поэтому построим замкнутый цикл пересчета и определим величины перераспределения груза.

Минимальной оценкой является $\Delta_{33} = -8$ для клетки (3,3).

Для определения количества груза λ подлежащего распределению, построим замкнутый цикл (указан стрелками) (Таблица 7.6.2). Одна из вершин цикла находится в незанятой клетке (3,3), которую отмечаем знаком « + ». Все остальные вершины цикла находятся в базисных клетках, с чередующимися знаками « − » и « + ». Найдем $\lambda = \min(30{,}20) = 20$, равное наименьшему из чисел, стоящих в отрицательных вершинах цикла. Значение λ записываем в незанятую клетку. Двигаясь далее по означенному циклу, вычитаем λ из объемов перевозок, расположенных в клетках, которые обозначены знаком « − », и прибавляем к объемам перевозок, находящихся в клетках, отмеченных знаком « + ». Элементы таблицы, не входящие в цикл, остаются без изменений. Таблица нового плана представлена в Таблице 7.6.3.

Таблица 7.6.2 – Таблица перерасчёта

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

Таблица 7.6.3 – Таблица нового плана

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

Стоимость перевозок по этому плану:

$f_1 = f_0 + \Delta_{33}\lambda = 670 - 8 * 20 = 510$ (ед.)

Вычислим потенциалы u_i и v_j исходя из базисных переменных. Для их нахождения используем условие $u_i + v_j = c_{ij}$.

$$u_1 + v_1 = 2; u_1 + v_2 = 3; u_2 + v_2 = 4; u_2 + v_3 = 9;$$

$$u_2 + v_4 = 4; u_3 + v_3 = 2;$$

Считая $u_1 = 0$, имеем $u_1 = 0; v_1 = 2; v_2 = 3; \mathrm{u}_2 = 1; v_3 = 8; v_4 = 3; \mathrm{u}_3 = -6$.

Для каждой свободной клетки вычислим относительные оценки:

$$\Delta_{13} = c_{13} - (u_1 + v_3) = 5 - 0 - 8 = -3;$$

$$\Delta_{14} = c_{14} - (u_1 + v_4) = 1 - 0 - 3 = -2;$$

$$\Delta_{21} = c_{21} - (u_2 + v_1) = 3 - 1 - 2 = 0;$$

$$\Delta_{31} = c_{31} - (u_3 + v_1) = 2 + 6 - 2 = 6;$$

$$\Delta_{32} = c_{32} - (u_3 + v_2) = 5 + 6 - 3 = 8;$$

$$\Delta_{34} = c_{34} - (u_3 + v_4) = 5 + 6 - 3 = 8;$$

Условие оптимальности плана перевозок $\Delta_{ij} \geq 0$ не выполняется, поэтому построим замкнутый цикл пересчета и определим величины перераспределения груза.

Минимальной оценкой является $\Delta_{13} = -3$ для клетки (1,3).

Для определения количества груза λ подлежащего распределению, построим замкнутый цикл (указан стрелками) (Таблица 7.6.4). Одна из вершин цикла находится в незанятой клетке (1,3), которую отмечаем знаком « $+$ ». Все остальные вершины цикла находятся в базисных клетках, с чередующимися знаками « $-$ » и « $+$ ». Найдем $\lambda = \min(10{,}20) = 10$, равное наименьшему из чисел, стоящих в отрицательных вершинах цикла. Значение λ записываем в незанятую клетку. Двигаясь далее по означенному циклу, вычитаем λ из объемов перевозок, расположенных в клетках, которые обозначены знаком « $-$ », и прибавляем к объемам перевозок, находящихся в клетках, отмеченных знаком « $+$ ». Элементы таблицы, не входящие в цикл, остаются без изменений. Таблица нового плана представлена в Таблице 7.6.5.

Таблица 7.6.4 – Таблица перерасчёта

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

Таблица 7.6.5 – Таблица нового плана

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

Стоимость перевозок по этому плану:

$f_2 = f_1 + \Delta_{13}\lambda = 510 - 3 * 10 = 480$ (ед.)

Вычислим потенциалы u_i и v_j исходя из базисных переменных. Для их нахождения используем условие $u_i + v_j = c_{ij}$.

$$u_1 + v_1 = 2; u_1 + v_2 = 3; u_1 + v_3 = 5; u_2 + v_2 = 4;$$

$$u_2 + v_4 = 4;\ u_3 + v_3 = 2.$$

Считая $u_1 = 0$, имеем $u_1 = 0; v_1 = 2; v_2 = 3; v_3 = 5; u_2 = 1; v_4 = 3; v_5 = 2; u_3 = -3$.

Для каждой свободной клетки вычислим относительные оценки:

$$\Delta_{14} = c_{14} - (u_1 + v_4) = 1 - 0 - 3 = -2;$$

$$\Delta_{21} = c_{21} - (u_2 + v_1) = 3 - 1 - 2 = 0;$$

$$\Delta_{23} = c_{23} - (u_2 + v_3) = 9 - 1 - 5 = 3;$$

$$\Delta_{31} = c_{31} - (u_3 + v_1) = 2 + 3 - 2 = 3;$$

$$\Delta_{32} = c_{32} - (u_3 + v_2) = 5 + 3 - 3 = 5;$$

$$\Delta_{34} = c_{34} - (u_3 + v_4) = 5 + 3 - 3 = 5;$$

Условие оптимальности плана перевозок $\Delta_{ij} \geq 0$ не выполняется, поэтому построим замкнутый цикл пересчета и определим величины перераспределения груза.

Минимальной оценкой является $\Delta_{14} = -2$ для клетки (1,4).

Для определения количества груза λ подлежащего распределению, построим замкнутый цикл (указан стрелками) (Таблица 7.6.6). Одна из вершин цикла находится в незанятой клетке $(1{,}4)$, которую отмечаем знаком « $+$ ». Все остальные вершины цикла находятся в базисных клетках, с чередующимися знаками « $-$ » и « $+$ ». Найдем $\lambda = \min(10, 50) = 10$, равное наименьшему из чисел, стоящих в отрицательных вершинах цикла. Значение λ записываем в незанятую клетку. Двигаясь далее по означенному циклу, вычитаем λ из объемов перевозок, расположенных в клетках, которые обозначены знаком « $-$ », и прибавляем к объемам перевозок, находящихся в клетках, отмеченных знаком « $+$ ». Элементы таблицы, не входящие в цикл, остаются без изменений. Таблица нового плана представлена в Таблице 7.6.7.

Таблица 7.6.6 – Таблица перерасчёта

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

Таблица 7.7.7 – Таблица нового плана

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

Стоимость перевозок по этому плану:

$f_3 = f_2 + \Delta_{14}\lambda = 480 - 2 * 10 = 460$ (ед.)

Вычислим потенциалы u_i и v_j исходя из базисных переменных. Для их нахождения используем условие $u_i + v_j = c_{ij}$.

$$u_1 + v_1 = 2; u_1 + v_3 = 5; u_1 + v_4 = 1; u_2 + v_2 = 4;$$

$$u_2 + v_4 = 4;\ u_3 + v_3 = 2;$$

Считая $u_1 = 0$, имеем $u_1 = 0; v_1 = 2; v_3 = 5; v_4 = 1; u_2 = 3; v_2 = 1; u_3 = -3$.

Для каждой свободной клетки вычислим относительные оценки:

$$\Delta_{12} = c_{12} - (u_1 + v_2) = 3 - 0 - 1 = 2;$$

$$\Delta_{21} = c_{21} - (u_2 + v_1) = 3 - 3 - 2 = -2;$$

$$\Delta_{23} = c_{23} - (u_2 + v_3) = 9 - 3 - 5 = 1;$$

$$\Delta_{31} = c_{31} - (u_3 + v_1) = 2 + 3 - 2 = 3;$$

$$\Delta_{32} = c_{32} - (u_3 + v_2) = 5 + 3 - 1 = 7;$$

$$\Delta_{34} = c_{34} - (u_3 + v_4) = 5 + 3 - 1 = 7;$$

Условие оптимальности плана перевозок $\Delta_{ij} \geq 0$ не выполняется, поэтому построим замкнутый цикл пересчета и определим величины перераспределения груза.

Минимальной оценкой является $\Delta_{21} = -2$ для клетки (2,1).

Для определения количества груза λ подлежащего распределению, построим замкнутый цикл (указан стрелками) (Таблица 7.7.8). Одна из вершин цикла находится в незанятой клетке $(2,1)$, которую отмечаем знаком « $+$ ». Все остальные вершины цикла находятся в базисных клетках, с чередующимися знаками « $-$ » и « $+$ ». Найдем $\lambda = \min(40,40) = 40$, равное наименьшему из чисел, стоящих в отрицательных вершинах цикла. Значение λ записываем в незанятую клетку. Двигаясь далее по означенному циклу, вычитаем λ из объемов перевозок, расположенных в клетках, которые обозначены знаком « $-$ », и прибавляем к объемам перевозок, находящихся в клетках, отмеченных знаком « $+$ ». Элементы таблицы, не входящие в цикл, остаются без изменений. Таблица нового плана представлена в Таблице 7.7.9.

Таблица 7.7.8 – Таблица перерасчёта

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

Таблица 7.7.9 – Таблица нового плана

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

Стоимость перевозок по этому плану:

$f_4 = f_3 + \Delta_{21}\lambda = 460 - 2 * 40 = 380$ (ед.)

Вычислим потенциалы u_i и v_j исходя из базисных переменных. Для их нахождения используем условие $u_i + v_j = c_{ij}$.

$$u_1 + v_3 = 5; u_1 + v_4 = 1; u_2 + v_1 = 3; u_2 + v_2 = 4;$$

$$u_2 + v_4 = 4; u_3 + v_3 = 2;$$

Считая $u_1 = 0$, имеем $u_1 = 0; v_3 = 5; v_4 = 1; u_3 = -3; u_2 = 3; v_2 = 1; v_1 = 0;$

Для каждой свободной клетки вычислим относительные оценки:

$$\Delta_{11} = c_{11} - (u_1 + v_1) = 2 - 0 - 0 = 2;$$

$$\Delta_{12} = c_{12} - (u_1 + v_2) = 3 - 0 - 1 = 2;$$

$$\Delta_{23} = c_{23} - (u_2 + v_3) = 9 - 3 - 5 = 1;$$

$$\Delta_{31} = c_{31} - (u_3 + v_1) = 2 + 3 - 0 = 5;$$

$$\Delta_{32} = c_{32} - (u_3 + v_2) = 5 + 3 - 1 = 7;$$

$$\Delta_{34} = c_{34} - (u_3 + v_4) = 5 + 3 - 1 = 7;$$

Отрицательных оценок нет, значит решение $x_{13} = 10; x_{14} = 50; x_{21} = 40; x_{22} = 30; x_{33} = 20$ является оптимальным. Стоимость перевозок при этом составляет $f_0 = 380$ (ед.)

7.7 Результаты выполнения программы

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

Таблица с исходными данными:

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

Рисунок 7.7.1 – Изначальная таблица

```
Начальный опорный план, полученный методом минимальной стоимости:
------------------------------------------------------------------------------------------
              |v1 = 0        |v2 = 0        |v3 = 0        |v4 = 0        |              |
              |--------------|--------------|--------------|--------------|              |
Пункты        |B1            |B2            |B3            |B4            |Запасы        |
------------------------------------------------------------------------------------------
u1 = 0        |             2|             3|             5|             1|60            |
A1            |10            |              |              |50            |              |
------------------------------------------------------------------------------------------
u2 = 0        |             3|             4|             9|             4|70            |
A2            |10            |30            |30            |              |              |
------------------------------------------------------------------------------------------
u3 = 0        |             2|             5|             2|             5|20            |
A3            |20            |              |              |              |              |
------------------------------------------------------------------------------------------
Потребности   |40            |30            |30            |50            |150           |
------------------------------------------------------------------------------------------
Стоимость перевозок по этому плану: 530 единиц
```

Рисунок 7.7.2 – Метод минимальной стоимости

```
Начальный опорный план, полученный методом северо-западного угла:

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

Рисунок 7.7.3 – Метод северо-западного угла

```
Стоимость перевозок по этому плану: 670 единиц
Вычислим потенциалы ui и vi, исходя из базисных переменных. Для их нахождения используем условия ui + vj = cij
u1 + v1 = 2
u1 + v2 = 3
u2 + v2 = 4
u2 + v3 = 9
u2 + v4 = 4
u3 + v4 = 5
Считая, что u1 = 0, имеем:
u1 = 0; u2 = 1; u3 = 2; v1 = 2; v2 = 3; v3 = 8; v4 = 3
------------------------------------------------------------------------------------------
              |v1 = 2        |v2 = 3        |v3 = 8        |v4 = 3        |              |
              |--------------|--------------|--------------|--------------|              |
Пункты        |B1            |B2            |B3            |B4            |Запасы        |
------------------------------------------------------------------------------------------
u1 = 0        |             2|             3|             5|             1|60            |
A1            |40            |20            |              |              |              |
------------------------------------------------------------------------------------------
u2 = 1        |             3|             4|             9|             4|70            |
A2            |              |10            |30            |30            |              |
------------------------------------------------------------------------------------------
u3 = 2        |             2|             5|             2|             5|20            |
A3            |              |              |              |20            |              |
------------------------------------------------------------------------------------------
Потребности   |40            |30            |30            |50            |150           |
------------------------------------------------------------------------------------------
Для каждой свободной клетки вычислим относительные оценки:
Δ13 = -3;
Δ14 = -2;
Δ21 = 0;
Δ31 = -2;
Δ32 = 0;
Δ33 = -8;
Остались отрицательные оценки, произведём перерасчёт плана...
```

Рисунок 7.7.4 – Первая итерация метода потенциалов

```
Итерация № 1:
------------------------------------------------------------------------------------------
              |v1 = 2       |v2 = 3       |v3 = 8       |v4 = 3       |              |
              |-------------|-------------|-------------|-------------|              |
Пункты        |B1           |B2           |B3           |B4           |Запасы        |
------------------------------------------------------------------------------------------
u1 = 0        |            2|            3|            5|            1|60            |
A1            |40           |20           |             |             |              |
------------------------------------------------------------------------------------------
u2 = 1        |            3|            4|            9|            4|70            |
A2            |             |10           |10           |50           |              |
------------------------------------------------------------------------------------------
u3 = 2        |            2|            5|            2|            5|20            |
A3            |             |             |20           |             |              |
------------------------------------------------------------------------------------------
Потребности   |40           |30           |30           |50           |150           |
------------------------------------------------------------------------------------------
Стоимость перевозок по этому плану: 510 единиц
Вычислим потенциалы ui и vi, исходя из базисных переменных. Для их нахождения используем условия ui + vj = cij
u1 + v1 = 2
u1 + v2 = 3
u2 + v2 = 4
u2 + v3 = 9
u2 + v4 = 4
u3 + v3 = 2
Считая, что u1 = 0, имеем:
u1 = 0; u2 = 1; u3 = -6; v1 = 2; v2 = 3; v3 = 8; v4 = 3
Для каждой свободной клетки вычислим относительные оценки:
Δ13 = -3;
Δ14 = -2;
Δ21 = 0;
Δ31 = 6;
Δ32 = 8;
Δ34 = 8;
Остались отрицательные оценки, произведём перерасчёт плана...
```

Рисунок 7.7.5 – Вторая итерация метода потенциалов

```
Итерация № 2:
------------------------------------------------------------------------------------------
              |v1 = 2       |v2 = 3       |v3 = 8       |v4 = 3       |              |
              |-------------|-------------|-------------|-------------|              |
Пункты        |B1           |B2           |B3           |B4           |Запасы        |
------------------------------------------------------------------------------------------
u1 = 0        |            2|            3|            5|            1|60            |
A1            |40           |10           |10           |             |              |
------------------------------------------------------------------------------------------
u2 = 1        |            3|            4|            9|            4|70            |
A2            |             |20           |             |50           |              |
------------------------------------------------------------------------------------------
u3 = -6       |            2|            5|            2|            5|20            |
A3            |             |             |20           |             |              |
------------------------------------------------------------------------------------------
Потребности   |40           |30           |30           |50           |150           |
------------------------------------------------------------------------------------------
Стоимость перевозок по этому плану: 480 единиц
Вычислим потенциалы ui и vi, исходя из базисных переменных. Для их нахождения используем условия ui + vj = cij
u1 + v1 = 2
u1 + v2 = 3
u1 + v3 = 5
u2 + v2 = 4
u2 + v4 = 4
u3 + v3 = 2
Считая, что u1 = 0, имеем:
u1 = 0; u2 = 1; u3 = -3; v1 = 2; v2 = 3; v3 = 5; v4 = 3
Для каждой свободной клетки вычислим относительные оценки:
Δ14 = -2;
Δ21 = 0;
Δ23 = 3;
Δ31 = 3;
Δ32 = 5;
Δ34 = 5;
Остались отрицательные оценки, произведём перерасчёт плана...
```

Рисунок 7.7.6 – Третья итерация метода потенциалов

```
Итерация № 3:
-------------------------------------------------------------------------------------------
               |v1 = 2       |v2 = 3       |v3 = 5       |v4 = 3       |              |
               |-------------|-------------|-------------|-------------|              |
Пункты         |B1           |B2           |B3           |B4           |Запасы        |
-------------------------------------------------------------------------------------------
u1 = 0         |            2|            3|            5|            1|60            |
A1             |40           |             |10           |10           |              |
-------------------------------------------------------------------------------------------
u2 = 1         |            3|            4|            9|            4|70            |
A2             |             |30           |             |40           |              |
-------------------------------------------------------------------------------------------
u3 = -3        |            2|            5|            2|            5|20            |
A3             |             |             |20           |             |              |
-------------------------------------------------------------------------------------------
Потребности    |40           |30           |30           |50           |150           |
-------------------------------------------------------------------------------------------
Стоимость перевозок по этому плану: 460 единиц
Вычислим потенциалы ui и vi, исходя из базисных переменных. Для их нахождения используем условия ui + vj = cij
u1 + v1 = 2
u1 + v3 = 5
u1 + v4 = 1
u2 + v2 = 4
u2 + v4 = 4
u3 + v3 = 2
Считая, что u1 = 0, имеем:
u1 = 0; u2 = 3; u3 = -3; v1 = 2; v2 = 1; v3 = 5; v4 = 1
Для каждой свободной клетки вычислим относительные оценки:
Δ12 = 2;
Δ21 = -2;
Δ23 = 1;
Δ31 = 3;
Δ32 = 7;
Δ34 = 7;
Остались отрицательные оценки, произведём перерасчёт плана...
```

Рисунок 7.7.7 – Четвёртая итерация метода потенциалов

```
Итерация № 4:
-------------------------------------------------------------------------------------------
               |v1 = 2       |v2 = 1       |v3 = 5       |v4 = 1       |              |
               |-------------|-------------|-------------|-------------|              |
Пункты         |B1           |B2           |B3           |B4           |Запасы        |
-------------------------------------------------------------------------------------------
u1 = 0         |            2|            3|            5|            1|60            |
A1             |             |             |10           |50           |              |
-------------------------------------------------------------------------------------------
u2 = 3         |            3|            4|            9|            4|70            |
A2             |40           |30           |             |0            |              |
-------------------------------------------------------------------------------------------
u3 = -3        |            2|            5|            2|            5|20            |
A3             |             |             |20           |             |              |
-------------------------------------------------------------------------------------------
Потребности    |40           |30           |30           |50           |150           |
-------------------------------------------------------------------------------------------
Стоимость перевозок по этому плану: 380 единиц
Вычислим потенциалы ui и vi, исходя из базисных переменных. Для их нахождения используем условия ui + vj = cij
u1 + v3 = 5
u1 + v4 = 1
u2 + v1 = 3
u2 + v2 = 4
u2 + v4 = 4
u3 + v3 = 2
Считая, что u1 = 0, имеем:
u1 = 0; u2 = 3; u3 = -3; v1 = 0; v2 = 1; v3 = 5; v4 = 1
Для каждой свободной клетки вычислим относительные оценки:
Δ11 = 2;
Δ12 = 2;
Δ23 = 1;
Δ31 = 5;
Δ32 = 7;
Δ34 = 7;
Оптимальное решение найдено!
```

Рисунок 7.7.8 – Последняя итерация метода потенциалов

7.8 Заключение

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

ЗАКЛЮЧЕНИЕ

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

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

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

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

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

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

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

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

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

СПИСОК ИНФОРМАЦИОННЫХ ИСТОЧНИКОВ

1. Болотова Л. С. Многокритериальная оптимизация. Болотова Л. С., Сорокин А. Б. [Электронный ресурс] / Метод. указания по вып. курсовой работы — М.: МИРЭА, 2015.
2. Сорокин А. Б. Методы оптимизации: гибридные генетические алгоритмы. Сорокин А. Б. [Электронный ресурс] / Учебно-метод. пособие — М.: МИРЭА, 2016.
3. Сорокин А. Б. Линейное программирование: практикум. Сорокин А. Б., Бражникова Е. В., Платонова О. В. [Электронный ресурс] / Учебно-метод. пособие — М.: МИРЭА, 2017.
4. Сухарев, А.Г. Курс методов оптимизации: учеб. пособие / А.Г. Сухарев, А.В. Тимохов, В.В. Федоров. –М.: Физматлит, 2011. – 384 с.
5. Афанасьев М.Ю. Прикладные задачи исследования операций: Учеб. пособие / М.Ю. Афанасьев, К.А. Багриновский, В.М. Матюшок – М.: ИНФРАМ, 2006 – 352 с.
6. Ашманов С.А. Теория оптимизации в задачах и упражнениях: Учеб. пособие для вузов / С.А. Ашманов, А.В. Тимохов. – СПб.: Лань, 2012. – 447 с.

ПРИЛОЖЕНИЯ

Приложение А – Код реализации метода Парето на языке Python.

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

Приложение В – Код реализации МАИ на языке Python.

Приложение Г – Код реализации симплексного метода на языке Python.

Приложение Д – Код реализации двойственной задачи на языке Python.

Приложение Е – Код реализации транспортной задачи на языке Python.

Приложение А

Код реализации метода Парето на языке Python

Листинг А.1. Реализация Парето.

```
import csv
import pandas as pd


def print_table(data):
    '''Функция для вывода таблицы'''
    print(pd.DataFrame(data).to_markdown())


def compare_alternatives(a, b):
    '''Функция для сравнения  альтернатив по отношению Парето-доминирования'''
    counter = 0
    for key in a:
        if '+' in key:
            counter += (float(a[key]) > float(b[key]))
        elif '-' in key:
            counter += (float(a[key]) < float(b[key]))
    return 1 if counter == len(a) - 1 else -1 if counter == 0 else 0

def create_Pareto_set(data):
    '''Функция для создания оптимального множества Парето по входящему множеству
альтернатив'''
    losers, winners = [], []
    for i in range(len(data)):
        for j in range(i+1, len(data)):
            n = compare_alternatives(data[i], data[j])
            if n == 1:
                losers.append(data[j])
            elif n == -1:
                losers.append(data[i])
    for i in range(len(data)):
        if data[i] not in losers:
            winners.append(data[i])
    return winners

def branches_and_boundaries(data, branches):
    '''Метод указания верхних и нижних границ критериев'''
    winners = []
    for i in data:
        flag = False
        for j in branches:
            key, value = list(j.items())[0]
            if key.count('-'):
                if float(i[key]) > value:
                    flag = True
            else:
                if float(i[key]) < value:
                    flag = True
        if not flag:
            winners.append(i)
    return create_Pareto_set(winners)


def suboptimization(data, branches, main_criteria):
    '''Метод субоптимизации'''
```

Продолжение Листинга А.1.

```
    data = branches_and_boundaries(data, branches)
    maxi = max(data, key=lambda i: i[main_criteria])
    return list(filter(lambda x: x[main_criteria] == maxi[main_criteria], data))


def lexical_optimization(data, priority):
    '''Лексикографический метод'''
    return [max(data, key = lambda item: tuple(item[key] for key in priority))]


with open('TPR_PRACT1_LIST.csv', encoding='utf-8') as file:
    data = [d for d in csv.DictReader(file)]

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

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

    print("Установка верхних и нижних границ:".center(201))
    branches = [{"Проходной балл (+)": 270}, {"Рейтинг университета (+)": 840},
                {"Расстояние до общежития (-)": 14}]
    print_table(branches_and_boundaries(data, branches))

    print("Субоптимизация:".center(201))
    branches = [{"Проходной балл (+)": 290}, {"Расстояние до общежития (-)":
14}]
    main_criteria = "Рейтинг университета (+)"
    print_table(suboptimization(data, branches, main_criteria))

    print("Лексикографическая оптимизация:".center(201))
    priority = ("Рейтинг университета (+)", "Проходной балл (+)", "Стоимость
обучения (+)",
                "Кол-во бюджетных мест (-)", "Расстояние до общежития (-)",
                "Размер стипендии (+)")
    print_table(lexical_optimization(data, priority))
```

Приложение Б

Код реализации метода Электра II на языке Python.

Листинг Б.1. Реализация метода Электра II.

```
import csv
import math
from graphviz import Digraph


def print_matrix(c = 1):
    '''Функция для вывода матрицы предпочтений с порогом'''
    print('-' * (10 * (len(matrix)+1) + 4))
    print(11 * ' ', end='')
    for i in range(1, len(matrix) + 1):
        print(f'{i}'.ljust(11), end='')
    print('\n' + '-' * (10 * (len(matrix)+1) + 4))
    for i in range(len(matrix)):
        print(str(i+1).ljust(10), end='|')
        for j in range(len(matrix)):
            if matrix[i][j] < c:
                matrix[i][j] = 0
            print(str(matrix[i][j]).ljust(10), end=' ')
        print()
    print('-' * (10 * (len(matrix)+1) + 4))


def compare_alternatives(i, j, alt_i, alt_j, criteria):
    '''Функция для сравнения альтернатив по кодам'''
    P, N = 0, 0
    P_STR, N_STR = f'P{i}{j} =', f'N{i}{j} ='
    for i in range(len(criteria)):
        counter_i, counter_j = 0, 0
        code = criteria[i]['Код'].split(';')
        for border in criteria[i]['Шкала'].split(';')[1:]:
            if alt_i[i] < float(border):
                counter_i += 1
            if alt_j[i] < float(border):
                counter_j += 1
        alt_i[i] = float(code[counter_i])
        alt_j[i] = float(code[counter_j])
        if criteria[i]['Стремление'] == '-':
            alt_i[i] *= (-1)
            alt_j[i] *= (-1)
        if alt_i[i] > alt_j[i]:
            P += int(criteria[i]['Вес критерия'])
            P_STR += (' ' + criteria[i]['Вес критерия'] + ' +')
            N_STR += (' ' + str(0) + ' +')
        elif alt_i[i] < alt_j[i]:
            N += int(criteria[i]['Вес критерия'])
            N_STR += (' ' + criteria[i]['Вес критерия'] + ' +')
            P_STR += (' ' + str(0) + ' +')
        else:
            N_STR += (' ' + str(0) + ' +')
            P_STR += (' ' + str(0) + ' +')
    return P_STR.rstrip(' +') + f' = {P}', N_STR.rstrip(' +') + f' = {N}', P, N


def generate_matrix():
    '''Функция для генерации матрицы предпочтений'''
```

Продолжение Листинга Б.1

```
    def get_D(P, N):
        '''Функция для расчёта D-стремления'''
        if N == 0 and P == 0:
            return 1
        elif N == 0 and P != 0:
            return math.inf
        value = P/N
        if math.floor(value) == math.ceil(value):
            value = int(value)
        else:
            value = round(value, 2)
        return value

    def generate_D_STR(i, j, P, N):
        '''Функция для генерации D-стремления'''
        value = get_D(P, N)
        if value <= 1:
            return f'D{i}{j} = P{i}{j} / N{i}{j} = {P}/{N} = {value} <= 1 -
отбрасываем.'
        else:
            if value == math.inf:
                value = '\u221e'
            return f'D{i}{j} = P{i}{j} / N{i}{j} = {P}/{N} = {value} > 1 -
принимаем.'

    for i in range(1, len(data)+1):
        for j in range(i+1, len(data)+1):
            print(f'Рассмотрим альтернативы {i} и {j} (i = {i}, j = {j}):')
            alt_i, alt_j = data[i-1].copy(), data[j-1].copy()
            P_STR, N_STR, P, N = compare_alternatives(
                i, j, alt_i, alt_j, criteria)
            print(P_STR+';', N_STR+';', sep='\n')
            print(generate_D_STR(i, j, P, N))
            D = get_D(P, N)
            if D > 1:
                matrix[i-1][j-1] = D
            print(f'P{j}{i}{N_STR[3:]};', f'N{j}{i}{P_STR[3:]};', sep='\n')
            print(generate_D_STR(j, i, N, P))
            D = get_D(N, P)
            if D > 1:
                matrix[j-1][i-1] = D


def draw_graph(c=1):
    '''Функция для рисования хаотичного графа с порогом'''
    dot = Digraph(f'Хаотичный Граф с порогом = {c}')
    for i in range(len(matrix)):
        dot.node(str(i+1))
    for i in range(len(matrix)):
        for j in range(len(matrix)):
            if matrix[i][j] >= c:
                dot.edge(str(i+1), str(j+1))
    dot.render(view=True)


def smart_draw_graph(levels, c=1):
    '''Функция для рисования графа по уровням с порогом'''
    dot1 = Digraph(f"Граф с порогом = {c}")
    for i in range(len(matrix)):
        for j in range(len(matrix)):
            if matrix[i][j] >= c:
```

Продолжение Листинга Б.1

```
                dot1.edge(str(i+1), str(j+1))
    for i in range(len(levels)):
        sub = Digraph(name='Подграф'+str(i))
        sub.attr(rank='same')
        sub.node(f'{i+1}-ый уровень')
        for j in levels[i]:
            sub.node(f'{j+1}')
        dot1.subgraph(sub)
    dot1.render(view=True)


def get_levels(c=1):
    '''Вспомогательная функция для определения уровня вершин'''
    levels = []  # массив всех вершин
    visited = []  # массив посещённых вершин
    while len(visited) < len(matrix):
        level = []
        for i in range(len(matrix)):
            if i in visited:
                continue
            flag = True
            for j in range(len(matrix)):
                if matrix[j][i] >= c:
                    flag = any(j in lev for lev in levels)
                if not flag:
                    break
            if flag:
                level.append(i)
                visited.append(i)
        levels.append(level)
        print(f'{len(levels)}-ый уровень: ' +
              ', '.join(map(lambda x: str(x+1), level)))
    return levels


with open('TPR_PRACT2_LIST.csv', encoding='utf-8') as file, \
        open('codes.csv', encoding='utf-8') as criteria_file:
    criteria = [i for i in csv.DictReader(
        criteria_file)]  # Информация о критериях
    data = list(map(lambda x: [float(i) for i in x], [
                i[1:] for i in csv.reader(file)][1:]))  # Значения критериев для
рассматриваемых альтернатив
    matrix = [[0]*len(data) for _ in range(len(data))]  # Матрица предпочтений
    generate_matrix()
    print("Матрица предпочтений:".center(201))
    print_matrix()
    draw_graph()
    arg = 1.76
    smart_draw_graph(get_levels(c=arg), c=arg)
```

Приложение В

Код реализации МАИ на языке Python.

Листинг В.1. Реализация МАИ.

```
import pandas as pd
import functools


NUM_CRITERIA = 5  # Количество критериев для сравнения
NUM_ALTERNATIVES = 5  # Количество альтернатив
СИ = 1.12  # среднее значение индекса случайной согласованности


def print_table(data):
    '''Функция для вывода таблицы'''
    print(pd.DataFrame(data).to_markdown())


def relative_value(data, MATRIX_SIZE=NUM_CRITERIA):
    '''Функция для определения относительной ценности элемента (геометрическое 
среднее)'''
    return round(functools.reduce(lambda a, b: a * b, data) ** (1 / 
MATRIX_SIZE), 3)


def compare_by_criteria(data, index):
    '''Функция для сравнения в пределах матрицы сравнения по критерию N'''
    V = []
    for i in range(NUM_ALTERNATIVES):
        val = relative_value(data[i])
        print(
            f'Строка №{i + 1}\nVk{index}{i+1} = ({" * ".join([str(i) for i in 
data[i]])}) ^ 1/{NUM_ALTERNATIVES} = {val}')
        V.append(val)
    print(
        f'Проведена нормализация полученных чисел. Для этого определен 
нормирующий коэффициент ∑VK{index}Y.')
    print(
        f'∑VK{index}Y  = VK{index}1 + VK{index}2 + VK{index}3 + VK{index}4 + 
VK{index}5 = {" + ".join([str(i) for i in V])} = {round(sum(V), 3)}.')
    print(
        f'Найдена важность приоритетов W3K{index}Y, для этого каждое из чисел 
VK{index}Y разделено на ∑VK{index}Y.')
    Y = []
    for i in range(NUM_ALTERNATIVES):
        Y.append(round(V[i] / sum(V), 3))
        print(
            f'Строка №{i + 1}\nW3K{index}{i + 1} = {V[i]} / ∑Vi = {V[i]} / 
{sum(V)} = {Y[i]};')

    print('В результате получаем вектор приоритетов:')
    print(f'W3K{index}Y = ({"; ".join([f"Y3{index}{i + 1} = {Y[i]}" for i in 
range(len(Y))])}), '
          + f'где индекс 3 означает, что вектор приоритетов относится к третьему 
уровню иерархии критерия K{index}.')
    return Y



def check_matrix_consistency(data, priority_vector, index):
```

Продолжение Листинга В.1.

```
    print(
        f'Определены индекс согласованности и отношение согласованности для
матрицы K{index}')
    print('Определяется сумма каждого столбца матрицы суждений.')
    counter = 1
    S = []
    for i in zip(*data):
        S.append(sum(i))
        print(
            f'S{counter}K{index} = {" + ".join([str(i) for i in list(i)])} =
{sum(i)}')
        counter += 1
    print('Затем полученный результат умножен на компоненту нормализованного
вектора приоритетов.')
    P = []
    for i in range(len(S)):
        P.append(round(S[i] * priority_vector[i], 3))
        print(f'P{i + 1}K{index} = S{i + 1} * W3K{index}{i + 1} = {P[i]}')
    print('Найдена пропорциональность предпочтений.')
    print(f'λmaxK{index} = P1K{index} + P2K{index} + P3K{index} + P4K{index} +
P5K{index} = {round(sum(P), 3)}')
    print('Отклонение от согласованности выражается индексом согласованности.')
    ИС = round((round(sum(P), 3) - 5) / (5 - 1), 3)
    print(
        f'ИСk{index} = (λmaxK{index} - n)/(n - 1) = ({round(sum(P), 3)}-5)/(5-1)
= {ИС}.')
    print('Найдено отношение согласованности ОС.')
    print(f'ОСk{index} = ИС/СИ = {round(ИС / СИ, 3)}.')
def synthesis_of_alternatives(Y, full_Y):
    print(f'Векторы приоритетов и отношения согласованности определяются для
всех матриц суждений, начиная со второго уровня.\n' +
          f'Для определения приоритетов альтернатив локальные приоритеты
умножены на приоритет соответствующего критерия' +
          f'на высшем уровне и найдены суммы по каждому элементу в соответствии
с критериями, на которые воздействует этот элемент.')
    print(
        f'W2i = ({"; ".join([f"Y2{i + 1} = {Y[i]}" for i in range(len(Y))])});')
    for i in range(len(full_Y)):
        print(
            f'W3K{i + 1}Y = ({"; ".join([f"Y3{i + 1}{j + 1} = {full_Y[i][j]}"
for j in range(len(full_Y[i]))])});')
    print('Приоритеты альтернатив получены следующим образом:')
    winners = []
    counter = 0
    for v in zip(*full_Y):
        curr_str = str()
        w = 0
        counter += 1
        curr_str += f'W{counter} = '
        for i in range(len(v)):
            curr_str += f'W2{i + 1} * W3K{i + 1}{counter} + '
            w += (v[i] * Y[i])
        w = round(w, 3)
        print(curr_str.rstrip(' + ') + ' = ' + str(w))
        winners.append(w)
    print('Таким образом, приоритеты альтернатив равны:')
    for i in range (NUM_ALTERNATIVES):
        print(f'альтернатива A{i+1} - W{i + 1} приоритет равен {winners[i]}')
    return winners
```

Продолжение Листинга В.1.

```
with open('TPR_PRACT3.csv', encoding='utf-8') as file:
    title = file.readline().rstrip()  # Текущая матрица, которая будет считана
    print(title.center(201))
    criteria_paired_comparison_matrix = [[float(i) for i in
file.readline().rstrip(
    ).split(',')] for _ in range(NUM_CRITERIA)]  # Матрица парного сравнения
критериев
    print_table(criteria_paired_comparison_matrix)
    print('Для определения относительной ценности каждого элемента необходимо
найти геометрическое' +
          ' среднее и с этой целью перемножить n элементов каждой строки и из
полученного' +
          ' результата извлечь корни n-й степени (размерность матрицы n=5).')
    V = []

    for i in range(NUM_CRITERIA):
        val = relative_value(criteria_paired_comparison_matrix[i])
        print(
            f'Строка №{i + 1}\nV{i + 1} = ({" * ".join([str(i) for i in
criteria_paired_comparison_matrix[i]])}) ^ 1/{NUM_CRITERIA} = {val}')
        V.append(val)
    print(
        f'∑Vi = V1 + V2 + V3 + V4 + V5 = {" + ".join([str(i) for i in V])} =
{round(sum(V),3)}')
    print('Найдена важность приоритетов W2i, для этого каждое из чисел Vi
разделено на ∑Vi.')
    Y = []  # Вектор приоритетов W2i

    for i in range(NUM_CRITERIA):
        Y.append(round(V[i] / sum(V), 3))
        print(
            f'Строка №{i + 1}\nW2{i + 1} = {V[i]} / ∑Vi = {Y[i]} = Y{2}{i + 1}')
    print(f'В результате получен вектор приоритетов:\nW2i = ({"; ".join([f"Y2{i
+ 1} = {Y[i]}" for i in range(len(Y))])}), '
          + 'где индекс 2 означает, что вектор приоритетов относится ко второму
уровню иерархии.')

    big_data, full_Y = [], []
    for i in range(NUM_CRITERIA):
        title = file.readline().rstrip()
        print(title.center(201))
        data = [[float(i) for i in file.readline().rstrip().split(',')] for _ in
range(
            NUM_ALTERNATIVES)]  # Матрица сравнения по i + 1-ому критерию
        big_data.append(data)
        print_table(data)
        Yi = compare_by_criteria(data, i + 1)
        full_Y.append(Yi)
    print('Определены индекс согласованности и отношение согласованности для
матрицы «Выбор лучшего технического вуза»')

    counter = 1
    S = []
    for i in zip(*criteria_paired_comparison_matrix):
        S.append(sum(i))
        print(
            f'S{counter} = {" + ".join([str(i) for i in list(i)])} = {sum(i)}')
        counter += 1
    print(f'Полученный результат умножен на компоненту нормализованного вектора
приоритетов, ' +
          f'т.е. сумму суждений первого столбца на первую компоненту, сумму
```

Продолжение Листинга В.1.

```
суждений второго столбца - на вторую и т.д.')
    P = []
    for i in range(len(S)):
        P.append(round(S[i] * Y[i], 3))
        print(f'P{i + 1} = S{i + 1} * W2{i + 1} = {P[i]}')
    print(f'Сумма чисел Pj отражает пропорциональность предпочтений, ' +
          f'чем ближе эта величина к n (числу объектов и видов действия в
матрице парных сравнений), тем более согласованны суждения.')
    print(f'λmax = P1 + P2 + P3 + P4 + P5 = {round(sum(P), 3)}')
    print('Отклонение от согласованности выражается индексом согласованности.')
    ИС = round((round(sum(P), 3) - 5) / (5 - 1), 3)
    print(f'ИС = (λmax - n)/(n - 1) = ({round(sum(P), 3)}-5)/(5-1) = {ИС}.')
    print('Отношение индекса согласованности ИС к среднему значению случайного
индекса согласованности СИ называется отношением согласованности ОС.')
    print(f'ОС = ИС/СИ = {round(ИС / СИ, 3)}.')
    for i in range(NUM_CRITERIA):
        print('\n')
        check_matrix_consistency(big_data[i], full_Y[i], i + 1)
    synthesis_of_alternatives(Y, full_Y)
```

Приложение Г

Код реализации симплексного метода на языке Python.

Листинг Г.1. Реализация симплексного метода.

```
import re


NUM_CRITERIA = 4  # Количество ограничений в математической моделе
PRECISION = 4  # Количество знаков после запятой при округлении
SEP = 25 # Разделитель для вывода таблицы


def print_table(system, coef_basis, coef_not_basis, basis_values,
not_basis_values):
    print(' '.ljust(SEP), end='')
    print('Cj'.ljust(SEP), end='')
    for i in range(len(coef_not_basis) + 1):
        if i == len(coef_not_basis):
            print(' '.ljust(SEP))
        else:
            print(str(coef_not_basis[i]).ljust(SEP), end='')
    print('Cv'.ljust(SEP), end='')
    for i in range(len(not_basis_values) + 1):
        if i == 0:
            print(''.ljust(SEP), end='')
        else:
            print(str(not_basis_values[i-1]).ljust(SEP), end='')
    print('A0'.ljust(SEP))
    system.insert(0, coef_basis + [' '])
    system.insert(1, basis_values + ['f'])
    for column in range(len(system[0])):
        for row in range(len(system)):
            print(str(system[row][column]).ljust(SEP), end='')
        print()
    del system[0]
    del system[0]


def get_coefficients(data):
    '''Функция для получения списка коэффициентов из системы ограничений'''
    criteria_coefficients, boundaries = list(), list()
    for exp in data:
        if '<=' in exp:
            parse = exp.split('<=')
        elif '>=' in exp:
            parse = exp.split('>=')
        elif '<' in exp:
            parse = exp.split('<')
        elif '>' in exp:
            parse = exp.split('>')
        elif '=' in exp:
            parse = exp.split('=')
        parse = list(map(str.strip, parse))
        boundaries.append(float(parse[1]))
        criteria_coefficients.append(list(map(float, [i.group(1) for i in
re.finditer(
            r'(\d+(\.\d+)?) {0,}[*]? {0,}\w', parse[0])])))
    return criteria_coefficients, boundaries
```

Продолжение Листинга Г.1.

```
def count_scalar_product(vec1, vec2):
    '''Функция для рассчёта скалярного произведения двух векторов'''
    res = 0
    for i in range(len(vec1)):
        res += (vec1[i] * vec2[i])
    return res


def create_simplex_table(system, coef_basis, coef_not_basis, basis_values,
not_basis_values):
    F_str = [0] * len(not_basis_values)
    for i in range(len(not_basis_values)):
        F_str[i] = count_scalar_product(
            coef_basis, system[i]) - coef_not_basis[i]
    Q = count_scalar_product(coef_basis, system[-1])
    for i in range(len(F_str)):
        system[i].append(F_str[i])
    system[-1].append(Q)
    return F_str, Q


def simplex_iteration(system, coef_basis, coef_not_basis, basis_values,
not_basis_values, F_str, Q):
    index_column = F_str.index(min(F_str))
    mini = 1e10
    for i in range(len(system[-1]) - 1):
        tmp = system[-1][i] / system[index_column][i]
        if tmp < mini:
            mini = tmp
            index_row = i
    key_element = system[index_column][index_row]
    basis_values.insert(index_row, not_basis_values[index_column])
    not_basis_values.insert(index_column, basis_values.pop(index_row + 1))
    del not_basis_values[index_row]
    coef_basis[index_row], coef_not_basis[index_column] =
coef_not_basis[index_column], coef_basis[index_row]
    new_key_element = round(1 / key_element, PRECISION)
    data = [[0] * (len(coef_basis) + 1)
            for _ in range(len(coef_not_basis) + 1)]
    for i in range(len(system[index_column])):
        data[index_column][i] = - round(system[index_column][i] / key_element,
PRECISION)
    for i in range(len(system)):
        data[i][index_row] = round(
            system[i][index_row] / key_element, PRECISION)
    data[index_column][index_row] = new_key_element
    for row in range(len(data[0])):
        for column in range(len(data)):
            if data[column][row] == 0:
                data[column][row] = round(((system[column][row] * key_element) -
(
                    system[index_column][row] * system[column][index_row])) /
key_element, PRECISION)
    F_str = [data[i][-1] for i in range(len(data) - 1)]
    Q = data[-1][-1]
    return data, coef_basis, coef_not_basis, basis_values, not_basis_values,
F_str, Q

with open('TPR_PRACT5.csv', encoding='utf-8') as file:
    target_function = file.readline().rstrip()  # Целевая функция
    target_coefficients = list(map(float, [i.group(1) for i in re.finditer(
```

Продолжение Листинга Г.1.

```
        r'(\d+(\.\d+)?) {0,}[*]? {0,}\w', target_function)]))  # Список
коэффициентов целевой функции
    criteria_function = [file.readline().rstrip() for _ in range (NUM_CRITERIA)]
    criteria_coefficients, boundaries = get_coefficients(criteria_function)
    print('Переходим к задаче линейного программирования:',
          target_function, sep='\n')
    for i in criteria_function:
        print("{ " + i)
    system = list(map(list, list(zip(*criteria_coefficients))))
    system.append(boundaries)
    # Вектор коэффициентов целевой функции при базисных переменных
    coef_basis = [0, 0, 0, 0]
    # Коэффициенты целевой функции, соответствующие небазисным переменным
    coef_not_basis = target_coefficients.copy()
    not_basis_values = re.findall(r'\w\d{1,}', target_function)
    basis_values = [f'{not_basis_values[-1][0]}{i}' for i in range(
        int(not_basis_values[-1][1]) + 1, NUM_CRITERIA + int(not_basis_values[-
1][1]) + 1)]
    F_str, Q = create_simplex_table(
        system, coef_basis, coef_not_basis, basis_values, not_basis_values)
    num_iteration = 0
    while num_iteration < 50 and min(F_str) < 0:
        print(
            ('\x1b[6;30;42m' + f"Итерация №{num_iteration}" +
'\x1b[0m').center(201))
        # print(f'Итерация №{num_iteration}'.center(201))
        system, coef_basis, coef_not_basis, basis_values, not_basis_values,
F_str, Q = simplex_iteration(
            system, coef_basis, coef_not_basis, basis_values, not_basis_values,
F_str, Q)
        print_table(system, coef_basis, coef_not_basis,
                    basis_values, not_basis_values)
        num_iteration += 1
    if num_iteration != 50:
        print(f'Решение найдено! Общая прибыль составила {Q} денежных единиц')
    else:
        print('Поставленная задача решения не имеет')
```

Приложение Д

Код реализации двойственной задачи на языке Python.

Листинг Д.1. Реализация двойственной задачи.

```
import re
import math
import sympy
import numpy as np


NUM_CRITERIA = 4  # Количество ограничений в математической моделе
PRECISION = 8  # Количество знаков после запятой при округлении
SEP = 25  # Разделитель для вывода таблицы


def print_table(system, coef_basis, coef_not_basis, basis_values,
not_basis_values):
    print(' '.ljust(SEP), end='')
    print('Cj'.ljust(SEP), end='')
    for i in range(len(coef_not_basis) + 1):
        if i == len(coef_not_basis):
            print(' '.ljust(SEP))
        else:
            print(str(coef_not_basis[i]).ljust(SEP), end='')
    print('Cv'.ljust(SEP), end='')
    for i in range(len(not_basis_values) + 1):
        if i == 0:
            print(''.ljust(SEP), end='')
        else:
            print(str(not_basis_values[i-1]).ljust(SEP), end='')
    print('A0'.ljust(SEP))
    system.insert(0, coef_basis + [' '])
    system.insert(1, basis_values + ['f'])
    for column in range(len(system[0])):
        for row in range(len(system)):
            print(str(system[row][column]).ljust(SEP), end='')
        print()
    del system[0]
    del system[0]


def get_coefficients(data):
    '''Функция для получения списка коэффициентов из системы ограничений'''
    criteria_coefficients, boundaries = list(), list()
    for exp in data:
        if '<=' in exp:
            parse = exp.split('<=')
        elif '>=' in exp:
            parse = exp.split('>=')
        elif '<' in exp:
            parse = exp.split('<')
        elif '>' in exp:
            parse = exp.split('>')
        elif '=' in exp:
            parse = exp.split('=')
        parse = list(map(str.strip, parse))
        boundaries.append(float(parse[1]))
        criteria_coefficients.append(list(map(float, [i.group(1) for i in
re.finditer(
            r'(\d+(\.\d+)?) {0,}[*]? {0,}\w', parse[0])])))
```

```
    return criteria_coefficients, boundaries


def count_scalar_product(vec1, vec2):
    '''Функция для рассчёта скалярного произведения двух векторов'''
    res = 0
    for i in range(len(vec1)):
        res += (vec1[i] * vec2[i])
    return res


def create_simplex_table(system, coef_basis, coef_not_basis, basis_values,
not_basis_values):
    F_str = [0] * len(not_basis_values)
    for i in range(len(not_basis_values)):
        F_str[i] = count_scalar_product(
            coef_basis, system[i]) - coef_not_basis[i]
    Q = count_scalar_product(coef_basis, system[-1])
    for i in range(len(F_str)):
        system[i].append(F_str[i])
    system[-1].append(Q)
    return F_str, Q


def simplex_iteration(system, coef_basis, coef_not_basis, basis_values,
not_basis_values, F_str, Q):
    index_column = F_str.index(min(F_str))
    mini = 1e10
    for i in range(len(system[-1]) - 1):
        tmp = system[-1][i] / system[index_column][i]
        if tmp < mini:
            mini = tmp
            index_row = i
    key_element = system[index_column][index_row]
    basis_values.insert(index_row, not_basis_values[index_column])
    not_basis_values.insert(index_column, basis_values.pop(index_row + 1))
    del not_basis_values[index_row]
    coef_basis[index_row], coef_not_basis[index_column] =
coef_not_basis[index_column], coef_basis[index_row]
    new_key_element = round(1 / key_element, PRECISION)
    data = [[0] * (len(coef_basis) + 1)
            for _ in range(len(coef_not_basis) + 1)]
    for i in range(len(system[index_column])):
        data[index_column][i] = - \
            round(system[index_column][i] / key_element, PRECISION)
    for i in range(len(system)):
        data[i][index_row] = round(
            system[i][index_row] / key_element, PRECISION)
    data[index_column][index_row] = new_key_element
    for row in range(len(data[0])):
        for column in range(len(data)):
            if data[column][row] == 0:
                data[column][row] = round(((system[column][row] * key_element) -
(
                    system[index_column][row] * system[column][index_row])) /
key_element, PRECISION)
    F_str = [data[i][-1] for i in range(len(data) - 1)]
    Q = data[-1][-1]
    return data, coef_basis, coef_not_basis, basis_values, not_basis_values,
F_str, Q
```

```
def check_inequality(inequality, variables,
индексы_базисных_переменных_оптимального_плана):
    inequality = inequality.replace('*', '')
    for i in range(len(variables)):
        if variables[i] not in inequality:
            continue
        inequality = inequality.replace(
            variables[i], '*' +
str(индексы_базисных_переменных_оптимального_плана[i]))
    # Символьное вычисление неравенства
    inequality += '- 0.1'
    result = str(sympy.sympify(inequality))
    # Возвращение True, если неравенство выполняется, иначе False
    return eval(result)

def dual_task():
    коэффициенты_целевой_функции = np.array(target_coefficients)
    свободные_члены_неравенств = np.array(boundaries)
    матрица_ограничений, _ = get_coefficients(criteria_function)
    транспонированная_матрица_ограничений = np.transpose(матрица_ограничений)
    индексы_базисных_переменных_оптимального_плана = np.array(system[-1][:-1])
    y = np.array([])
    D = list()
    for i in range(len(basis_values)):
        index = int(basis_values[i][1:]) - 1
        if index < len(транспонированная_матрица_ограничений):
            D.append(транспонированная_матрица_ограничений[index])
        else:
            D.append(
                np.array([1 if i == j else 0 for j in range(NUM_CRITERIA)]))
    D_inversed = np.linalg.inv(np.transpose(D))

    def first_duality_theorem():
        y = np.dot(np.array(coef_basis), D_inversed)
        G_min = np.dot(свободные_члены_неравенств, y)
        print(f"Gmin is {G_min} by first_duality_theorem")
        assert abs(G_min - Q) < 0.00001

    def second_duality_theorem():
        nonlocal y
        zeros = list()
        for i in range(NUM_CRITERIA):
            if check_inequality(
                    criteria_function[i], basis_values,
индексы_базисных_переменных_оптимального_плана):
                zeros.append(i)
        система_уравнений = транспонированная_матрица_ограничений.copy()
        for i in range(len(zeros)):
            система_уравнений = np.delete(
                система_уравнений, zeros[i], 1)
            for j in range(i + 1, len(zeros)):
                zeros[j] -= 1
        y = np.linalg.solve(система_уравнений, коэффициенты_целевой_функции)
        for i in range(len(zeros)):
            y = np.insert(y, zeros[i], 0)
            for j in range(i + 1, len(zeros)):
                zeros[j] += 1
        G_min = np.dot(свободные_члены_неравенств, y)
```

Продолжение Листинга Д.1.

```
        print(f"Gmin is {G_min} by second_duality_theorem")
        assert abs(G_min - Q) < 0.00001

    def third_duality_theorem():
        нижняя_граница = list()
        верхняя_граница = list()
        b = list()
        for i in range(len(D_inversed) - 1, -1, -1):
            pozitive = list()
            negative = list()
            bH = - math.inf
            bB = math.inf
            for j in range(len(D_inversed)):
                if D_inversed[j][i] > 0:
                    pozitive.append(
                        (свободные_члены_неравенств[j], D_inversed[j][i]))
                elif D_inversed[j][i] < 0:
                    negative.append(
                        (свободные_члены_неравенств[j], D_inversed[j][i]))
            if len(pozitive) > 1:
                elem = min(pozitive, key=lambda x: abs(
                    pozitive[0][0] / pozitive[0][1]))
                нижняя_граница.append(elem[0] / elem[1])
            elif len(pozitive) == 1:
                нижняя_граница.append(pozitive[0][0] / pozitive[0][1])
            else:
                нижняя_граница.append(bH)

            if len(negative) > 1:
                elem = max(negative, key=lambda x: abs(
                    negative[0][0] / negative[0][1]))
                верхняя_граница.append(abs(elem[0] / elem[1]))
            elif len(negative) == 1:
                верхняя_граница.append(negative[0][0] / negative[0][1])
            else:
                верхняя_граница.append(bB)

            b.append(свободные_члены_неравенств[i])
            print(f'Ресурс №{len(D_inversed)-i}')
            print(
                f'b{len(D_inversed)-i} ∈ ({нижняя_граница[-1]};
{верхняя_граница[-1]})')
            print(f'{len(D_inversed)-i}-й ресурс изменяется в интервале: ',
end='')
            if нижняя_граница[-1] == - math.inf:
                print(f'({нижняя_граница[-1]}; ', end='')
            else:
                print(f'({b[-1] - нижняя_граница[-1]}; ', end='')
            if верхняя_граница[-1] == math.inf:
                print(f'{верхняя_граница[-1]})')
            else:
                print(f'{b[-1] + верхняя_граница[-1]})')
        total = 0
        for i in range(len(y)):
            if y[i] != 0:
                total += y[i] * верхняя_граница[i]
                print(f'ΔGmax{i + 1} = y{i+1} * bB{i +
                      1} = {y[i] * верхняя_граница[i]}')
        print(f'Совместное влияние изменений этих ресурсов приводит к изменению
максимальной стоимости продукции Gmax на величину: {total}')
        print(f'Следовательно, оптимальное значение целевой функции при
```

Продолжение Листинга Д.1.

```
 максимальном изменении ресурсов: {Q+total}')

    first_duality_theorem()
    second_duality_theorem()
    third_duality_theorem()


with open('TPR_PRACT5.csv', encoding='utf-8') as file:
    target_function = file.readline().rstrip()  # Целевая функция
    target_coefficients = list(map(float, [i.group(1) for i in re.finditer(
        # Список коэффициентов целевой функции
        r'(\d+(\.\d+)?) {0,}[*]? {0,}\w', target_function)]))
    criteria_function = [file.readline().rstrip() for _ in range(NUM_CRITERIA)]
    criteria_coefficients, boundaries = get_coefficients(criteria_function)
    print('Переходим к задаче линейного программирования:',
          target_function, sep='\n')
    for i in criteria_function:
        print("{ " + i)
    system = list(map(list, list(zip(*criteria_coefficients))))
    system.append(boundaries.copy())
    # Вектор коэффициентов целевой функции при базисных переменных
    coef_basis = [0] * NUM_CRITERIA
    # Коэффициенты целевой функции, соответствующие небазисным переменным
    coef_not_basis = target_coefficients.copy()
    not_basis_values = re.findall(r'[A-Za-z]\d{1,}', target_function)
    basis_values = [f'{not_basis_values[-1][0]}{i}' for i in range(
        int(not_basis_values[-1][1]) + 1, NUM_CRITERIA + int(not_basis_values[-
1][1]) + 1)]
    F_str, Q = create_simplex_table(
        system, coef_basis, coef_not_basis, basis_values, not_basis_values)
    num_iteration = 0
    while num_iteration < 50 and min(F_str) < 0:
        print(
            ('\x1b[6;30;42m' + f"Итерация №{num_iteration}" +
'\x1b[0m').center(201))
        system, coef_basis, coef_not_basis, basis_values, not_basis_values,
F_str, Q = simplex_iteration(
            system, coef_basis, coef_not_basis, basis_values, not_basis_values,
F_str, Q)
        print_table(system, coef_basis, coef_not_basis,
                    basis_values, not_basis_values)
        num_iteration += 1
    if num_iteration != 50:
        print(f'Решение найдено! Общая прибыль составила {
              round(Q, 3)} денежных единиц')
    else:
        print('Поставленная задача решения не имеет')
        exit(0)
    dual_task()
```

Приложение Е

Код реализации транспортной задачи на языке Python.

Листинг Е.1. Реализация транспортной задачи.

```
import csv
import sympy
import copy
import re
import math
import numpy as np
from itertools import product


def transport_task():
    '''Функция, отвечающая за решение закрытой транспортной задачи'''
    # поставщики, потребители, c = input_data()
    поставщики, потребители, c = input_data_from_file()
    assert sum(поставщики) == sum(
        потребители), 'Транспортная задача не является закрытой'
    C = np.vstack(c)  # Стоимости перевозок единицы груза из Ai в Bi
    X = np.zeros_like(C)
    basis = list()  # Базисные переменные (заполненные клетки)
    U = np.zeros_like(поставщики)  # Потенциалы пунктов Ai
    V = np.zeros_like(потребители)  # Потенциалы пунктов Bj
    delta = np.zeros_like(C)  # Относительные оценки клеток
    marks = list(product([f'u{i}' for i in range(
        1, len(поставщики) + 1)], [f'v{i}' for i in range(1, len(потребители) +
1)]))
    num_iteration = 0  # Номер итерации в методе потенциалов

    def print_table(potential=True):
        '''Функция для вывода таблицы'''
        print('-' * (14 * (len(потребители) + 2) + len(потребители) + 2))
        if potential:
            print(' ' * 14, end='|')
            for i in range(len(потребители)):
                print(f'v{i + 1} = {V[i]}'.ljust(14), end='|')
            print(''.ljust(14), end='|')
            print('\n' + ' ' * 14 + ('|' + '-' * 14)
                  * len(потребители), end='|')
            print(''.ljust(14), end='|')
            print()
        print('Пункты'.ljust(14), end='|')
        for i in range(len(потребители)):
            print(f'B{i + 1}'.ljust(14), end='|')
        print('Запасы'.ljust(14), end='|')
        for i in range(len(поставщики)):
            print('\n' + '-' * (14 * (len(потребители) + 2) + len(потребители) +
2))
            if potential:
                print(f'u{i + 1} = {U[i]}'.ljust(14), end='|')
            else:
                print(f'A{i + 1}'.ljust(14), end='|')
            for j in range(len(потребители)):
                print(f'{C[i][j]}'.rjust(14), end='|')
            print('\033[94m' + f'{поставщики[i]
                              }'.ljust(14) + '\033[0m', end='|')
            if potential:
```

Продолжение Листинга Е.1.

```
                print('\n' + f'A{i + 1}'.ljust(14), end='|')
            else:
                print('\n' + ' ' * 14, end='|')
            for j in range(len(потребители)):
                if (i, j) not in basis:
                    print(f''.ljust(14), end='|')
                else:
                    print('\033[102m' +
                          f'{X[i][j]}'.ljust(14) + '\033[0m', end='|')
            print(''.ljust(14), end='|')
        print('\n' + '-' * (14 * (len(потребители) + 2) + len(потребители) + 2))
        print('Потребности'.ljust(14), end='|')
        for i in range(len(потребители)):
            print('\033[94m' + f'{потребители[i]
                                 }'.ljust(14) + '\033[0m', end='|')
        print('\033[95m' + f'{sum(потребители)
                              }'.ljust(14) + '\033[0m', end='|')
        print('\n' + '-' * (14 * (len(потребители) + 2) +
              len(потребители) + 2))

    def northwest_corner_method():
        '''Метод северо-западного угла нахождения начального опорного решения'''
        for i in range(len(поставщики)):
            for j in range(len(потребители)):
                X[i][j] = min(поставщики[i] - np.sum(X[i, :]),
                              потребители[j] - np.sum(X[:, j]))
                if X[i][j] != 0:
                    basis.append((i, j))

    def get_min_indexes(suppliers, consumers):
        '''Вспомогательная функция для определения индексов ячейки с минимальной
стоимостью'''
        min_cost = np.inf  # хранит наименьший элемент, который нашли на текущий
момент в матрице C
        min_indexes = (None, None)  # индексы наименьшего элемента
        for i in range(len(C)):
            for j in range(len(C[0])):
                if min_cost > C[i][j] and C[i][j] > 0 and X[i][j] == 0:
                    if suppliers[i] > 0 and consumers[j] > 0:
                        # назначаем (i, j) элемент новым наименьшим
                        min_cost = C[i][j]
                        min_indexes = (i, j)
        return min_indexes

    def min_price_method():
        suppliers = np.copy(поставщики)  # Копия вектора поставщиков
        consumers = np.copy(потребители)  # Копия вектора потребителей
        '''Метод минимальной стоимости нахождения опорного решения'''
        while True:
            i, j = get_min_indexes(suppliers, consumers)
            if i is None and j is None:
                # все потребности удовлетворены и/или все возможности
использованы
                break
            resources = min(suppliers[i], consumers[j])
            suppliers[i] = suppliers[i] - resources
            consumers[j] = consumers[j] - resources
            X[i][j] = resources
            basis.append((i, j))
```

Продолжение Листинга Е.1.

```
    def count_function():
        '''Расчёт стоимости перевозок по текущему плану'''
        return np.dot(C.reshape(len(поставщики) * len(потребители)),
X.reshape(len(поставщики) * len(потребители)))

    def count_potentials(calculation_output=False):
        ''' Функция для подсчёта потенциалов Ui, Vi'''
        system = []
        for u, v in marks:
            if (int(u[1]) - 1, int(v[1]) - 1) in basis:
                system.append(f'{u} + {v} = {C[int(u[1]) - 1][int(v[1]) - 1]}')
        if calculation_output:
            print("Вычислим потенциалы ui и vi, исходя из базисных переменных.
Для их нахождения используем условия ui + vj = cij")
            print(*system, sep='\n')
        first_equation = system[0]
        first_variable = first_equation.split()[0]
        variables = set()
        for equation in system:
            for symbol in equation.split():
                if re.fullmatch(r'[uv]\d*', symbol):
                    variables.add(symbol)
        symbols_dict = {symbol: sympy.symbols(symbol) for symbol in variables}
        system[0] = first_equation.replace(first_variable, '0')
        equations = []
        for equation in system:
            left, right = equation.split('=')
            equations.append(
                sympy.Eq(sympy.sympify(left), sympy.sympify(right)))
        equations.append(sympy.Eq(symbols_dict[first_variable], 0))
        solution = sympy.solve(equations, list(symbols_dict.values()))
        for key, value in solution.items():
            if str(key)[0] == 'u':
                U[int(str(key)[1]) - 1] = value
            else:
                V[int(str(key)[1]) - 1] = value
        if calculation_output:
            print(f'Считая, что {first_variable} = 0, имеем:')
            print(*[f'{key} = {value}' for key,
                   value in solution.items()], sep='; ')

    def count_delta(calculation_output=False):
        nonlocal delta
        if calculation_output:
            print('Для каждой свободной клетки вычислим относительные оценки:')
        delta = np.zeros_like(C)
        for i in range(len(поставщики)):
            for j in range(len(потребители)):
                if (i, j) not in basis:
                    delta[i][j] = C[i][j] - (U[i] + V[j])
                    if count_delta:
                        print(f'Δ{i + 1}{j + 1} = {delta[i][j]};')

    def recalculate_optimal_plan():
        '''Функкия перерасчёта оптимального плана'''
        min_i, min_j = math.inf, math.inf
        lowest_mark = math.inf
        for i in range(len(поставщики)):
```

Продолжение Листинга Е.1.

```
            for j in range(len(потребители)):
                if delta[i][j] < lowest_mark:
                    lowest_mark = delta[i][j]
                    min_i = i
                    min_j = j
        basis.append((min_i, min_j))
        available_ways = copy.copy(basis)
        cycle = [(min_i, min_j)]
        curr_i = min_i
        curr_j = min_j
        while True:
            dead_end = True
            for i, j in available_ways:
                if i == curr_i and j != curr_j and ((i, j) not in cycle or (i, j) == (min_i, min_j)):
                    if len(cycle) > 1 and cycle[-2][0] != i:
                        dead_end = False
                        curr_j = j
                        break
                    elif len(cycle) == 1:
                        dead_end = False
                        curr_j = j
                        break
                elif i != curr_i and j == curr_j and ((i, j) not in cycle or (i, j) == (min_i, min_j)):
                    if len(cycle) > 1 and cycle[-2][1] != j:
                        dead_end = False
                        curr_i = i
                        break
                    elif len(cycle) == 1:
                        dead_end = False
                        curr_i = i
                        break
            if not dead_end:
                cycle.append((curr_i, curr_j))
            elif cycle[0] != cycle[-1]:
                del available_ways[available_ways.index((curr_i, curr_j))]
                curr_i = min_i
                curr_j = min_j
                cycle = [(min_i, min_j)]
            else:
                break
        lam = math.inf
        for index, point in enumerate(cycle[:-1]):
            i, j = point
            if index % 2 and X[i][j] < lam:
                lam = X[i][j]
        deleted_from_basis = 0
        for index, point in enumerate(cycle[:-1]):
            i, j = point
            if index % 2:
                X[i][j] -= lam
            else:
                X[i][j] += lam
            if X[i][j] == 0 and deleted_from_basis == 0:
                del basis[basis.index((i, j))]
                deleted_from_basis += 1

    print('Таблица с исходными данными:')
    print_table(False)
```

Продолжение Листинга Е.1.

```
    min_price_method()
    print('Начальный опорный план, полученный методом минимальной стоимости:')
    print_table()
    print(f'Стоимость перевозок по этому плану: {count_function()} единиц')
    X = np.zeros_like(C)  # Очитка массива X
    basis.clear()  # Очитка базиса
    northwest_corner_method()
    print('\033[101m' + 'Начальный опорный план, полученный методом северо-
западного угла:' + '\033[0m')
    print_table(False)
    print(f'Стоимость перевозок по этому плану: {count_function()} единиц')
    count_potentials(True)
    print_table()
    count_delta(True)
    while np.min(delta) < 0 and num_iteration < 20:
        print('Остались отрицательные оценки, произведём перерасчёт плана...')
        num_iteration += 1
        print('\033[101m' + f'Итерация № {num_iteration}:' + '\033[0m')
        recalculate_optimal_plan()
        print_table()
        print(f'Стоимость перевозок по этому плану: {count_function()} единиц')
        count_potentials(True)
        count_delta(True)
    if num_iteration == 20:
        print(f'За {num_iteration} не удалось найти оптимальное решение(')
    else:
        print('\033[101m' + 'Оптимальное решение найдено!' + '\033[0m')

def input_data():
    '''Ручной ввод данных'''
    поставщики = np.array(list(map(int, input(
        "Введите запас груза у каждого поставщика через пробел: ").split())))
    потребители = np.array(list(map(int, input(
        "Введите потребность груза у каждого потребителя через пробел: 
").split())))
    c = list()  # Коэффициенты Cij
    for i in range(len(поставщики)):
        c.append(np.array(list(map(int, input(f"Введите стоимости перевозки от {
                  i + 1}-ого поставщика к каждому потребителю через пробел: 
").split()))))
    return поставщики, потребители, c

def input_data_from_file():
    '''Автоматический ввод данных из файла'''
    with open('TPR_PRACT7.csv', encoding='utf-8') as file:
        rows = csv.reader(file)
        поставщики = np.array(list(map(int, next(rows))))
        потребители = np.array(list(map(int, next(rows))))
        c = list()  # Коэффициенты Cij
        for row in rows:
            c.append(np.array(list(map(int, row))))
    return поставщики, потребители, c

transport_task()
```