Министерство науки и высшего образования Российской Федерации
ФГБОУ ВО «Кубанский государственный технологический университет»

Кафедра информационных систем и программирования

МЕТОДЫ ОПТИМИЗАЦИИ

Методические указания по выполнению лабораторных работ
для студентов всех форм обучения
направления 09.03.04 Программная инженерия

Краснодар
2019

Составитель**:** канд. техн. наук, доцент, Янаева Марина Викторовна

Методы оптимизации: методические указания по выполнению лабораторных работ для студентов всех форм обучения направления 09.03.04 Программная инженерия. / Сост. М.В. Янаева; Кубан. гос. технол. ун-т. Кафедра информационных систем и программирования. – Краснодар. 2019. – 180 с. Режим доступа: http://moodle.kubstu.ru (по паролю).

Методические указания по выполнению лабораторных работ дисциплины «Методы оптимизации» для студентов составлены в соответствии с требованиями к обязательному минимуму содержания дисциплины, входящей в основную образовательную программу подготовки студентов направления 09.03.04 Программная инженерия государственного образовательного стандарта высшего профессионального образования, и в соответствии с рабочей программой дисциплины. Содержат краткие описания лабораторных работ, указания к их выполнению, задания и требования к оформлению отчета.

Ил.28. Библиогр.: 17 назв.

Рецензенты: Руководитель отдела телекоммуникаций Краснодарского регионального информационного центра сети «Консультант Плюс», канд.техн.наук. Н.Ф. Григорьев;
д-р. техн. наук, профессор каф. ИСП КубГТУ
В.Н. Марков

Содержание

Введение

Целью преподавания дисциплины «Методы оптимизации» является ознакомление бакалавров с теоретическими основами исследования операций и их применением в практической деятельности. Дисциплина обеспечивает совершенствование навыков, полученных при изучении основ высшей математики, вычислительной математики и программирования. Дисциплина «Методы оптимизации» ориентирована на решение практических задач, которые можно корректно описать с помощью той или иной математической модели с целью получения оптимального решения.

Дисциплина «Методы оптимизации» предназначена для освоения базовых моделей и методов принятия оптимальных решений. Для достижения поставленной цели выделяются следующие задачи курса:

– овладение основными понятиями и приемами построения математических моделей в области исследования операций;

– знакомство с основными классами задач исследования операций и методами их решения;

– получение навыков по построению моделей и применению методов решения задач исследования операций;

– формирование у бакалавров представления о сути и современном состоянии теории исследования операций;

– исследование возможности использования математических методов и моделей исследования операций при решении разнообразных теоретических и практических задач, возникающих в самых различных приложениях и связанные с управлением, алгоритмами высокопроизводительных вычислений, а также технологиями искусственного интеллекта;

– овладение способами применения математических методов для получения результатов в исследованиях или для обработки результатов исследований.

– ознакомление с достаточно полным спектром концепций, подходов, методов современной теории управления и исследования операций;

– изучение основных типов математических моделей, используемых при описании сложных систем и при принятии решений;

– обучение построению и выбору метода исследования комбинированных моделей, использующих результаты из различных научных областей;

– овладение методологией системного анализа реальных ситуаций в целях построения адекватных им моделей и методов, в целях сравнительного анализа моделей и методов, выбора наилучших в рассматриваемой ситуации решений.

Лабораторная работа
«Решение задач линейного программирования»

Цель работы: изучение принципов составления математических моделей для задач линейного программирования, получение навыков программной реализации применения данного метода на практике.

Порядок выполнения:

1. Согласно варианту составить математическую модель задачи линейного программирования.
2. Решить задачу графически.
3. Разработать программу решающую поставленную задачу на выбранном языке программирования.
4. Составить отчет согласно требованиям.

Теоритические сведения

Линейное программирование (ЛП) — это метод математического моделирования, разработанный для *оптимизации* использования *ограниченных* ресурсов. ЛП успешно применяется в военной области, индустрии, сельском хозяйстве, транспортной отрасли, экономике, системе здравоохранения и даже в социальных науках. Широкое использование этого метода также подкрепляется высокоэффективными компьютерными алгоритмами, реализующими данный метод. Задачи линейного программирования и методы их решения широко используются в промышленном производстве, экономике, логистике, военном деле и других областях целенаправленной деятельности человека.

Ограничения в модели линейного программирования

Компания производит краску для внутренних и наружных работ из сырья двух типов: M1 и M2. Следующая таблица представляет основные данные для задачи.

	Расход сырья (в тоннах) на тонну краски		Максимально возможный ежедневный расход сырья
	для наружных работ	для внутренних работ	
Сырье M1	6	4	24
Сырье M2	1	2	6
Доход (в $1000) на тонну краски	5	4	

Отдел маркетинга компании ограничил ежедневное производство краски для внутренних работ до 2 т (из-за отсутствия надлежащего спроса), а также поставил условие, чтобы ежедневное производство краски для внутренних работ не превышало более чем на тонну аналогичный показатель

производства краски для внешних работ. Компания хочет определить оптимальное (наилучшее) соотношение между видами выпускаемой продукции для максимизации общего ежедневного дохода.

Задача (модель) линейного программирования, как и любая задача исследования операций, включает три основных элемента.

1. **Переменные,** которые следует определить.
2. **Целевая функция,** подлежащая оптимизации.
3. **Ограничения,** которым должны удовлетворять переменные.

Определение переменных — первый шаг в создании модели. После определения переменных построение ограничений и целевой функции обычно не вызывает трудностей.

В нашем примере необходимо определить ежедневные объемы производства краски для внутренних и наружных работ. Обозначим эти объемы как переменные модели: x_1 — ежедневный объем производства краски для наружных работ; x_2 — ежедневный объем производства краски для внутренних работ.

Используя эти переменные, далее строим целевую функцию. Логично предположить, что целевая функция, как суммарный ежедневный доход, должна возрастать при увеличении ежедневных объемов производства красок. Обозначим эту функцию через *z* (она измеряется в тысячах долларов) и положим, что $z= 5X1+ 4X_2$. В соответствии с целями компании получаем задачу: Максимизировать $z = 5*x_1 + 4x_2$.

Итак, остался не определенным последний элемент модели — условия (ограничения), которые должны учитывать ограниченные возможности ежедневного потребления сырья и ограниченность спроса на готовую продукцию. Другими словами, ограничения на сырье можно записать следующим образом.

$$\left(\begin{array}{l}\text{Используемый объем}\\ \text{сырья для производства}\\ \text{обоих видов краски}\end{array}\right) \le \left(\begin{array}{l}\text{Максимально возможный}\\ \text{ежедневный расход сырья}\end{array}\right)$$

Из таблицы с данными имеем следующее.

Используемый объем сырья M1 = $6x1 + 4x2$ (т)

Используемый объем сырья M2 = $1x1 + 2x_2$ (т)

Так как ежедневный расход сырья M1 и M2 ограничен соответственно 24 и 6 тоннами, получаем следующие ограничения.

$6x1 + 4x_2 <= 24$ (сырье M1)

$1x1 + 2x2 <= 6$ (сырье M2)

Существует еще два ограничения по спросу на готовую продукцию: максимальный ежедневный объем производства краски для внутренних работ не должен превышать 2 т и ежедневный объем производства краски для внутренних работ не должен превышать ежедневный объем производства краски для наружных работ более чем на одну тонну. Первое ограничение простое и записывается как $x_2 <= 2$. Второе можно сформулировать так:

разность между ежедневными объемами производства красок для внутренних и наружных работ не должна превышать одной тонны, т.е. $x_2 - x_1 <= 1$.

Еще одно неявное ограничение состоит в том, что переменные x_1 и x_2 должны быть неотрицательными. Таким образом, к сформулированным выше ограничениям необходимо добавить условие не отрицательности переменных: $x_1 >= 0$, $x_2 >= 0$.

Окончательно задача будет записана следующим образом:

Максимизировать $z = 5x_1 + 4x_2$ при выполнении ограничений

$$6x_1 + 4x_2 \le 24,$$
$$x_1 + 2x_2 \le 6,$$
$$-x_1 + x_2 \le 1,$$
$$x_2 \le 2,$$
$$x_1 \ge 0, x_2 \ge 0.$$

Любое решение, удовлетворяющее ограничениям модели, является допустимым. Например, решение $x_1 = 3$ и $x_2 = 1$ будет допустимым, так как не нарушает ни одного ограничения, включая условие не отрицательности. Чтобы удостовериться в этом, подставьте значения $x_1 = 3$ и $x_2 = 1$ в левые части неравенств системы ограничений и убедитесь, что ни одно неравенство не нарушается. Значение целевой функции при этом решении будет равно z = 5x3 + 4x1 = 19 (тысяч долларов).

Итак, задача сформулирована, теперь встает вопрос о нахождении оптимального допустимого решения, доставляющего максимум целевой функции. После некоторых раздумий приходим к выводу, что задача имеет много (фактически, бесконечно много) допустимых решений. По этой причине невозможна подстановка значений переменных для поиска оптимума, т.е. нельзя применить простой перебор всех допустимых решений. Следовательно, необходима эффективная процедура отбора допустимых решений для поиска оптимального.

Графическое решение задачи линейного программирования

Графический способ решения задачи ЛП состоит из двух этапов.

1. Построение пространства допустимых решений, удовлетворяющих всем ограничениям модели.

2. Нахождение оптимального решения среди всех точек пространства допустимых решений.

Пример 1. Нахождение максимума целевой функции

Мы используем модель, построенную для предыдущей задачи по производству краски, чтобы показать оба этапа графического решения задачи ЛП.

Этап 1. Построение пространства допустимых решений.

Сначала проведем оси: на горизонтальной будут указываться значения переменной x_1 а на вертикальной — x_2 (рисунок 1). Далее рассмотрим

условие не отрицательности переменных: $x_1 >= 0$ и $x_2 >= 0$. Эти два ограничения показывают, что пространство допустимых решений будет лежать в первом квадранте. Чтобы учесть оставшиеся ограничения, проще всего заменить неравенства на равенства, в результате чего получим уравнения прямых, а затем на плоскости провести эти прямые. Например, неравенство $6x_1 + 4x_2 <= 24$ заменяется уравнением прямой $6x1 + 4x_2 = 24$. Чтобы провести эту линию, надо найти две различные точки, лежащие на этой прямой. Можно положить $x_1=0$, тогда $x_2= 24/4 = 6$. Аналогично для $x_2 = 0$ находим $x_1= 24/6 = 4$. Итак, наша прямая проходит через две точки (0, 6) и (4, 0). Эта прямая обозначена на рисунке 1 как линия (1). Теперь рассмотрим, как графически интерпретируются неравенства. Каждое неравенство делит плоскость (x_1, x_2) на два полупространства, которые располагаются по обе стороны прямой, которая, как показано выше, соответствует данному неравенству. Точки плоскости, расположенные по одну сторону прямой, удовлетворяют неравенству (допустимое полупространство), а точки, лежащие по другую сторону, — нет. «Тестовой» точкой, проверяющей, точки какого полупространства удовлетворяют неравенству, а какого— нет, может служить точка (0,0). Например, эта точка удовлетворяет первому неравенству $6x_1 + 4x_2 <= 24$ (здесь 6*0 + 4*0 = 0 <=24). Это означает, что точки полупространства, содержащего начальную точку (0, 0), удовлетворяют этому неравенству. На рисунке допустимые полупространства показаны стрелочками.

В том случае, когда точка (0,0) не удовлетворяет неравенству, допустимым полупространством будет то, которое не содержит эту точку. Если же прямая проходит через эту точку, следует в качестве «тестовой» взять какую-либо другую точку.

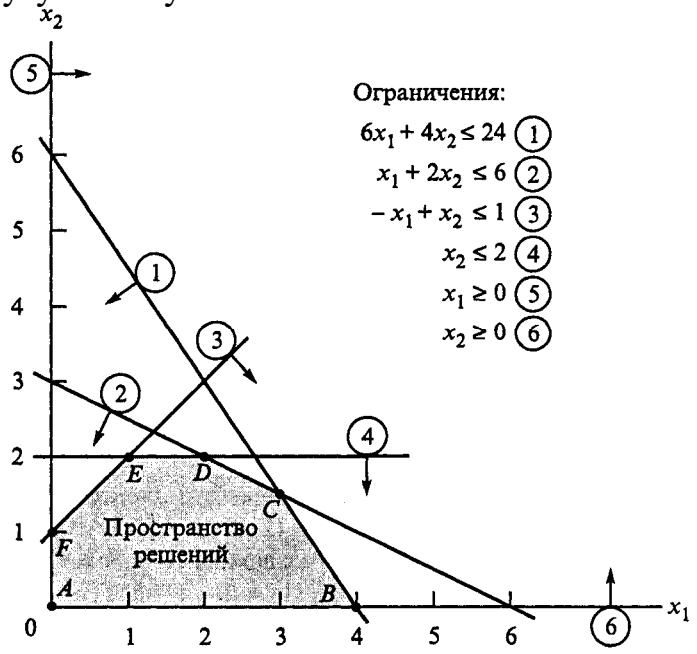


Рисунок 1 – Построение пространства допустимых решений

Этап 2. Нахождение оптимального решения.

Точки пространства допустимых решений, показанного на рисунке 2, удовлетворяют одновременно всем ограничениям. Это пространство ограничено отрезками прямых, которые соединяются в угловых точках *A, B, C, D, E* и *F.* Любая точка, расположенная внутри или на границе области, ограниченной ломаной *ABCDEF,* является допустимым решением, т.е. удовлетворяет всем ограничениям. Поскольку пространство допустимых решений содержит бесконечное число точек, необходима некая процедура поиска оптимального решения. Нахождение оптимального решения требует определения направления возрастания целевой функции $z= 5x_1 + 4x_2$ (напомним, что мы *максимизируем* функцию z). Мы можем приравнять z к нескольким возрастающим значениям, например 10 и 15. Эти значения, подставленные вместо z в выражение целевой функции, порождают уравнения прямых: для значений 10 и 15 получаем уравнения прямых $5x_1 + 4x_2 = 10$ и $5x_1 + 4x_2 = 15$. На рисунке 2 эти прямые показаны штриховыми линиями, а направление возрастания целевой функции — толстой стрелкой. Целевая функция может возрастать до тех пор, пока прямые, соответствующие возрастающим значениям этой функции, пересекают область допустимых решений. Точка пересечения области допустимых решений и прямой, соответствующей максимально возможному значению целевой функции, и будет точкой оптимума.

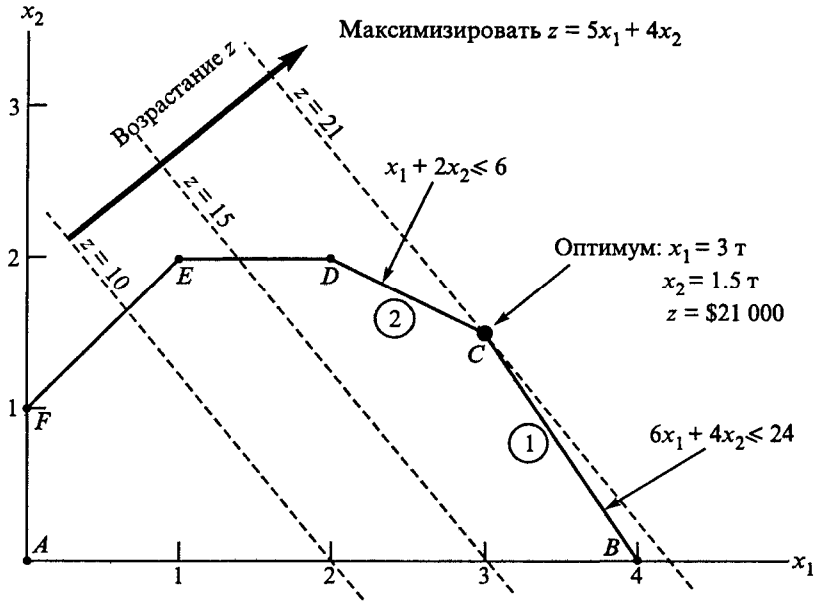


Рисунок 2 - Нахождение оптимального решения

На рисунке 2 видно, что оптимальное решение соответствует точке *C.* Эта точка является местом пересечения прямых (1) и (2), поэтому ее координаты x_1 и x_2 находятся как решение системы уравнений, задающих эти прямые:

$$6x_1 + 4x_2 = 24,$$
$$x_1 + 2x_2 = 6.$$

Решением этой системы будет $x_1 = 3$ и $x_2 = 1.5$, при этом значение целевой функции равно z= 5*3+4*1.5 = 21. Полученное решение означает,

что для компании оптимальным выбором будет ежедневное производство 3 т краски для наружных работ и 1.5 т — для внутренних работ с ежедневным доходом в $21 000.

Пример разработанной программы, решающую поставленную задачу в общем виде

1. Форма задания и ввода исходных данных – рисунок 3.

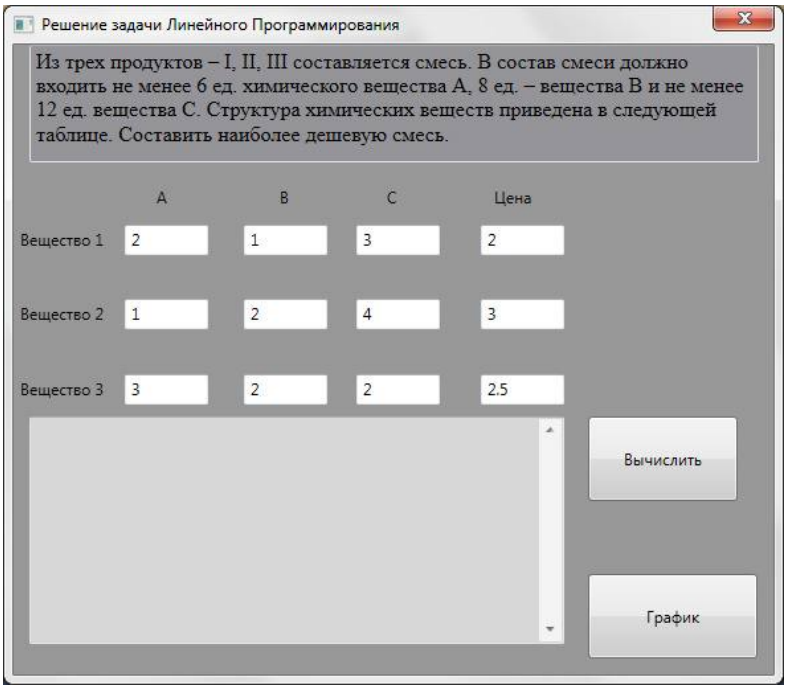


Рисунок 3 – Форма ввода исходных данных

2. Этап решения задачи приведен на рисунке 4.

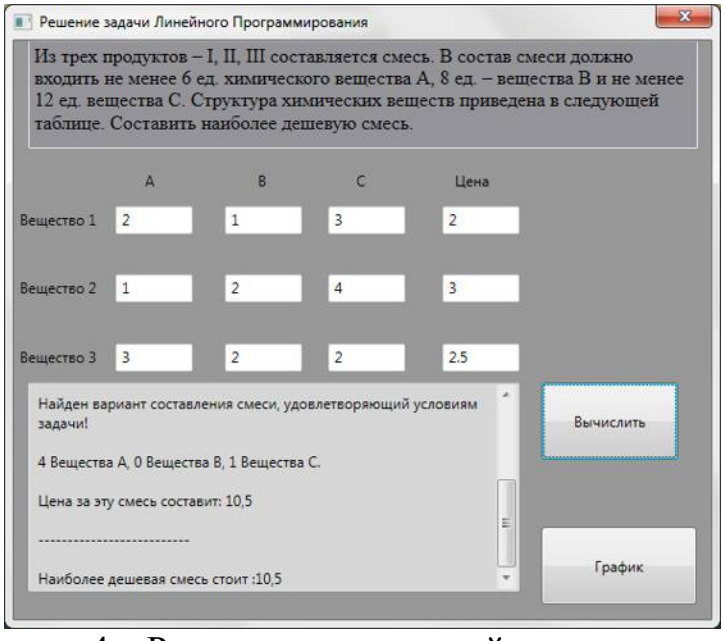


Рисунок 4 – Решение задачи линейного программирования

3. Графическая интерпретация решения задачи линейного программирования приведена на рисунке 5.

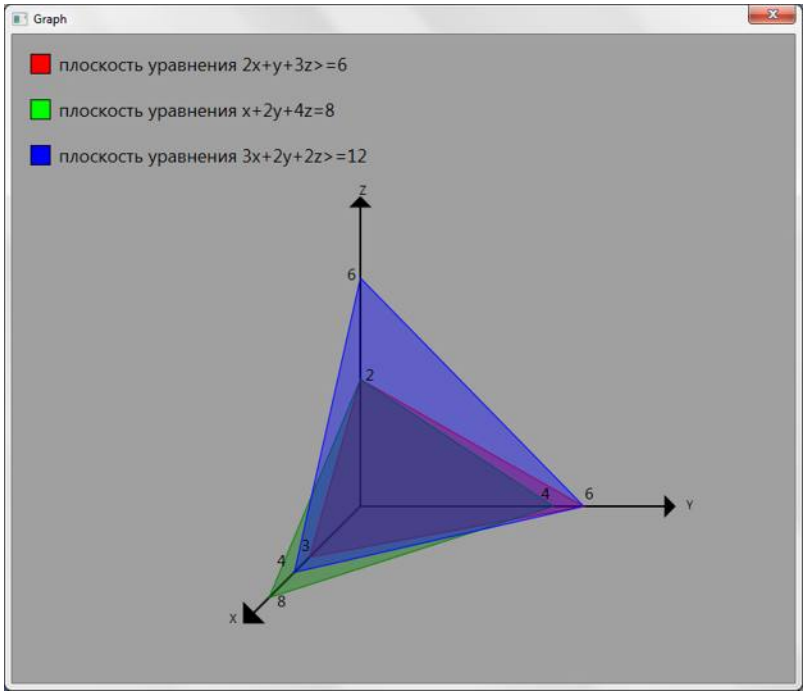


Рисунок 8 – Графическая интерпретация решения задачи

Листинг основного модуля решения задачи линейного программирования:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Navigation;
using System.Windows.Shapes;

namespace WpfApplication1
{
  public partial class MainWindow
  {
    public MainWindow()
    {
      InitializeComponent();

    }

    private void button1_Click(object sender, RoutedEventArgs e)
    {
      Window1 f = new Window1();
      f.Show();
    }
```

```
        private void button2_Click(object sender, RoutedEventArgs e)
        {
            int x;
            int y;
            int z;
            double f = 100;
            for (x = 0; x <= 6; x++)
            {
                for (y = 0; y <= 8; y++)
                {
                    for (z = 0; z <= 12; z++)
                    {
                        if ((2 * x + y + 3 * z) >= 6)
                        {
                            if ((x + 2 * y + 4 * z) == 8)
                            {
                                if ((3 * x + 2 * y + 2 * z) >= 12)
                                {
                                    if ((2 * x + 3 * y + 2.5 * z) < f)
                                    {
                                        f = 2 * x + 3 * y + 2.5 * z;
                                        richTextBox1.AppendText("Найден вариант составления смеси, удовлетворяющий
условиям задачи!" + Environment.NewLine);
                                        richTextBox1.AppendText(x + " Вещества A, " + y + " Вещества B, " + z +" Вещества
C. "+ Environment.NewLine);
                                        richTextBox1.AppendText("Цена за эту смесь составит: " + f.ToString() +
Environment.NewLine);
                                        richTextBox1.AppendText("-------------------------"+Environment.NewLine);
                                    }
                                    else
                                    {
                                        continue;
                                    }
                                }
                            }
                        }
                    }
                }
            }
            richTextBox1.AppendText("Наиболее дешевая смесь стоит :" + f.ToString());
        }

        private void Window_Loaded(object sender, RoutedEventArgs e)
        {

        }
    }
}
```

Задачи для самостоятельного решения

В лабораторной работе необходимо решить задачу линейного программирования согласно варианту графически и выполнить программную реализацию в общем виде. Отчет к лабораторной работе должен содержать:

1. Формулировку задачи.
2. Математическое решение задачи графическим методом.

3. Программную реализацию задачи в общем виде. Программная реализация должна включать помимо математического расчета графическую интерпретацию решения.

Варианты заданий:

Задача 1

В состав рациона кормления животных входят сено, силос и концентраты. Содержание питательных веществ и минимально необходимые нормы их потребления приведены в таблице. Определить рацион, стоимость которого была бы минимальной, если предельные нормы суточной выдачи се — не более 18 кг, силоса — не более 24 кг, концентратов — не более 16 кг.

	Белок	Кальций	Витамины	Цена
Сено	40	5	2	30
Силос	20	4	1	20
Концентраты	160	4	2	50
Нормы	2000	120	40	

Задача 2

Совхоз закупает корма трех видов. Цены на корма разные. В кормах содержатся питательные вещества четырех видов. Требуется так составить кормовой рацион, чтобы в нем содержалось необходимое количество питательных веществ и затраты на покупку кормов были минимальными. Данные приводятся в таблице. Каковы минимальные затраты на покупку кормов ?

Питательные вещества	Виды кормов			Нормы содержания веществ
	B1	B2	B3	
A1	2	4	6	Не менее 20
A2	3	1	0	Равно 5.28
A3	5	8	3	Не менее 25, не более 35
Цена за 1 тонну	400	200	300	

Задача 3

В аптеке продается семь наименований поливитаминов. Каждое наименование содержит витамины трех различных типов. Цены на витамины различны. Необходимо пройти профилактический курс, в течение которого с минимальными суммарными затратами получить необходимое количество единиц витаминов. Необходимое количество поливитаминов покупается одновременно. Каковы минимальные затраты на профилактический курс?

Витамины	Содержание витаминов							Всего
	P1	P2	P3	P4	P5	P6	P7	необходимо
A	5	0	2	0	3	1	2	100
C	3	1	5	0	2	0	1	80
B_6	1	0	3	1	2	0	6	120

Задача 4

"Южная алкогольная компания" импортирует смеси трех сортов виски — ирландского, шотландского и канадского. Смешивают их согласно рецептам, устанавливающим максимум или минимум процентного содержания ирландского и канадского виски в каждой смеси, как показано в первой таблице. Запасы трех основных видов виски и их стоимость показаны ниже во второй таблице. Какова максимальная прибыль в день?

Смесь	Спецификация	Цена за 1л смеси
«Старый Джек»	Не меньше, чем 60% ирландского Не больше, чем 20% канадского	68
«Специальное»	Не меньше, чем 60% канадского Не больше, чем 15% ирландского	57
«Юный Френзи»	Не больше, чем 50% канадского	45

Сорт виски	Наличие	Стоимость
Ирландское	2000	70
Шотландское	1500	50
Канадское	1200	40

Задача 5

Мощности завода по производству удобрений позволяют произвести в текущем месяце 1000 т нитратов, 1800 т фосфатов и 1200 т поташа. В результате смешения этих активных ингредиентов с инертными, запасы которых не ограничены, могут быть получены три типа удобрений. В таблице указано содержание активных ингредиентов (нитратов, фосфатов и поташа) в смеси. Цена инертных ингредиентов составляет 5 тыс.р./т. Затраты смешения, упаковки и продажи составляют 15 тыс.р./т для каждого типа удобрении. Существует соглашение о поставке 6000 т удобрений типа 1. Какова максимальная прибыль?

Тип удобрений	Процентное содержание активных ингредиентов			Цена
	нитраты	фосфаты	поташ	
1	5	10	5	40
2	5	10	10	50
3	10	10	10	60
Цена	160	40	100	

Задача 6

На кондитерской фабрике изготовляют три вида восточных сладостей, для которых используют миндаль, фундук и арахис. Миндаль покупается по цене 6,5 тыс.р. за кг , фундук — 2,5 тыс.р., [a] арахис — 3,5 тыс.р.

Продукт I должен содержать не менее 50% миндаля и не бо-[œ] 25% фундука, продукт 2 — не менее 25% миндаля и не более

50% фундука, продукт 3 может содержать любое количество миндаля, фундука и арахиса. Продажная цена продукта 1 — 5 тыс.р. зц кг, продукта 2 — 3,5 тыс.р., продукта 3 — 2,5 тыс.р. за кг. Запасы сырья ограничены, миндаля — 100 кг, фундука — 100 кг, арахиса — 60 кг. Какова максимальная прибыль?

Зпдача 7

Из 500 листов железа первого размера и 300 листов железа второго размера несколькими способами выкраиваются три вида деталей. Даны нормы одновременного выхода деталей по различным способам. Определите максимальное число комплектов деталей, если комплект состоит из четырех деталей вида I, трех деталей вида 2 и двух деталей вида 3.

	Листы размера 1			Листы размера 2	
	Способы раскроя				
Вид детали	1	2	3	1	2
	Количество деталей				
1	0	2	9	6	5
2	4	3	4	5	4
3	10	16	0	8	0

Задача 8

При раскрое деталей единственного изделия на швейной фабрике используются два артикула ткани. Изделие собирается из двух деталей, причем каждая из этих деталей может быть получена путем раскроя ткани любого типа. Ткани

можно раскраивать тремя способами, выход деталей каждого типа указан в следующей таблице. На фабрику ткани 1 поступает в два раза больше (по длине), чем ткани 2. Выход готовых изделий должен быть максимальным.

Способ раскроя	Ткань 1		Ткань 2	
	1 тип детали	2 тип детали	1 тип детали	2 тип детали
1	8	0	12	0
2	0	3	0	4
3	4	1	6	2

Задача 9

Фирма производит и продает столы и шкафы из древесины хвойных и лиственных пород. Расход каждого вида в кубометрах на каждое изделие задан в таблице. Определите количество столов и шкафов, которое следует поставлять на продажу для получения максимального дохода фирмы.

	Расход древесины		Цена изделия
	хвойные	лиственные	
Стол	0,15	0,2	0,8
Шкаф	0,3	0,1	1,5
Запасы древесины	75	40	

Задача 10

Из трех продуктов – I, II, III составляется смесь. В состав смеси должно входить не менее 6 ед. химического вещества А, 8 ед. – вещества В и не менее 12 ед. вещества С. Структура химических веществ приведена в следующей таблице. Составить наиболее дешевую смесь.

Продукт	Содержание хим. вещества в 1 ед. продукции			Стоимость 1 ед. продукции
	A	B	C	
I	2	1	3	2
II	1	2	4	3
III	3	2	2	2,5

Задача 11

Фирма решила открыть на основе технологии производства чешского стекла, фарфора и хрусталя линию по изготовлению ваз и графинов и их декорирование. Затраты сырья на производство этой продукции представлены в таблице. Определите объем выпуска продукции, обеспечивающий максимальный доход от продаж, если спрос на вазы не превышает 200 шт. в неделю.

Сырье	Расход сырья на производство		Поставки сырья в неделю
	ваза	графин	
Кобальт	20	18	3
Сусальное 24-каратное золото	13	10	1,2
Оптовая цена,	700	560	

Задача 12

Кондитерская фабрика в Покрове освоила выпуск новых видов шоколада "Лунная ночь" и "Малиновый дождь", спрос на которые составляет соответственно не более 10 тонн и 7,7 тонны в месяц. По причине занятости трех цехов выпуском традиционных видов шоколада, каждый цех может выделить только ограниченный ресурс времени в месяц. В силу специфики технологического оборудования затраты времени на производство шоколада разные и представлены в таблице. Определить объем выпуска шоколада, обеспечивающий максимальную выручку от продажи.

Номер цеха	Время на производство шоколада		Время, отведенное цехами под производство
	"Лунная ночь"	"Малиновый дождь"	
I	1	7	56
II	2	3	35
Оптовая цена	8000	6000	

Задача 13

Фирма производит одежду для охотников, туристов и охранных структур. Дополнительно фирма решила изготавливать шапки и подстежки из натурального меха. Затраты на производство этих изделий и запасы сырья представлены в таблице. Спрос на шапки составляет не более 600 шт. в месяц, а

подстежек – не более 400 шт. в месяц. Определить объемы производства этих изделий, обеспечивающих максимальный доход от продажи.

Сырье	Расход сырья на производство, дм.		Средний запас в месяц, дм.
	шапки	подстежки	
Мех	22	140	61500
Ткань	1,5	30	15000
Оптовая цена,	410	840	

Задача 14

Коммерческие расчеты, проведенные студентами в деревне, привели к более выгодному использованию плодов яблок и груш путем их засушки и последующей продажи зимой в виде смеси сухофруктов, варианты которых представлены в таблице. Подсчитано, что из 1 кг. плодов получается 200 г. сушеных яблок, а груш – 250 г. Определить оптимальное количество упаковок сухофруктов по 1 кг. смесей первого и второго вида, обеспечивающие максимальный доход от продажи.

Плоды	Вес в 1 кг. сухофруктов		Сбор плодов, кг/день.
	смесь 1	смесь 2	
Анис (яблоки)	0,25	0,25	14
Штрейфлинг (яблоки)	0,75	0,25	22,5
Груши	0	0,5	12
Оптовая цена упаковки	40	50	

Задача 15

Предприятие имеет собственную котельную. Для ее работы требуется закупить топливо в количестве, удовлетворяющем потребность в тепле, составляющую 4000 гкал. Затраты на закупку должны быть минимальны. Общее количество золы после сжигания топлива не должно превысить 56 тонн. На рынке имеются три сорта топлива, различающиеся теплотворной способностью, зольностью и ценой. Все сведения о них даны в таблице.

Сорт топлива	1-ый	2-ой	3-ий
Зольность	0,02	0,04	0,06

Теплотворная способность	10	8	4
Стоимость	10	6	2

Задача 16

Конкуренция приводит к необходимости торговым предприятиям заниматься еще и выпуском продукции собственного производства, например, салатов, пиццы и т.п. Нормы затрат на производство разных видов пиццы, объемы ресурсов и стоимость приведены в таблице. Определить, в каком количестве следует производить пиццы, чтобы доход был максимальным. Тесто нужно использовать полностью, так как при хранении оно теряет свои вкусовые качества.

Продукты	Нормы затрат на изготовление 100 шт. пиццы			Запасы продуктов
	ассорти	грибная	салями	
Грибы	6	7	2	20
Колбаса	5	2	8	18
Тесто	10	8	6	25
Цена за 100 шт	9	6	5	

Задача 17

Предприятие занимается «отверточной» сборкой бытовой электронной аппаратуры трех наименований: телевизоров, стереосистем и акустических систем. Замечено, что спрос на стереосистемы в два раза выше, чем на акустические системы. Склад предприятия вмещает ограниченное количество компонентов: кинескопов, громкоговорителей, источников питания, комплектов радиодеталей. Считая, что на момент расчета склад заполнен до отказа и зная, какую прибыль можно получить от единицы каждого изделия, необходимо определить программу выпуска, приносящую максимальную общую прибыль.

Название детали	Количество на складе	Потребность на 1 изделие		
		телевизоры	стереосистемы	акустические системы
Кинескоп	250	1	0	0
Громкоговоритель	800	2	2	1
Источник питания	450	1	1	0
Комплект	600	2	1	1

радиодеталей				
Прибыль от 1 изделия		75	50	35

Задача 18

На масложиркомбинате открывается новый цех по производству майонеза площадью 1800 м2. Количество обслуживающего персонала в новом цехе равно 140 человек, они будут работать в две смены. Максимально доступное количество воды и электроэнергии в сутки составляет, соответственно, 150 м3. и 800 кВт/ч. Требуется установить оборудование в цехе таким образом, чтобы не осталось ни одного бесполезного уголка, при этом общая производительность цеха была максимальной. На выбор предложены 4 вида производственных линий с различными техническими характеристиками.

Показатель	Номера линий по производству майонеза			
	1	2	3	4
Производительность, литров в сутки	150	300	600	1000
Занимаемая площадь	48	54	60	72
Количество обслуживающего персонала	2	2	3	3
Потребление ледяной воды,	2	4	5	7
Потребление электроэнергии	14	17	25	37

Задача 19

Фирма "Лесоруб" располагает участком леса площадью в 300 га, который состоит из двух выделов, отличающихся типом древостоя (смешанные сосновые древостои и смешанные лиственные древостои). Планируемый 60-летний горизонт проведения лесозаготовок на этом участке разделен на 3 промежутка по 20 лет каждый. Имеется таблица хода роста древостоев. Цель управления вырубками леса заключается в максимизации выхода ликвидной древесины за планируемые 60 лет. Лесозаготовки выполняются по сплошносечной технологии с учетом необходимости сохранения живой первозданной природы. Вырубаемая лесосека для каждого типа древостоя в каждый период обеспечивает вегетационную структуру для обитателей. Поэтому биологи требуют, чтобы максимальная вырубаемая площадь по каждому типу древостоя за каждый

период составляла: для древостоя 1 типа – 40 га, 2 типа – 90 га. Желательно рубить лес равномерно, то есть обеспечить одинаковый объем расчетной лесосеки в каждом периоде.

Тип древостоя	Площадь, га	Запас вырубаемой древесины в будущем, м3/га, через		
		0-20 лет	21-40 лет	41-60 лет
1	100	3	10	30
2	200	12	17	20

Задача 20

Туристская фирма располагает флотилией из трех типов судов, характеристики которых представлены в таблице. В месяц выделяется 70 000 т. горючего. Наличие рабочей силы не превышает 1100 чел. Определите количество судов каждого типа, отправляющихся в круиз, чтобы при минимальных затратах обеспечить путевками 10 тыс. желающих

Показатели	Судно		
	Буревестник	Альбатрос	Чайка
Пассажировместимость	3000	2000	1000
Расход горючего	20000	12000	7000
Экипаж	350	250	100
Затраты на содержание и обслуживание	10	8	5

Лабораторная работа
«Симплекс-метод»

Цель работы: научится находить оптимальное решения для линейной функции при заданных ограничениях симплекс-методом, а также применение данного метода на практике.

Порядок выполнения:

1. Составить математическую модель для своего варианта;
2. Решить свою модель симплекс-методом
3. Разработать программу решающую поставленную задачу в общем виде;
4. Составить отчет по работе.

Теоретические сведения

Графический способ решения задачи линейного программирования (ЛП) из лабораторной работы №1 показывает, что оптимальное решение этой задачи всегда ассоциируется с угловой точкой пространства решений (в математике она также называется крайней точкой множества). Это является ключевой идеей при разработке общего алгебраического симплекс-метода для решения любой задачи линейного программирования.

Переход от геометрического способа решения задачи ЛП к симплекс-методу лежит через алгебраическое описание крайних точек пространства решений. Для реализации этого перехода сначала надо привести задачу ЛП к стандартной форме, преобразовав неравенства ограничений в равенства путем введения дополнительных переменных.

Стандартная форма задачи ЛП необходима, потому что она позволяет получить базисное решение (используя систему уравнений, порожденную ограничениями). Это (алгебраическое) базисное решение полностью определяет все (геометрические) крайние точки пространства решений. Симплекс-метод позволяет эффективно найти оптимальное решение среди всех базисных.

Стандартная форма задачи ЛП

Стандартная форма записи задачи ЛП предполагает выполнение следующих требований:

1. Все ограничения (включая ограничения неотрицательности переменных) преобразуются в равенства с неотрицательной правой частью.

2. Все переменные неотрицательные.

3. Целевую функцию следует или максимизировать, или минимизировать.

Преобразование неравенства

Неравенства любого типа (со знаками неравенств < или >) можно преобразовать в равенства путем добавления в левую часть неравенств дополнительных переменных — остаточных или избыточных.

Пример неравенства типа " ≤ ". Неравенство $x_1 + 2x_2 \leq 3$ эквивалентно равенству $x_1 + 2x_2 + r_1 \leq 3$, где r_1, — остаточная переменная и $r_1 \geq 0$.

Пример неравенства типа " ≥ ". Неравенство $3x_1 + 4x_2 \geq 5$ эквивалентно равенству $3x_1 + 4x_2 + r_2 \geq 5$, где r_2, — избыточная переменная и $r_2 \geq 0$.

Определение базисных решений

Задача линейного программирования, записанная в стандартной форме, содержи m линейных равенств с n неизвестными переменными (m < n). Разделим n переменных на два множества: (1) n – m переменные, которые положим равны 0, и (2) оставшиеся m переменные, значения которых определяются как решение системы из m линейных уравнений. Если это решение единственное, тогда соответствующие m переменные называются базисными, а остальные n – m нулевые переменные – небазисные. В этом случае результирующие значения переменных составляют базисное решение. Если все переменные принимают неотрицательные значения, то такое базисное решение является допустимым. В противном случае — недопустимым.

Основываясь на этих определениях, нетрудно подсчитать, что количество всех положительных базисных решений для m уравнений с n неизвестными не превосходит:

$$\binom{n}{m} = \frac{n!}{m!\,(n-m)!}$$

Алгоритм симплекс-метода

Основываясь на определениях данные выше, мы можем найти оптимальное решение задачи линейного программирования, записанной в стандартной форме, путем простого перебора всех базисных (допустимых) решений. Но, конечно, такая процедура не эффективна. Алгоритм симплекс-метода находит оптимальное решение, рассматривая ограниченное количество допустимых базисных решений.

Алгоритм симплекс-метода всегда начинается с некоторого допустимого базисного решения и затем пытается найти другое допустимое базисное решение, “улучшающее” значение целевой функции. Это возможно только в том случае, если возрастание какой-либо нулевой (небазисной) переменной ведет к улучшению значения целевой функции.Но для того, чтобы небазисная переменная стала полижите, надо одну из текущих базисных переменных сделать нулевой, т.е. перевести в небазисные. Это необходимо, чтобы новое решение содержало в точности m базисных переменных. В соответствии с терминологией симплекс-метода выбранная нулевая переменная называется вводимой (в базис), а удаляемая базисная переменная — исключаемой (из базиса).

Существуют два правила выбора вводимых и исключающих переменных в симплекс-методе назовем условием оптимальности и условием допустимости. Сформулируем эти правила, а также рассмотрим последовательность действий, выполняемых при реализации симплекс-метода.

Условие оптимальности. Вводимой переменной в задаче максимизации (минимизации) является небазисная переменная, имеющая наибольший отрицательный (положительный) коэффициент в 2-строке. Если в 2-строке есть несколько таких коэффициентов, то выбор вводимой переменной делается произвольно. Оптимальное решение достигнуто тогда, когда в 2-строке все коэффициенты при небазисных переменных будут неотрицательными (неположительными).

Условие допустимости. Как в задаче максимизации, так и в задаче минимизации в качестве исключаемой выбирается базисная переменная, для которой отношение значения правой части ограничения к положительному коэффициенту ведущего столбца минимально. Если базисных переменных с таким свойством несколько, то выбор исключаемой переменной выполняется произвольно.

Теперь приведем последовательность действий, выполняемых в симплекс-методе.

Шаг 0. Находится начальное допустимое базисное решение.

Шаг 1. На основе условия оптимальности определяется вводимая переменная.

Если вводимых переменных нет, вычисления заканчиваются.

Шаг 2. На основе условия допустимости выбирается исключаемая переменная.

Шаг 3. Методом Гаусса-Жордана вычисляется новое базисное решение. Переход к шагу 1.

Вычисления в симплекс-методе выполняются итерационно в том смысле, что условия оптимальности и допустимости, а также вычисления применяются к текущей симплекс-таблице, в результате чего получается следующая таблица. Мы будем называть последовательные симплекс-таблицы итерациями.

Пример решения математической модели симплекс-методом

Найдем наибольшее значение линейной функции

$$L = x_1 + 2x_2 + x_3 - x_4$$

При следующих ограничениях:

$$-x_1 + 5x_2 + x_3 + x_4 + x_5 = 10$$

$$-2x_1 + x_2 - x_3 + 3x_4 = -6$$

$$10x_2 + x_3 + 2x_4 + 3x_5 = 25$$

Нам необходимо найти начальное опорное (абсолютно произвольное) решение для функции L, которое бы удовлетворяло системе наложенных ограничений. Далее, применяя симплекс таблицы, мы будем получать решения, при которых значение функции будет, как минимум, не убывать. И так до тех пор, пока не достигнем оптимально решения, при котором функция достигает своего максимума. Если, конечно, рассматриваемая нами линейная функция обладаем максимальным значением при заданной системе ограничений. Обратите внимание, что хесли нам нужно найти наименьшее значение линейной функции, то максимальное значение рассматриваемой функции равно наименьшему значению исходной функции по модулю, но значения противоположны по знаку. Другими словами, получившийся ответ мы должны будем умножить на -1.

Перед применением симплекс таблиц, необходимо преобразовать систему линейных ограничений и рассматриваемую нами функцию L к вполне определенному виду.

Свободные члены системы ограничений должны быть неотрицательными, поэтому умножим коэффициенты ограничения 2 на -1, свободный член ограничения станет положительным.

$$-x_1 + 5x_2 + x_3 + x_4 + x_5 = 10$$

$$2x_1 - x_2 + x_3 - 3x_4 = 6$$

$$10x_2 + x_3 + 2x_4 + 3x_5 = 25$$

Система ограничений должна быть приведена к каноническому виду. В нашем случае система ограничений уже приведена к каноническому виду, т.е. все условия системы представляют собой уравнения.

Определимся с начальным опорным решением. Наличие единичного базиса в системе ограничений позволяет легко найти начальное опорное решение. Рассмотрим подробнее:

1. В уравнении 1 нет переменной, которая входила бы в него с коэффициентом 1, а в остальные уравнения системы входила бы с коэффициентом ноль. Добавим к данному уравнению искусственную переменную r_1. Очевидно, переменная r_1 будет являться базисной переменной, т.к. входит в уравнение 1 с коэффициентом 1 и не входит в оставшиеся уравнения системы ограничений.

2. В уравнении 2 нет переменной, которая входила бы в него с коэффициентом 1, а в остальные уравнения системы входила бы с коэффициентом ноль. Добавим к данному уравнению искусственную переменную r_2. Очевидно, переменная r_2 будет являться базисной переменной, т.к. входит в уравнение 2 с коэффициентом 1 и не входит в оставшиеся уравнения системы ограничений.

3. В уравнении 3 нет переменной, которая входила бы в него с коэффициентом 1, а в остальные уравнения системы входила бы с коэффициентом ноль. Добавим к данному уравнению искусственную переменную r_3. Очевидно, переменная r_3 будет являться базисной переменной, т.к. входит в уравнение 3 с коэффициентом 1 и не входит в оставшиеся уравнения системы ограничений.

$$-x_1 + 5x_2 + x_3 + x_4 + x_5 + r_1 = 10$$
$$2x_1 - x_2 + x_3 - 3x_4 + r_2 = 6$$
$$10x_2 + x_3 + 2x_4 + 3x_5 + r_3 = 25$$

Переменные, которые не являются базисными называются свободными переменными. Приравняв свободные переменные нулю в получившийся системе ограничений мы получим начальное решение.

$$X_{\text{нач}} = (0, 0, 0, 0, 0, 10, 6, 25)$$

Для получения единичного базиса нам пришлось ввести искусственные переменные, поэтому наше начальное решение является псевдорешением. Для нахождения начального опорного решения функции L, сначала придется решить вспомогательную задачу. Введем в рассмотрение вспомогательную функцию W :

$$W = -r_1 - r_2 - r_3$$

Найдем наибольшее значение функции W. В процессе решения данной задачи возможны два варианта. Если максимальное значение вспомогательной функции W равно нулю, т.е. все искусственные переменные обращаются в нуль - это будет свидетельствовать о том, что мы нашли начальное опорное решение функции L. В противном случае, не существует решений, удовлетворяющих системе ограничений нашей задачи.

Функция L и вспомогательная функция W не должны содержать базисных переменных. Из уравнения 1 последней системы выразим r_1 и подставим в выражение функции W, получим

$$W = -10 - x_1 + 5x_2 + x_3 + x_4 + x_5 - r_2 - r_3$$

Из уравнения 2 последней системы выразим r_2 и подставим в выражение функции W, получим

$$W = -16 + x_1 + 4x_2 + 2x_3 - 2x_4 + x_5 - r_3$$

Из уравнения 3 последней системы выразим r_3 и подставим в выражение функции W, получим

$$W = -41 + x_1 + 14x_2 + 3x_3 + 4x_5$$

Значение функции W для начального решения:

$$W(X_{\text{нач}}) = -41$$

Функция L и вспомогательная функция W не содержат базисных переменных. Для составления начальной симплекс таблицы мы выполнили все условия.

При составлении исходной симплекс таблицы, коэффициенты при переменных функции L записываются с противоположными знаками, а свободный член со своим знаком. Для функции W правило аналогичное.

Составим исходную симплекс таблицу. За ведущий выберем столбец 2, так как -14 наименьший элемент в W строке. Элемент W строки, принадлежащий столбцу свободных членов не рассматриваем. За ведущую выберем строку 1, так как отношение свободного члена к соответствующему элементу выбранного столбца для 1 строки является наименьшим. Обратите внимание, что отношение мы вычисляем только для положительных элементов столбца 2.

Базисные переменные	x_1	x_2	x_3	x_4	x_5	r_1	r_2	r_3	Свободные члены	Отношение
r_1	-1	5	1	1	1	1	0	0	10	2
r_2	2	-1	1	-3	0	0	1	0	6	-
r_3	0	10	1	2	3	0	0	1	25	5/2
L	-1	-2	-1	1	0	0	0	0	0	-
W	-1	-14	-3	0	-4	0	0	0	-41	-

Разделим элементы строки 1 на 5.

Базисные переменные	x_1	x_2	x_3	x_4	x_5	r_1	r_2	r_3	Свободные члены
r_1	-1/5	1	1/5	1/5	1/5	1/5	0	0	2

r_2	2	-1	1	-3	0	0	1	0	6
r_3	0	10	1	2	3	0	0	1	25
L	-1	-2	-1	1	0	0	0	0	0
W	-1	-14	-3	0	-4	0	0	0	-41

От элементов строки 2 отнимает соответствующие элементы строки 1, умноженные на -1. От элементов строки 3 отнимает соответствующие элементы строки 1, умноженные на 10. От элементов строки L отнимает соответствующие элементы строки 1, умноженные на -2. От элементов строки W отнимает соответствующие элементы строки 1, умноженные на -14. Элементы столбца r_1 можно не пересчитывать, так как переменная r_1 больше не является базисной, x_2 теперь базисная переменная.

Базисные переменные	x_1	x_2	x_3	x_4	x_5	r_2	r_3	Свободные члены
x_2	-1/5	1	1/5	1/5	1/5	0	0	2
r_2	9/5	0	6/5	-14/5	1/5	1	0	8
r_3	2	0	-1	0	1	0	1	5
L	-7/5	0	-3/5	7/5	2/5	0	0	4
W	-19/5	0	-1/5	14/5	-6/5	0	0	-13

$X_1 = (0, 2, 0, 0, 0, 8, 5)$

$W = -13 + \frac{19}{5}x_1 + \frac{1}{5}x_3 - \frac{14}{5}x_4 + \frac{6}{5}x_5$

Значение функции W для данного решения:

$W(X_1) = -13$

За ведущий выберем столбец 1 , так как -19/5 наименьший элемент в W строке. Элемент W строки, принадлежащий столбцу свободных членов не рассматриваем. За ведущую выберем строку 3, так как отношение свободного члена к соответствующему элементу выбранного столбца для 3 строки является наименьшим. Обратите внимание, что отношение мы вычисляем только для положительных элементов столбца 1.

Базисные переменные	x_1	x_2	x_3	x_4	x_5	r_2	r_3	Свободные члены	Отношение
x_2	-1/5	1	1/5	1/5	1/5	0	0	2	-
r_2	9/5	0	6/5	-14/5	1/5	1	0	8	40/9

r_3	2	0	-1	0	1	0	1	5	5/2
L	-7/5	0	-3/5	7/5	2/5	0	0	4	-
W	-19/5	0	-1/5	14/5	-6/5	0	0	-13	-

Разделим элементы строки 3 на 2

Базисные переменные	x_1	x_2	x_3	x_4	x_5	r_2	r_3	Свободные члены
x_2	-1/5	1	1/5	1/5	1/5	0	0	2
r_2	9/5	0	6/5	-14/5	1/5	1	0	8
r_3	1	0	-1/2	0	1/2	0	1/2	5/2
L	-7/5	0	-3/5	7/5	2/5	0	0	4
W	-19/5	0	-1/5	14/5	-6/5	0	0	-13

От элементов строки 1 отнимает соответствующие элементы строки 3, умноженные на -1/5. От элементов строки 2 отнимает соответствующие элементы строки 3, умноженные на 9/5. От элементов строки L отнимает соответствующие элементы строки 3, умноженные на -7/5. От элементов строки W отнимает соответствующие элементы строки 3, умноженные на -19/5. Элементы столбца r3 можно не пересчитывать, так как переменная r3 больше не является базисной, x1 теперь базисная переменная.

Базисные переменные	x_1	x_2	x_3	x_4	x_5	r_2	Свободные члены
x_2	0	1	1/10	1/5	3/10	0	5/2
r_2	0	0	21/10	-14/5	-7/10	1	7/2
x_1	1	0	-1/2	0	1/2	0	5/2
L	0	0	-13/10	7/5	11/10	0	15/2
W	0	0	-21/10	14/5	7/10	0	-7/2

$$X_2 = (\frac{5}{2}, \frac{5}{2}, 0, 0, 0, \frac{7}{2})$$

$$W = -\frac{7}{2} + \frac{21}{10}x_3 - \frac{14}{5}x_4 - \frac{7}{10}x_5$$

Значение функции W для данного решения:

$$W(X_2) = -\frac{7}{2}$$

За ведущий выберем столбец 3 , так как -21/10 наименьший элемент в W строке. Элемент W строки, принадлежащий столбцу свободных членов не рассматриваем. За ведущую выберем строку 2, так как отношение

свободного члена к соответствующему элементу выбранного столбца для 2 строки является наименьшим. Обратите внимание, что отношение мы вычисляем только для положительных элементов столбца 3.

Базисные переменные	x_1	x_2	x_3	x_4	x_5	r_2	Свободные члены	Отношение
x_2	0	1	1/10	1/5	3/10	0	5/2	25
r_2	0	0	21/10	-14/5	-7/10	1	7/2	5/3
x_1	1	0	-1/2	0	1/2	0	5/2	-
L	0	0	-13/10	7/5	11/10	0	15/2	-
W	0	0	-21/10	14/5	7/10	0	-7/2	-

Разделим элементы строки 2 на 21/10.

Базисные переменные	x_1	x_2	x_3	x_4	x_5	r_2	Свободные члены
x_2	0	1	1/10	1/5	3/10	0	5/2
r_2	0	0	1	-4/3	-1/3	10/21	5/3
x_1	1	0	-1/2	0	1/2	0	5/2
L	0	0	-13/10	7/5	11/10	0	15/2
W	0	0	-21/10	14/5	7/10	0	-7/2

От элементов строки 1 отнимает соответствующие элементы строки 2. От элементов строки 3 отнимает соответствующие элементы строки 2, умноженные на -1/2. От элементов строки L отнимает соответствующие элементы строки 2, умноженные на -13/10. т элементов строки W отнимает соответствующие элементы строки 2, умноженные на -21/10. Элементы столбца r_2 можно не пересчитывать, так как переменная r_2 больше не является базисной .

Базисные переменные	x_1	x_2	x_3	x_4	x_5	Свободные члены
x_2	0	1	0	1/3	1/3	7/3
x_3	0	0	1	-4/3	-1/3	5/3
x_1	1	0	0	-2/3	1/3	10/3
L	0	0	0	-1/3	2/3	29/3
W	0	0	0	0	0	0

$$X_3 = (\frac{10}{3}, \frac{7}{3}, \frac{5}{3}, 0, 0)$$
$$W = 0$$

Значение функции W для данного решения:

$W(X_3) = 0$

Строка W нам больше не нужна. Мы нашли начальное опорное решение функции L .

$X_{\text{нач}} = (\frac{10}{3}, \frac{7}{3}, \frac{5}{3}, 0, 0)$

$\square = \frac{29}{3} + \frac{1}{3}x_4 - \frac{2}{3}x_5$

Значение функции L для данного решения:

$L(X_{\text{нач}}) = \frac{29}{3}$

За ведущий выберем столбец 4, так как -1/3 наименьший элемент в L строке. Элемент L строки, принадлежащий столбцу свободных членов не рассматриваем. За ведущую выберем строку 1, так как отношение свободного члена к соответствующему элементу выбранного столбца для 1 строки является наименьшим. Обратите внимание, что отношение мы вычисляем только для положительных элементов столбца 4.

Базисные переменные	x_1	x_2	x_3	x_4	x_5	Свободные члены	Отношение
x_2	0	1	0	1/3	1/3	7/3	7
x_3	0	0	1	-4/3	-1/3	5/3	-
x_1	1	0	0	-2/3	1/3	10/3	-
L	0	0	0	-1/3	2/3	29/3	-

Разделим элементы строки 1 на 1/3.

Базисные переменные	x_1	x_2	x_3	x_4	x_5	Свободные члены
x_2	0	3	0	1	1	7
x_3	0	0	1	-4/3	-1/3	5/3
x_1	1	0	0	-2/3	1/3	10/3
L	0	0	0	-1/3	2/3	29/3

От элементов строки 2 отнимает соответствующие элементы строки 1, умноженные на -4/3. От элементов строки 3 отнимает соответствующие элементы строки 1, умноженные на -2/3. От элементов строки L отнимает соответствующие элементы строки 1, умноженные на -1/3.

Базисные переменные	x_1	x_2	x_3	x_4	x_5	Свободные члены
x_4	0	3	0	1	1	7

x_3	0	4	1	0	1	11
x_1	1	2	0	0	1	8
L	0	1	0	0	1	12

Учитывая, что все $x_i \geq 0$, по условию задачи, наибольшее значение функции L равно свободному члену 12, т.е. мы получили оптимальное решение.

$X_{\text{опт}} = (8,0,11,7,0)$

$L = 12 - x_2 - x_5$

Значение функции L для данного решения:

$L(X_{\text{опт}}) = 12$

Пример разработанной программы, решающую поставленную задачу в общем виде

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

namespace Симплекс_метод
{
   public partial class Form1 : Form
   {
      public Form1()
      {
         InitializeComponent();
      }

      float[,] Xarray = new float[3, 9]; //Коэф при ограничениях
      float[] Larray = new float[9]; //Коэф линейной функции
      float[] Warray = new float[9]; //Вспомогательная функция
      List<int> NoNeed = new List<int>(); //столбец, которые не учитывать
      int[,] NewBazis = new int[3,5]; //Новый базис


      private void button1_Click(object sender, EventArgs e)
      {
         //линейная функция
         //коэффициенты при переменных функции L записываются с противоположными знаками
         Larray[0] = -1*Convert.ToInt32(L1.Text);
         Larray[1] = -1 * Convert.ToInt32(L2.Text);
         Larray[2] = -1 * Convert.ToInt32(L3.Text);
         Larray[3] = -1 * Convert.ToInt32(L4.Text);
         Larray[4] = -1 * Convert.ToInt32(L5.Text);

         //исскуственные базисные переменные
         Xarray[0, 6] = 1;
         Xarray[1, 7] = 1;
```

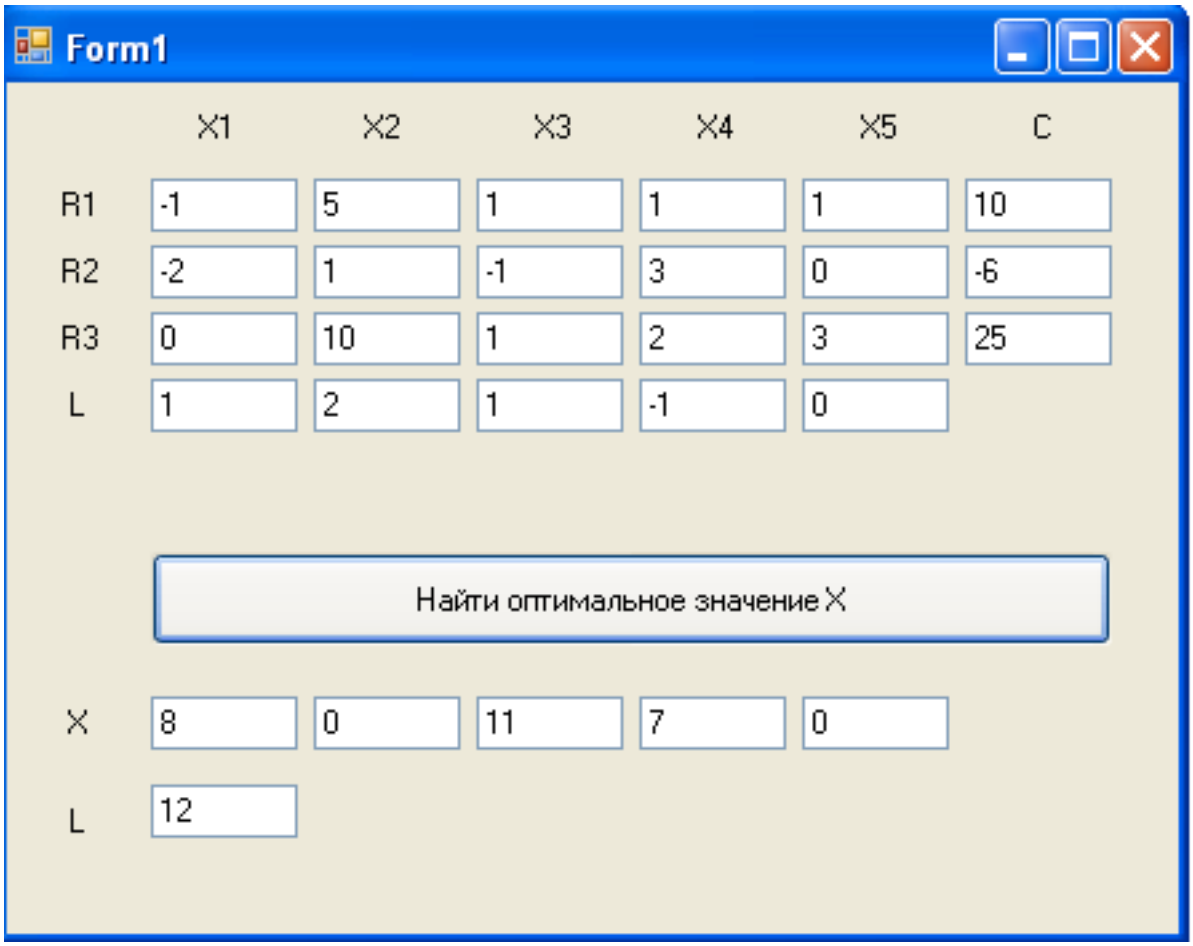
```
Xarray[2, 8] = 1;

//коэф ограничений
//Свободные члены системы ограничений должны быть неотрицательными

if (Convert.ToDouble(C1.Text) < 0)
{
   Xarray[0, 0] = -1 * Convert.ToInt32(X11.Text);
   Xarray[0, 1] = -1 * Convert.ToInt32(X12.Text);
   Xarray[0, 2] = -1 * Convert.ToInt32(X13.Text);
   Xarray[0, 3] = -1 * Convert.ToInt32(X14.Text);
   Xarray[0, 4] = -1 * Convert.ToInt32(X15.Text);
   Xarray[0, 5] = -1 * Convert.ToInt32(C1.Text);
}
else
{
   Xarray[0, 0] = Convert.ToInt32(X11.Text);
   Xarray[0, 1] = Convert.ToInt32(X12.Text);
   Xarray[0, 2] = Convert.ToInt32(X13.Text);
   Xarray[0, 3] = Convert.ToInt32(X14.Text);
   Xarray[0, 4] = Convert.ToInt32(X15.Text);
   Xarray[0, 5] = Convert.ToInt32(C1.Text);
}

if (Convert.ToDouble(C2.Text) < 0)
{
   Xarray[1, 0] = -1 * Convert.ToInt32(X21.Text);
   Xarray[1, 1] = -1 * Convert.ToInt32(X22.Text);
   Xarray[1, 2] = -1 * Convert.ToInt32(X23.Text);
   Xarray[1, 3] = -1 * Convert.ToInt32(X24.Text);
   Xarray[1, 4] = -1 * Convert.ToInt32(X25.Text);
   Xarray[1, 5] = -1 * Convert.ToInt32(C2.Text);
}
else
{
   Xarray[1, 0] = Convert.ToInt32(X21.Text);
   Xarray[1, 1] = Convert.ToInt32(X22.Text);
   Xarray[1, 2] = Convert.ToInt32(X23.Text);
   Xarray[1, 3] = Convert.ToInt32(X24.Text);
   Xarray[1, 4] = Convert.ToInt32(X25.Text);
   Xarray[1, 5] = Convert.ToInt32(C2.Text);
}

if (Convert.ToDouble(C3.Text) < 0)
{
   Xarray[2, 0] = -1 * Convert.ToInt32(X31.Text);
   Xarray[2, 1] = -1 * Convert.ToInt32(X32.Text);
   Xarray[2, 2] = -1 * Convert.ToInt32(X33.Text);
   Xarray[2, 3] = -1 * Convert.ToInt32(X34.Text);
   Xarray[2, 4] = -1 * Convert.ToInt32(X35.Text);
   Xarray[2, 5] = -1 * Convert.ToInt32(C3.Text);
}
else
{
   Xarray[2, 0] = Convert.ToInt32(X31.Text);
   Xarray[2, 1] = Convert.ToInt32(X32.Text);
   Xarray[2, 2] = Convert.ToInt32(X33.Text);
   Xarray[2, 3] = Convert.ToInt32(X34.Text);
   Xarray[2, 4] = Convert.ToInt32(X35.Text);
   Xarray[2, 5] = Convert.ToInt32(C3.Text);
}
```

```
//Введем в рассмотрение вспомогательную функцию W

for (int i = 0; i < 5; i++)
{
   Warray[i] = Xarray[0, i] + Xarray[1, i] + Xarray[2, i];
}

Warray[5] = -Xarray[0, 5] - Xarray[1, 5] - Xarray[2, 5];

//коэффициенты при переменных функции W записываются с противоположными знаками
//а свободный член со своим знаком
for (int i = 0; i < 5; i++)
{
   Warray[i] = -1 * Warray[i];
}

//Вычисление опроного решения функции L
int count =0;

do
{
   //Поиск ведущего столбца
   float min = Warray[0];
   int minNumber = 0;
   for (int i = 0; i < 5; i++)
   {
      if (min > Warray[i])
      {
         min = Warray[i];
         minNumber = i;
      }
   }

   //Поиск ведущей строки
   float minOtnoshenie = 9999;
   int minStroka = 999;
   for (int i = 0; i < 3; i++)
   {
      if (Xarray[i, minNumber] > 0)
      {
         if (minOtnoshenie > (Xarray[i, 5] / Xarray[i, minNumber]))
         {
            minOtnoshenie = (Xarray[i, 5] / Xarray[i, minNumber]);
            minStroka = i;
         }
      }
   }

   //Формирование нового базиса
   NewBazis[minStroka, minNumber] = minNumber+1;

   //Вычисление значений ведущей строки
   float cache = Xarray[minStroka, minNumber];
   for (int i = 0; i < 9; i++)
   {
      Xarray[minStroka, i] = Xarray[minStroka, i] / cache;
   }

   //Вычисление новой таблицы с учетом ведущей стороки и столбца
   // fail 0
   for (int i = 0; i < 3; i++)
   {
```

```
            float cache2 = Xarray[i, minNumber];
            float cache3 = Larray[minNumber];
            float cache4 = Warray[minNumber];
            if (i != minStroka)
            {
                for (int j = 0; j < 9; j++)
                {
                    if (!NoNeed.Contains(j))
                    {
                    Xarray[i, j] = Xarray[i, j] - (Xarray[minStroka, j] * cache2);
                    Larray[j] = Larray[j] - (Xarray[minStroka, j] * cache3);
                    Warray[j] = Warray[j] - (Xarray[minStroka, j] * cache4);

                    }
                }
            }

        }

        NoNeed.Add(6 + minStroka);
        count++;
    } while (count<3);


    bool check=true;
    do
    {
        //Поиск ведущего столбца
        float min2 = Larray[0];
        int minNumber2 = 0;
        for (int i = 0; i < 5; i++)
        {
            if (min2 > Larray[i])
            {
                min2 = Larray[i];
                minNumber2 = i;
            }
        }

        //Поиск ведущей строки
        float minOtnoshenie2 = 9999;
        int minStroka2 = 999;
        for (int i = 0; i < 3; i++)
        {
            if (Xarray[i, minNumber2] > 0)
            {
                if (minOtnoshenie2 > (Xarray[i, 5] / Xarray[i, minNumber2]))
                {
                    minOtnoshenie2 = (Xarray[i, 5] / Xarray[i, minNumber2]);
                    minStroka2 = i;
                }
            }
        }

        //Вычисление значений ведущей строки
        float cache11 = Xarray[minStroka2, minNumber2];
        for (int i = 0; i < 6; i++)
        {
            Xarray[minStroka2, i] = Xarray[minStroka2, i] / cache11;
        }

        //Изменение базиса
```

```
    for (int i = 0; i < 3; i++)
    {
        if (i==minStroka2)
        {
            for (int j = 0; j < 5; j++)
            {
                NewBazis[i, j] = 0;
            }
            NewBazis[i, minNumber2] = minNumber2 + 1;
        }
    }

    //Вычисление новой таблицы с учетом ведущей стороки и столбца
    // fail 0
    for (int i = 0; i < 3; i++)
    {
        float cache2 = Xarray[i, minNumber2];
        float cache3 = Larray[minNumber2];

        if (i != minStroka2)
        {
            for (int j = 0; j < 6; j++)
            {

                Xarray[i, j] = Xarray[i, j] - (Xarray[minStroka2, j] * cache2);
                Larray[j] = Larray[j] - (Xarray[minStroka2, j] * cache3);

            }
        }

    }
    check=true;
    for (int i = 0; i < 3; i++)
        for (int j = 0; j < 6; j++)
            if (Xarray[i, j] < 0)
                check = false;
} while (!check);

//Вывод оптимальго решения
for (int i = 0; i < 3; i++)
{
    for (int j = 0; j < 5; j++)
    {
        if (NewBazis[i,j]!=0)
        {
        int caseswitch = NewBazis[i, j];
        switch (caseswitch)
        {
            case 1:
                X1.Text = Xarray[i, 5].ToString();
                break;
            case 2:
                X2.Text = Xarray[i, 5].ToString();
                break;
            case 3:
                X3.Text = Xarray[i, 5].ToString();
                break;
            case 4:
                X4.Text = Xarray[i, 5].ToString();
                break;
            case 5:
                X5.Text = Xarray[i, 5].ToString();
```

```
                    break;
                default:
                    MessageBox.Show("Ошибка");
                    break;
            }
        }
    }
}
if (X1.Text.Length == 0)
    X1.Text = "0";
if (X2.Text.Length == 0)
    X2.Text = "0";
if (X3.Text.Length == 0)
    X3.Text = "0";
if (X4.Text.Length == 0)
    X4.Text = "0";
if (X5.Text.Length == 0)
    X5.Text = "0";


L.Text = Larray[5].ToString(); //Вывод значения функции для данного решения


    }
  }
}
```



Задачи для самостоятельного решения

Задача 1

Фабрика производит столы и шкафы, которые входят в комплект офисного оборудования. Виды ресурсов, нормы их затрат на одно изделия каждого вида, запасы, а также прибыль от реализации одного изделия приведены в следующей таблице:

Ресурсы	Нормы затрат на одно изделие		Запасы
	Стол	Шкаф	ресурсов
Древесина(м3):			
Дуб	0,2	0,1	40
Орех	0,1	0,3	60
Трудоемкость (чел-часов)	1,2	1,5	371,4
Прибыль от одного изделия	$60	$80	

Определить, сколько столов и шкафов фабрике следует выпускать, чтобы получить максимальную прибыль от реализации.

Задача 2

Фирма имеет возможность рекламировать свою продукцию, используя местные радио- и телевизионную сети. Затраты на рекламу в бюджете фирмы ограничены величиной *1000* долларов в месяц. Каждая минута радиорекламы обходится в *5* долларов, а каждая минута телерекламы – в *90* долларов. Маркетинговая служба фирмы рекомендует, чтобы количество рекламных минут по радио было, по крайней мере, в *2* раза больше, чем по TV. Опыт прошлых лет показал, что объем сбыта, который обеспечивает каждая минута телерекламы, в *25* раз больше сбыта, обеспечиваемого одной минутой радиорекламы. Определить оптимальное распределение финансовых средств, ежемесячно отпускаемых на рекламу, между радио- и телерекламой.

Задача 3

Финансовая компания покупает акции двух акционерных обществ А и В, для чего решено затратить до *$930*. Акции *A* стоят *$27, B - $23*.

Необходимо закупить максимально возможное суммарное число акций. При этом число акций *B* не должно превышать число акций *A* более, чем на *10.* Сколько акций обоих обществ следует закупить при указанных условиях?

Задача 4

В цехе площадью *72 м²* необходимо установить станки новейшего поколения, на приобретение которых отпущено *8,4* млн. рублей. Существует два типа станков. Станок первого типа стоимостью *1,2* млн. рублей, требующий *12 м²* производственных площадей, обеспечивает изготовление *70* изделий в смену. Аналогичные характеристики станка второго типа составляют соответственно *800* тыс. рублей, *6 м²* и *40* изделий в смену. Найти оптимальный вариант приобретения станков, обеспечивающий максимальное производство изделий в смену.

Задача 5

Для продажи меха на престижных аукционах на ферме выращиваются голубая норка и песец. Чтобы обеспечить нормальные условия их выращивания, используют три вида кормов. Количество корма каждого вида, которое должно получать каждое животное, запасы корма каждого вида и прибыль от реализации одной шкурки животного приведены в таблице:

Вид корма	Количество корма, которое ежедневно получает (кг)		Запасы кормов (кг)
	норка	песец	
1	2	3	180
2	4	1	240
3	6	7	426
Прибыль от реализации одной шкурки (руб)	320	240	

Определить, сколько норок и песцов следует выращивать, чтобы прибыль от реализации их шкурок была максимальной.

Задача 6

Сухогруз может принять на борт не более *1020* тонн груза, объем которого не должен превосходить *500 м*^{3}. На причале находится груз двух наименований. Вес, объем и цена единицы груза каждого наименования приведены в таблице:

Наименование груза	Вес 1 единицы груза (т)	Объем 1 единицы груза (м3)	Цена 1 единицы груза (тыс.долл)
В1	50	45	1,5
В2	70	25	1,2

Сформулировать и решить задачу выбора варианта загрузки судна наиболее ценным грузом.

Задача 7

Цех комбината по производству пивобезалкогольной продукции наладил выпуск двух наиболее популярных напитков для детей: ”Ситро” и ”Фанта”. Для производства одного литра ”Ситро” требуется *0,02* часа работы оборудования, а для ”Фанты” – *0,04* часа. Расход специального ингредиента на них составляет *0,01* кг. и *0,04* кг. на один литр соответственно. Ежедневно в распоряжении цеха *16* кг. специального ингредиента и *24* часа работы оборудования. Доход от продажи одного литра ”Ситро” составляет *0,25* рубля, а ”Фанты” – *0,51* рубля. Определить ежедневный план производства напитков каждого вида, обеспечивающий максимальный доход от их продажи.

Задача 8

Филиал завода изготавливает корпуса для холодильников наиболее популярных марок “ Минск ” и ” Стинол ”, комплектуя затем их оборудованием, поставляемым другими предприятиями. В таблице указаны

нормы трудозатрат, затрат материалов для изготовления корпусов, месячные объемы ресурсов и прибыль от реализации холодильника каждой марки:

Наименование ресурса	Затраты на один холодильник марки A1	Затраты на один холодильник марки A2	Объемы ресурсов
Трудозатраты (чел-час)	2	3	900
Металл (м2)	2	2	850
Пластик (м2)	1	3	400
Краска (кг)	2	2	500
Прибыль от одного холодильника (долл)	40	100	

Найти месячный план выпуска холодильников, максимизирующий прибыль.

Задача 9

Требуется загрузить теплоход апельсинами и лимонами так, чтобы стоимость груза была максимальной. При этом известно, что объем грузового отсека теплохода составляет *1000 м*^{3}; картонные коробки, в которых перевозятся апельсины и лимоны, имеют стандартные размеры: *50×50×40 см*^{3}; стоимость одной коробки с апельсинами равна *$45*; с лимонами - *$60*. Потребитель данной продукции заинтересован, чтобы апельсинов по сравнению с лимонами было доставлено больше, по крайней мере, в *1,5* раза.

Задача 10

Компания производит книжные полки двух видов: *A* и *B*. Агенты по продаже считают, что в неделю на рынке может быть реализовано до *550* полок. Для каждой полки типа *A* требуется *2 м*^{2} материалов, а для полки типа *B – 5 м*^{2} материалов. Компания может получить до *1199 м*^{2} материала в неделю. Для изготовления одной полки типа *A* требуется *12* мин. машинного времени, а для изготовления одной полки типа *B – 30* мин. Машинное оборудование можно использовать не более *160* часов в неделю. Если прибыль от продажи одной полки типа *A* составляет *$3*, а от продажи одной

полки типа *B -$7*, то сколько полок каждого типа следует выпускать в неделю для получения максимальной прибыли?

Задача 11

Изделия 4-х типов проходят последовательную обработку на 2-х станках. Время обработки одного изделия каждого типа на каждом из станков приведено в таблице:

Станок	Время обработки одного изделия (час)			
	тип 1	тип 2	тип 3	тип 4
1	2	3	4	2
2	3	2	1	2

Затраты на производство одного изделия каждого типа определяются как величины, прямопропорциональные времени использования станков (в машино-часах). Стоимость машино-часа составляет *10* долларов для станка *1* и *15* долларов для станка *2*. Допустимое время использование станков для обработки изделий всех типов ограничено следующими значениями: *500* машино-часов для станка *1* и *380* - машино-часов для станка *2*. Цены изделий типов *1, 2, 3* и *4* равны соответственно *65, 70, 55* и *55* долларов соответственно.

Составить и решить задачу линейного программирования, считая целью максимизацию суммарной чистой прибыли.

Задача 12

При откорме животных каждое из них должно ежедневно получать питательного вещества a в диапазоне от pa до va, вещества b - от pb до vb и вещества c - от pc до vc. Указанные питательные вещества содержатся в трех видах кормов, цены которых составляют r1, r2 и r3 рублей за 1 кг.

Вариант	pa	pb	pc	va	vb	vc	r1	r2	r3
1	60	50	12	70	60	19	0.9	1.2	1
2	45	30	28	55	45	38	1.1	0.8	0.7
3	90	51	35	100	59	45	1.6	2	2.3

Содержание питательных веществ в 1 кг каждого из видов кормов задано в таблице. Составить рацион кормления, обеспечивающий получение необходимого количества питательных веществ при минимальных денежных затратах.

Вид корма	Содержание питательных веществ в кормах (ед/кг)								
	Вариант 1			Вариант 2			Вариант 3		
	a	b	c	a	b	c	a	b	c
1	1	3	4	2	3	2	12	16	23
2	2	4	2	3	4	5	15	8	15
3	1	4	3	7	5	1	17	19	5

Задача 13

Кондитерская фабрика для производства трех видов карамели a, b и c использует три вида сырья: сахарный песок, патоку и фруктовое пюре. Нормы расхода сырья на производство 1 кг. карамели заданы в таблице.

Наименование сырья	Нормы расхода (кг./кг.)		
	a	b	c
Сахарный песок	0.6	0.5	0.6
Патока	0.4	0.4	0.3
Фруктовое пюре	0.1	0.2	0.2

Запасы сырья на складе соответственно равны v1, v2 и v3 кг. Прибыль от реализации 1 кг. продукции каждого вида определяется значениями pa, pb и pc. Найти план производства карамели, обеспечивающий максимальную прибыль. Выяснить, какое сырье ограничивает рост прибыли.

Вариант	Запасы сырья (кг)			Прибыль от реализации (руб/кг)		
	v1	v2	v3	pa	pb	pc
1	800	600	120	1.08	1.12	1.28
2	400	400	250	1.20	1.34	1.40
3	300	400	100	1.00	1.10	1.18

Задача 14

Мебельная фабрика выпускает столы, стулья, кресла и кровати. При изготовлении этих товаров используется два вида досок. Запасы досок и трудовых ресурсов показаны в таблице.

Вариант	Ресурсы

	Доски 1 типа (м)	Доски 2 типа (м)	Трудовые
1	1500	1000	800
2	1500	1000	1500
3	2000	3000	2300

Нормы расхода сырья, трудоемкость производства продукции и прибыль от реализации единицы продукции показаны в таблице.

Изделия	Нормы расхода			
Ресурсы	Столы	Стулья	Кресла	Кровати
Доски 1 типа (м)	5	1	9	12
Доски 2 типа (м)	2	3	4	6
Трудоемкость (чел.-час.)	3	2	5	10
Прибыль (руб.)	12	5	15	18

С учетом спроса на товары фабрика должна выпустить не более 10 кроватей, а соотношение столов и стульев должно быть 1:6. Найти план производства мебели, дающий фабрике максимальную прибыль.

Задача 15

Нефтеперерабатывающий завод имеет запасы 4-х полуфабрикатов: алкилата (s1); крекинг-бензина (s2); бензина прямой перегонки (s3) и изопентона (s4). В результате смешивания этих компонентов в различных пропорциях, получают 3 сорта авиационного бензина: a, b и c. Запасы сырья, состав бензина и себестоимость его производства даны в таблице.

Вариант	Запасы сырья (л)				Состав бензина s1:s2:s3:s4		
	s1	s2	s3	s4	a	b	c
1	400000	250000	350000	100000	2:3:5:2	3:1:2:1	2:2:1:3
2	200000	300000 300000		150000			
3	180000	200000	100000	300000			
Прибыль от 1 тыс. л бензина (руб)					1200	1000	1500

Определить план производства бензина различных сортов, обеспечивающий максимальную прибыль производства.

Задача 16

Рацион стада крупного рогатого скота из 220 голов включает пищевые продукты A, B, C, d и E. В сутки одно животное должно съедать не менее 2 кг продукта вида a, 1,5 кг продукта B, 0,9 кг продукта C, 3 кг продукта d и 1,8 кг продукта e. Однако в чистом виде указанные продукты не производятся. Они содержатся в концентратах К-1, К-2 и К-3. Их цена соответственно 0,5; 0,4; 0,9 руб. за килограмм. Содержание продуктов в килограмме концентрата (в %) указано в таблице.

Концентраты	Продукты				
	a	b	c	d	e
К-1	15	22	0	0	4
К-2	19	17	0	14	7
К-3	5	12	25	5	8

Построить модель, на основе которой составить план покупки концентратов, при котором затраты на покупку будут минимальны.

Задача 17

В плановом году строительные организации города переходят к сооружению домов типов Д-1, Д-2, Д-3 и Д-4. Данные о количестве квартир разного типа в каждом из указанных типов домов, их плановая себестоимость приведены в таблице. Годовой план ввода жилой площади составляет соответственно 800, 1000, 900, 2000 и 7000 квартир указанных типов.

Показатели	Д-1	Д-2	Д-3	Д-4
Типы квартир однокомнатные	10	18	20	15
двухкомнатные: смежные	40		20	
несмежные		20		60
трехкомнатные.	60	90	10	
четырехкомнатные	20	10		5
Плановая себестоимость, тыс. руб.	830	835	360	450

На жилищное строительство утвержден объем капиталовложений в размере 40 млн. руб. (часть этих средств, которая не будет использована в

плановом году по прямому назначению, предназначена для расширения сети коммунальных предприятий города).Построить модель и найти план строительства домов на финансовый год, при котором себестоимость всех вводимых домов будет минимальной.

Задача 18

Авиакомпания МОГОЛ по заказу армии должна перевезти на некотором участке 700 человек. В распоряжении компании имеется два типа самолетов, которые можно использовать для перевозки. Самолет первого типа перевозит 30 пассажиров и имеет экипаж 3 человека, второго типа - 65 и 5 соответственно. Эксплуатация 1 самолета первого типа обойдется 5000$, а второго 9000$. Сколько надо использовать самолетов каждого типа, если для формирования экипажей имеется не более 60 человек.

Задача 19

В пекарне для выпечки четырех видов хлеба используется мука двух сортов, маргарин и яйца. Имеющееся оборудование, производственные площади и поставки продуктов таковы, что в сутки можно переработать не более 290 кг муки первого сорта, 150 кг муки второго сорта, 50 кг маргарина, 1280 шт. яиц. В таблице приведены нормы расхода продуктов, а также прибыль от продажи 1 кг хлеба каждого вида.

Наименование продукта	Нормы расхода на 1 кг хлеба (по видам)			
	1	2	3	4
Мука 1 сорта, кг	0,5	0,5	0	0
Мука 2 сорта, кг	0	0	0,5	0,5
Маргарин, кг	0,125	0	0	0,125
Яйцо, шт.	2	1	1	1
Прибыль, за 1 кг	14	12	5	6

Требуется определить суточный план выпечки хлеба, максимизирующий прибыль.

Задача 20

На предприятии в производстве используется 3 технологических способа - I, II, III. При этом трудовые ресурсы использованы полностью, а накладные расходы должны быть не меньше запаса ресурса.

Виды ресурсов	Технологические способы			Запасы ресурсов
Сырье	3	2	1	26
Трудовые ресурсы	1	1	2	11
Накладные расходы	7	9	5	32
Расход воды	2	4	1	

Лабораторная работа
«Двойственный симплекс – метод. Матричное представление симплексных решений»

Цель работы: изучение принципов составления математической модели для задач линейного программирования, получение навыков работы решения задач, а также применение данного метода на практике.

Порядок выполнения:

1. Найти оптимальное значение функции при заданных условиях для своего варианта;
2. Разработать программу решающую поставленную задачу в общем виде;
3. Составить отчет по работе.

Теоретические сведения

Оптимальное решение задачи линейного программирования определяется теми условиями, которые нашли отражение в модели в момент ее формирования, В реальной жизни условия, формирующие модель, не остаются неизменными. В связи с этим особое значение приобретают средства, позволяющие оценить изменения в оптимальном решении, вызванные изменениями в параметрах исходной модели. Таким средством является анализ чувствительности. Он предлагает эффективные вычислительные методы, позволяющие изучить динамическое поведение оптимального решения.

Исходную задачу линейного программирования называется прямой. Двойственная задача — это задача, формулируемая с помощью определенных правил непосредственно из прямой задачи.

Напомним, что задача ЛП в стандартной форме записывается следующим образом.

Максимизировать или минимизировать целевую функцию

$z = \sum_{j=0}^{n} c_j x_j$

при ограничениях

$\sum_{□=1}^{n} a_{ij} x_j = b_j, i = 1,2, \dots, m$

$x_j \geq 0, j = 1,2, \dots, n$

В состав n переменных x_j входят также дополнительные переменные. Стандартная форма задачи линейного программирования предполагает выполнение следующих условий.

1. Все ограничения записаны в виде равенств (с неотрицательной правой частью);
2. Все переменные неотрицательны;
3. Оптимизация определяется как максимизация или минимизация целевой функции.

Стандартная форма задачи линейного программирования порождает стандартную таблицу симплекс-метода. Таким образом, определив двойственную задачу на основе стандартной формы прямой задачи, после вычислений симплекс-метода мы автоматически получаем решение двойственной задачи.

Переменные и ограничения двойственной задачи формируются путем симметричных структурных преобразований прямой задачи по следующим правилам:

1. Каждому из m ограничений прямой задачи соответствует переменная двойственной задачи
2. Каждой из n переменных прямой задачи соответствует ограничение двойственной задачи.
3. Коэффициенты при какой-либо переменной в ограничениях прямой задачи становятся коэффициентами ограничения двойственной задачи, соответствующей этой переменной, а правая часть формируемого ограничения равна коэффициенту при этой переменной в выражении целевой функции.
4. Коэффициенты целевой функции двойственной задачи равны правым частям ограничений прямой задачи.

В двойственном симплекс-методе решение задачи линейного программирования начинается с недопустимого, но лучшего, чем оптимальное решение. Последовательные итерации этого метода приближают решение к области допустимости без нарушения оптимальности промежуточных решений. Когда будет достигнута область допустимых решений, процесс вычислений заканчивается, так как последнее решение будет оптимальным. Уже из этого краткого описания двойственного симплекс-метода видно, что он значительно отличается от обычного (прямого) симплекс-метода, где начальное решение всегда допустимое, но не оптимальное, и промежуточные решения никогда не выходят из пространства допустимых решений.

В двойственном симплекс-методе начальная симплекс-таблица обязательно должна иметь в базисном решении недопустимую (т.е. отрицательную) переменную. Для реализации двойственного симплекс-метода разработаны следующие два условия, выполнение которых гарантирует оптимальность последовательных промежуточных решений и приближение их к области допустимых решений.

Двойственное условие допустимости. В качестве исключаемой переменной x_r выбирается базисная переменная, имеющая наибольшее по абсолютной величине отрицательное значение. Если таких переменных несколько, то выбор произволен. Если все базисные переменные неотрицательные, процесс вычислений заканчивается.

Двойственное условие оптимальности. Вводимая в базис переменная определяется как переменная, на которой достигается следующий минимум:

$$\min_{\text{небазисные } j} \left\{ \left| \frac{z_j - c_j}{\alpha_{rj}} \right|, \alpha_{rj} < 0 \right\}$$

где α_{rj} — коэффициент из симплекс-таблицы, расположенный на пересечении ведущей строки (соответствующей исключаемой переменной x_r) и столбца, соответствующего переменной x_j. При наличии нескольких альтернативных переменных, выбор делается произвольно.

После некоторых раздумий вы должны заметить, что двойственное условие оптимальности основывается на той же основной идее, что и условие оптимальности прямого симплекс-метода.

Пример построения математической модели

Дана следующая задача линейного программирования.

Минимизировать $z = 3x_1 + 2x_2$ при ограничениях

$3x_1 + x_2 \geq 3$

$4x_1 + 3x_2 \geq 6$

$x_1 + x_2 \leq 3$

$x_1, x_2 \geq 0$

Начальная симплекс-таблица этой задачи имеет следующий вид.

Базис	x_1	x_2	x_3	x_4	x_5	Решение
z	-3	-2	0	0	0	0
x_3	-3	-1	1	0	0	-3
x_4	-4	-3	0	1	0	-6
x_5	1	1	0	0	1	3

Среди дополнительных переменных этой задачи, x_3 и x_4 являются избыточными, а x_5 — остаточной. Мы умножили каждое равенство, соответствующее избыточным дополнительным переменным, на -1; в результате правые части этих равенств непосредственно указывают на базисные переменные, которые являются недопустимыми ($x_3 = -3$, $x_4 = -6$, $x_5 = 3$). Этот подход всегда используется при реализации двойственного симплекс-метода. Поскольку $z_j - c_j < 0$ для всех у = 1,..., 5, начальное базисное решение является оптимальным (но не допустимым). Таким образом, приведенная таблица удовлетворяет требованиям начальной таблицы двойственного симплекс-метода, а именно — оптимальности и недопустимости.

Двойственное условие допустимости указывает на переменную x_4 (=-6) как на вводимую в базис. Теперь применим двойственное условие оптимальности для определения исключаемой переменной. Для этого используем следующую таблицу.

Переменные	x_1	x_2	x_3	x_4	x_5
z-строка (z_j-c_j)	-3	-2	0	0	0
x_4-строка (α_{4j})	-4	-3	0	1	0
Отношение $\left\lvert\frac{\square_j - c_j}{a_{4j}}\right\rvert$	$\frac{3}{4}$	$\frac{2}{3}$	-	-	-

Приведенные отношения показывают, что исключаемой переменной будет x_2. Отметим, что переменные x_j будут кандидатами на исключение из базисного решения только тогда, когда коэффициент α_{4j} будет строго отрицательным. По этому критерию переменные x_3, x_4 и x_5 не рассматриваются как кандидаты на исключение из базиса.

Следующая таблица получена с помощью известных операций над строками, применяемых в прямом симплекс-методе.

Базис	x_1	x_2	x_3	x_4	x_5	Решение
z	-1/3	0	0	-2/3	0	4
x_3	-5/3	0	1	-1/3	0	-1
x_2	4/3	1	0	-1/3	0	2
x_5	-1/3	0	0	1/3	1	1
Отношение	1/5	-	-	2	-	

Последняя таблица показывает, что из базиса исключается переменная x_3 и вводится x_1. В результате получаем следующую симплекс-таблицу.

Базис	x_1	x_2	x_3	x_4	x_5	Решение
z	0	0	-1/5	-3/5	0	21/5
x_1	1	0	-3/5	1/5	0	3/5
x_2	0	1	4/5	-3/5	0	6/5
x_5	0	0	-1/5	2/5	1	6/5

Решение, представленное в последней таблице, допустимо (и оптимально), поэтому вычисления заканчиваются. Это решение имеет вид $x_1=3/5$, $x_2=6/5$ и $x_3=21/5$.

Задачи для самостоятельного решения

Выполнить программное решение задачи согласно варианту двойственным симплекс методом.

Задача 1

$$\mathbf{x_1+5x_2+x_3+10x_4+x_5+3x_6\rightarrow min}$$

$$\mathbf{-x_1+x_2+x_3+x_4\geq 1}$$
$$\mathbf{x_1+2x_2-x_3+3x_4- x_5-x_6\geq 1}$$
$$\mathbf{x_i\geq 0,\ i=1..6}$$

Задача 2

$$\mathbf{3x_1+2x_2+3x_3+4x_4-x_5\rightarrow min}$$

$$\mathbf{x_1+x_2-2x_5\geq 1}$$
$$\mathbf{x_1+x_3\geq 1}$$
$$\mathbf{x_1+x_4\geq 1}$$
$$\mathbf{x_i\geq 0,\ i=1..5}$$

Задача 3

$$\mathbf{x_1+x_2+x_3+x_4+2x_5+10x_6+15x_7\rightarrow min}$$

$$\mathbf{x_1+x_3+x_4-x_5-3x_6+2x_7\geq 1}$$
$$\mathbf{x_1+x_2+x_3+x_5-x_6+x_7\geq 0}$$
$$\mathbf{x_1+x_2+x_3+x_5-x_6\geq 0}$$
$$\mathbf{x_i\geq 0,\ i=1..7}$$

Задача 4

$3x_1+5x_2+4x_3 \rightarrow min$

$x_1+2x_2+3x_3 \geq 1$
$2x_1+3x_2+x_3 \geq 1$
$3x_1+x_2+2x_3 \geq 1$
$6x_1+6x_2+6x_3 \geq 1$
$x_i \geq 0$, i=1..3

Задача 5
$x_1-2x_2+3x_3-x_4 \rightarrow min$

$x_1-x_2-x_3+4x_4 \geq 0$
$x_1+3x_3+3x_4 \geq -1$
$-x_1+2x_2+3x_3+4x_4 \geq -2$
$x_1+x_2-2x_3-x_4 \geq 0$
$x_i \geq 0$, i=1..4

Задача 6
$x_1-8x_2+2x_3-10x_4 \rightarrow max$

$7x_1+26x_2-x_3+17x4 \geq 5$
$x_1+7x_2+5x_3-13x_4 \geq 12$
$x_i \geq 0$, i=1..4

Задача 7
$x_1+x_2+3x_3-x_4 \rightarrow max$

$x_1+x_2+3x_3 \geq 1$
$x_1-x_2+2x_4 \geq -1$
$x_i \geq 0$, i=1..4

Задача 8
$x_1+x_2+3x_3-2x_4 \rightarrow max$

$x_1+2x_2+5x_3-x_4 \geq 3$
$3x_1-x_2+x_3-10x_4 \geq 2$
$x_i \geq 0$, i=1..4

Задача 9

$x_1+8x_2+10x_3 \rightarrow max$

$x_1+x_2+4x_3 \geq 2$
$x_1-x_2+2x_3 \geq 0$
$x_i \geq 0, i=1..3$

Задача 10
$x_1-x_2+x_3+2x_4 \rightarrow max$

$x_1+x_2+x_3+x_4 \leq 1$
$x_1+2x_2-x_3+x_4 \leq 1$
$x_i \geq 0, i=1..4$

Задача 11
$2x_1+3x_2-7x_3+14x_4 \rightarrow min$

$x_1+x_2+x_3+x_4 \leq 2$
$x_1+2x_2-4x_3+7x_4 \leq -2$
$x_i \geq 0, i=1..4$

Задача 12
$2x_1-8x_2+2x_4+15x_5-10x_6 \rightarrow max$

$x_1+x_3+x_4+x_5+3x_6 \leq 4$
$x_1-4x_2+x_4+10x_5-x_6 \leq 5$
$x_1+3x_2+7x_3-x_4-15x_5-7x_6 \leq 2$
$x_i \geq 0, i=1..6$

Задача 13
$x_1-x_2+x_3-x_4+x_5 \rightarrow max$

$x_1+x_2+4x_3+4x_4+3x_5 \leq 3$
$x_1-x_2-2x_3+2x_4+5x_5 \leq 1$
$x_i \geq 0, i=1..5$

Задача 14
$x_1+3x_2+2x_3-x_4 \rightarrow min$

$x_1+4x_2+4x_3 \geq 1$

$x_1-x_2+2x_4 \geq -1$
$x_i \geq 0$, i=1..4

Задача 15
$x_1+x_2+3x_3+2x_4+x_5+4x_6+5x_7 \rightarrow min$

$x_1+x_3+3x_4+2x_5-x_6+2x_7 \geq 1$
$x_1-x_2-x_3+x_5-x_6+x_7 \geq 0$
$x_1+x_2-x_5+x_6 \geq 0$
$x_i \geq 0$, i=1..7

Задача 16
$x_1-4x_2+2x_4+3x_5-5x_6 \rightarrow max$

$x_1+x_3+2x_4+x_5+x_6 \leq 2$
$x_1+4x_2-8x_4+4x_5-x_6 \leq 6$
$x_1+2x_2+2x_3-5x_4-x_5-x_6 \leq 4$
$x_i \geq 0$, i=1..6

Задача 17
$x_1-4x_2+6x_3+3x_4 \rightarrow max$

$3x_1+2x_2+x_3+x_4 \geq 6$
$x_1+7x_2-5x_3-3x_4 \geq 4$
$x_i \geq 0$, i=1..4

Задача 18
$x_1+4x_2+3x_3+x_4+x_5+3x_6 \rightarrow min$

$-x_1+x_2-x_3+x_4 \geq 3$
$x_1+x_2-2x_3+5x_4-3x_5-x_6 \geq 2$
$x_i \geq 0$, i=1..6

Задача 19
$4x_1+3x_2+x_3-4x_4 \rightarrow min$

$x_1+x_2-x_3+x_4 \leq 1$
$x_1-x_2-3x_3+7x_4 \leq -2$
$x_i \geq 0$, i=1..4

Задача 20

$x_1+4x_2+8x_3 \rightarrow max$

$x_1+2x_2-4x_3 \geq 2$

$x_1+x_2+2x_3 \geq 0$

$x_i \geq 0,\ i=1..3$

Лабораторная работа
«Решение транспортных задач»

Цель работы: изучение принципов решения транспортных задач, применение полученных методов на практике в виде программной реализации задачи согласно варианту.

Порядок выполнения:

1. Изучить теоретические сведения.
2. Построить математическую модель решения задачи согласно варианту.
3. Разработать программу решающую поставленную задачу в общем виде.
4. Составить отчет по работе.

Теоритические сведения

Транспортные модели (задачи) — специальный класс задач линейного программирования. Эти модели часто описывают перемещение (перевозку) какого-либо товара из пункта отправления (исходный пункт, например место производства) в пункт назначения (склад, магазин, грузохранилище). Назначение транспортной задачи — определить объем перевозок из пунктов отправления в пункты назначения с минимальной суммарной стоимостью перевозок. При этом должны учитываться ограничения, налагаемые на объемы грузов, имеющихся в пунктах отправления (предложения), и ограничения, учитывающие потребность грузов в пунктах назначения (спрос). В транспортной модели предполагается, что стоимость перевозки по какому-либо маршруту прямо пропорциональна объему груза, перевозимого по этому маршруту. В общем случае транспортную модель можно применять для описания ситуаций, связанных с управлением запасами, управлением движением капиталов, составлением расписаний, назначением персонала и др.

На рисунке 1 показано общее представление транспортной задачи в виде сети с n пунктами отправления и m пунктами назначения, которые показаны в виде узлов сети. Дуги, соединяющие узлы сети, соответствуют маршрутам, связывающим пункты отправления и назначения. С дугой (i, j), соединяющей пункт отправления i с пунктом назначения j, соотносятся два вида данных: стоимость c_{ij} перевозки единицы груза из пункта i в пункт j и количество перевозимого груза x_{ij}. Объем грузов в пункте отправления i

равен a_i, а объем грузов в пункте назначения j — b_j,. Задача состоит в определении неизвестных величин x_{ij}, минимизирующих суммарные транспортные расходы и удовлетворяющих ограничениям, налагаемым на объемы грузов в пунктах отправления (предложения) и пунктах назначения (спрос).

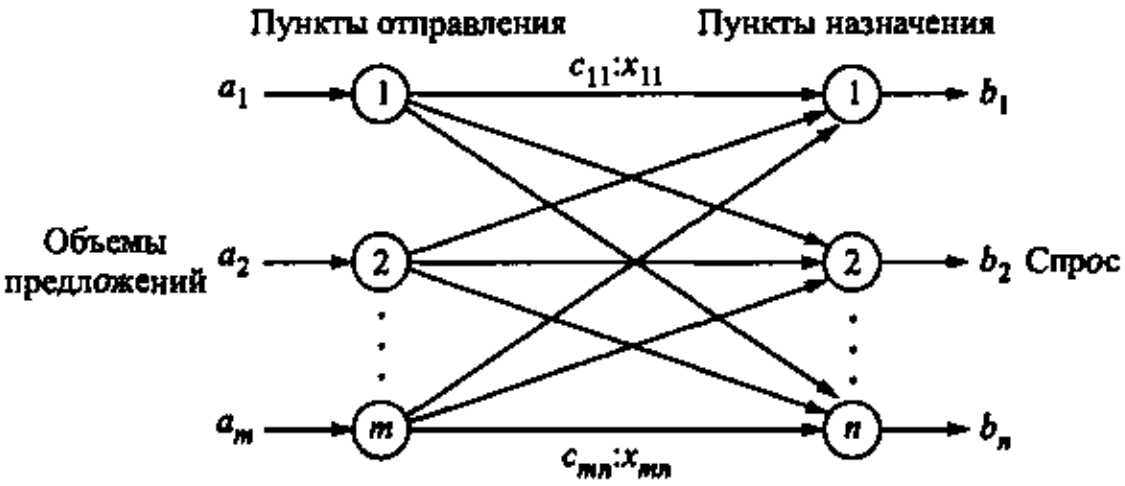


Рисунок 1 – Представление транспортной задачи

Специальная структура транспортной модели для построения начального решения позволяет применить следующие методы :

1. Метод северо-западного угла
2. Метод наименьшей стоимости
3. Метод Фогеля

В общем случае метод Фогеля дает наилучшее решение, а метод северо-западного угла — наихудшее. Однако метод северо-западного угла требует меньшего объема вычислений.

Метод северо-западного угла. Выполнение начинается с верхней левой ячейки (северо-западного угла) транспортной таблицы, т.е. с переменной x_{11}.

Шаг 1. Переменной x_{11} присваивается максимальное значение, допускаемое ограничениями на сирое и предложение.

Шаг 2. Вычеркивается строка (или столбец) с полностью реализованным предложением (с удовлетворенным спросом). Это означает, что в вычеркнутой строке (столбце) мы не будем присваивать значения остальным переменным (кроме переменной, определенной на первом этапе). Если одновременно удовлетворяются спрос и предложение, вычеркивается только строка или только столбец.

Шаг 3. Если не вычеркнута только одна строка или только один столбец, процесс останавливается. В противном случае переходим к ячейке справа, если вычеркнут столбец, или к нижележащей ячейке, если вычеркнута строка. Затем возвращаемся к первому этапу.

Метод наименьшей стоимости. Данный метод находит лучшее начальное решение, чем метод северо-западного угла, поскольку выбирает переменные, которым соответствуют наименьшие стоимости. Сначала по всей транспортной таблице ведется поиск ячейки с наименьшей стоимостью. Затем переменной в этой ячейке присваивается наибольшее значение, допускаемое ограничениями на спрос и предложение. (Если таких переменных несколько, выбор произволен.) Далее вычеркивается соответствующий столбец или строка, и соответствующим образом корректируются значения спроса и предложений. Если одновременно выполняются ограничения и по спросу, и по предложению, вычеркивается или строка, или столбец (точно так же, как в методе северо-западного угла). Затем просматриваются невычеркнутые ячейки, и выбирается новая ячейка с минимальной стоимостью. Описанный процесс продолжается до тех пор, пока не останется лишь одна невычеркнутая строка или столбец.

Метод Фогеля. Данный метод является вариацией метода наименьшей стоимости и в общем случае находит лучшее начальное решение. Алгоритм этого метода состоит из следующих шагов.

Шаг 1. Для каждой строки (столбца), которой соответствует строго положительное предложение (спрос), вычисляется штраф путем вычитания наименьшей стоимости из следующей по величине стоимости в данной строке (столбце).

Шаг 2. Выделяется строка или столбец с наибольшим штрафом. Если таковых несколько, выбор произволен. Из выделенной строки или столбца выбирается переменная, которой соответствует минимальная стоимость, и ей присваивается наибольшее значение, позволяемое ограничениями. Затем в соответствии с присвоенным значением переменной корректируются величины оставшегося неудовлетворенным спроса и нереализованного предложения. Строка или столбец, соответствующие выполненному ограничению, вычеркиваются из таблицы. Если одновременно выполняются ограничения и по спросу, и по предложению, вычеркивается только строка или только столбец, причем оставшейся строке (столбцу) приписывается нулевое предложение (спрос).

Шаг 3.

а) Если не вычеркнута только одна строка или только один столбец с нулевым спросом или предложением, вычисления заканчиваются.

б) Если не вычеркнута только одна строка (столбец) с положительным предложением (спросом), в этой строке (столбце) методом наименьшей стоимости находятся базисные переменные, и вычисления заканчиваются.

в) Если всем невычеркнутым строкам и столбцам соответствуют нулевые объемы предложения и спроса, методом наименьшей стоимости находятся нулевые базисные переменные, и вычисления заканчиваются.

г) Во всех остальных случаях необходимо перейти к шагу 1.

Пример построения математической модели

Рассмотрим в качестве примера метод наименьшей стоимости.

У поставщиков A1, A2, A3, A4, A5, находится соответственно 100, 150, 350, 200, 200 единиц однотипной продукции, которая должна быть доставлена потребителям B1, B2, B3, B4, B5 в количестве 100, 200, 200, 300, 200 единиц соответственно. Стоимость доставки единицы продукции от поставщика A1 к указанным потребителям равна 4, 3, 5, 2, 3 ден.ед. Стоимость доставки единицы продукции от поставщика A2 к указанным потребителям равна 7, 1, 2, 3, 1 ден.ед. Стоимость доставки единицы продукции от поставщика A3 к указанным потребителям равна 9, 2, 4, 5, 6 ден.ед. Стоимость доставки единицы продукции от поставщика A4 к указанным потребителям равна 1, 3, 6, 4, 10 ден.ед. Стоимость доставки единицы продукции от поставщика A5 к указанным потребителям равна 5, 8, 15, 6, 15 ден.ед. Требуется найти оптимальное решение доставки продукции от поставщиков к потребителям, минимизирующие стоимость доставки.

Согласно условию задачи составим таблицу:

Поставщик	Потребитель					Запас
	B1	B2	B3	B4	B5	
A1	4	3	5	2	3	100
A2	7	1	2	3	1	150
A3	9	2	4	5	6	350
A4	1	3	6	4	10	200
A5	5	8	15	6	15	200
Потребность	100	200	200	300	200	

Минимальный элемент матрицы тарифов находится в ячейке A2B2 и равен 1, т.е. из незадействованных маршрутов, маршрут доставки продукции от поставщика A2 к потребителю B2 наиболее рентабельный. Запасы поставщика A2 составляют 150 единиц продукции. Потребность потребителя B2 составляет 200 единиц продукции. От поставщика A2 к потребителю B2 будем доставлять min = { 150 , 200 } = 150 единиц продукции. Разместим в ячейку A2B2 значение равное 150. Мы полностью израсходовали запасы поставщика A2. Исключаем строку 2 таблицы из дальнейшего рассмотрения.

Поставщик	Потребитель					Запас
	B1	B2	B3	B4	B5	
A1	4	3	5	2	3	100
A2	7	150/1	2	3	1	150
A3	9	2	4	5	6	350
A4	1	3	6	4	10	200
A5	5	8	15	6	15	200
Потребность	100	200	200	300	200	

Минимальный элемент матрицы тарифов находится в ячейке A4B1 и равен 1, т.е. из незадействованных маршрутов, маршрут доставки продукции от поставщика A4 к потребителю B1 наиболее рентабельный. Запасы поставщика A4 составляют 200 единиц продукции. Потребность потребителя B1 составляет 100 единиц продукции. От поставщика A4 к потребителю B1 будем доставлять min = { 200 , 100 } = 100 единиц продукции. Разместим в ячейку A4B1 значение равное 100. Мы полностью удовлетворили потребность потребителя B1. Исключаем столбец 1 таблицы из дальнейшего рассмотрения.

Поставщик	Потребитель					Запас
	B1	B2	B3	B4	B5	
A1	4	3	5	2	3	100
A2	7	150/1	2	3	1	150
A3	9	2	4	5	6	350
A4	100/1	3	6	4	10	200
A5	5	8	15	6	15	200
Потребность	100	200	200	300	200	

Минимальный элемент матрицы тарифов находится в ячейке A1B4 и равен 2, т.е. из незадействованных маршрутов, маршрут доставки продукции от поставщика A1 к потребителю B4 наиболее рентабельный. Запасы поставщика A1 составляют 100 единиц продукции. Потребность потребителя B4 составляет 300 единиц продукции. От поставщика A1 к потребителю B4 будем доставлять min = { 100 , 300 } = 100 единиц продукции. Разместим в ячейку A1B4 значение равное 100. Мы полностью израсходовали запасы поставщика A1. Исключаем строку 1 таблицы из дальнейшего рассмотрения.

Поставщик	Потребитель					Запас
	B1	B2	B3	B4	B5	
A1	4	3	5	100/2	3	100
A2	7	150/1	2	3	1	150
A3	9	2	4	5	6	350
A4	100/1	3	6	4	10	200
A5	5	8	15	6	15	200
Потребность	100	200	200	300	200	

Минимальный элемент матрицы тарифов находится в ячейке A3B2 и равен 2, т.е. из незадействованных маршрутов, маршрут доставки продукции от поставщика A3 к потребителю B2 наиболее рентабельный. Запасы поставщика A3 составляют 350 единиц продукции. Потребность потребителя B2 составляет 50 единиц продукции. От поставщика A3 к потребителю B2 будем доставлять min = { 350 , 50 } = 50 единиц продукции. Разместим в ячейку A3B2 значение равное 50. Мы полностью удовлетворили потребность потребителя B2. Исключаем столбец 2 таблицы из дальнейшего рассмотрения.

Поставщик	Потребитель					Запас
	B1	B2	B3	B4	B5	
A1	4	3	5	100/2	3	100
A2	7	150/1	2	3	1	150
A3	9	50/2	4	5	6	350
A4	100/1	3	6	4	10	200
A5	5	8	15	6	15	200
Потребность	100	200	200	300	200	

Минимальный элемент матрицы тарифов находится в ячейке A3B3 и равен 4, т.е. из незадействованных маршрутов, маршрут доставки продукции от поставщика A3 к потребителю B3 наиболее рентабельный. Запасы поставщика A3 составляют 300 единиц продукции. Потребность потребителя B3 составляет 200 единиц продукции. От поставщика A3 к потребителю B3 будем доставлять min = { 300 , 200 } = 200 единиц продукции. Разместим в ячейку A3B3 значение равное 200. Мы полностью удовлетворили потребность потребителя B3. Исключаем столбец 3 таблицы из дальнейшего рассмотрения.

Поставщик	Потребитель					Запас
	B1	B2	B3	B4	B5	
A1	4	3	5	100/2	3	100
A2	7	150/1	2	3	1	150
A3	9	50/2	200/4	5	6	350
A4	100/1	3	6	4	10	200
A5	5	8	15	6	15	200
Потребность	100	200	200	300	200	

Минимальный элемент матрицы тарифов находится в ячейке A4B4 и равен 4, т.е. из незадействованных маршрутов, маршрут доставки продукции от поставщика A4 к потребителю B4 наиболее рентабельный. Запасы поставщика A4 составляют 100 единиц продукции. Потребность потребителя B4 составляет 200 единиц продукции. От поставщика A4 к потребителю B4 будем доставлять min = { 100 , 200 } = 100 единиц продукции. Разместим в ячейку A4B4 значение равное 100. Мы полностью израсходовали запасы поставщика A4. Исключаем строку 4 таблицы из дальнейшего рассмотрения.

Поставщик	Потребитель					Запас
	B1	B2	B3	B4	B5	
A1	4	3	5	100/2	3	100
A2	7	150/1	2	3	1	150
A3	9	50/2	200/4	5	6	350
A4	100/1	3	6	100/4	10	200
A5	5	8	15	6	15	200
Потребность	100	200	200	300	200	

Минимальный элемент матрицы тарифов находится в ячейке A3B4 и равен 5, т.е. из незадействованных маршрутов, маршрут доставки продукции от поставщика A3 к потребителю B4 наиболее рентабельный. Запасы поставщика A3 составляют 100 единиц продукции. Потребность потребителя B4 составляет 100 единиц продукции. От поставщика A3 к потребителю B4 будем доставлять 100 единиц продукции. Разместим в ячейку A3B4 значение равное 100. Мы полностью израсходовали запасы поставщика A3. Исключаем строку 3 таблицы из дальнейшего рассмотрения. Мы полностью удовлетворили потребность потребителя B4. Исключаем столбец 4 таблицы из дальнейшего рассмотрения.

Поставщик	Потребитель					Запас
	B1	B2	B3	B4	B5	
A1	4	3	5	100/2	3	100
A2	7	150/1	2	3	1	150
A3	9	50/2	200/4	100/5	6	350
A4	100/1	3	6	100/4	10	200
A5	5	8	15	6	15	200
Потребность	100	200	200	300	200	

Минимальный элемент матрицы тарифов находится в ячейке A5B5 и равен 15, т.е. из незадействованных маршрутов, маршрут доставки продукции от поставщика A5 к потребителю B5 наиболее рентабельный. Запасы поставщика A5 составляют 200 единиц продукции. Потребность потребителя B5 составляет 200 единиц продукции. От поставщика A5 к потребителю B5 будем доставлять 200 единиц продукции. Разместим в ячейку A5B5 значение равное 200. Мы полностью израсходовали запасы поставщика A5. Исключаем строку 5 таблицы из дальнейшего рассмотрения. Мы полностью удовлетворили потребность потребителя B5. Исключаем столбец 5 таблицы из дальнейшего рассмотрения.

Поставщик	Потребитель					Запас
	B1	B2	B3	B4	B5	
A1	4	3	5	100/2	3	100
A2	7	150/1	2	3	1	150
A3	9	50/2	200/4	100/5	6	350
A4	100/1	3	6	100/4	10	200

A5	5	8	15	6	15	200
Потребность	100	200	200	300	200	

Мы нашли начальное решение, т.е израсходовали все запасы поставщиков и удовлетворили все потребности потребителей.

S0 = 2 * 100 + 1 * 150 + 2 * 50 + 4 * 200 + 5 * 100 + 1 * 100 + 4 * 100 + 15 * 200 = 5250 ден. ед.

Пример разработанной программы, решающую поставленную задачу в общем виде

Рассмотрим решение задачи и нахождения минимальной суммы затрат на перевозку и распределения деталей и товаров. Задача будет решена методом минимальной стоимости, методом северо-западного угла и методом аппроксимации Фогеля. В примере рассматриваются задачи, в которых необходимо из фиксированного набора деталей, цена которых отличается, а количество ограниченно, определить какое предложение наиболее выгодно по цене. Результатом выполнения является программная реализация алгоритма нахождения наименьших затрат на поставку.

На рисунке 2 приведен пример основной формы приложения.

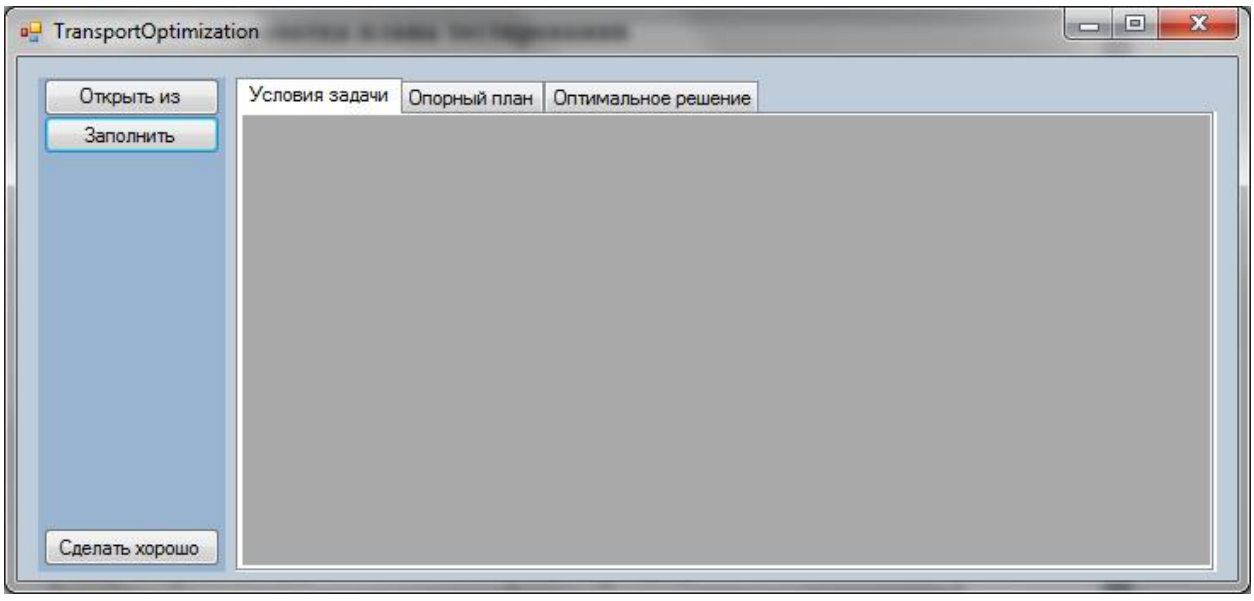
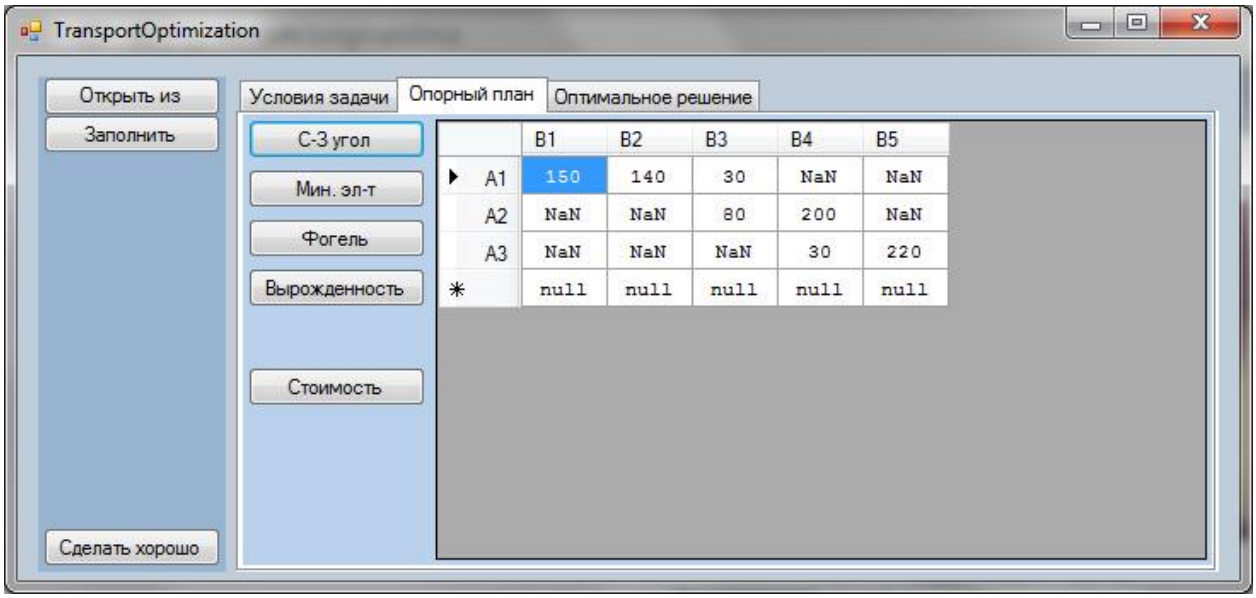


Рисунок 2 – Основное окно программы

После заполнения исходных данных можно перейти к выбору опорного плана и метода решения задачи (рисунок 3).



.

Рисунок 3 – Окно выбора метода решения

Вкладка оптимальное решение отражает полное решение задачи (рисунок 4)

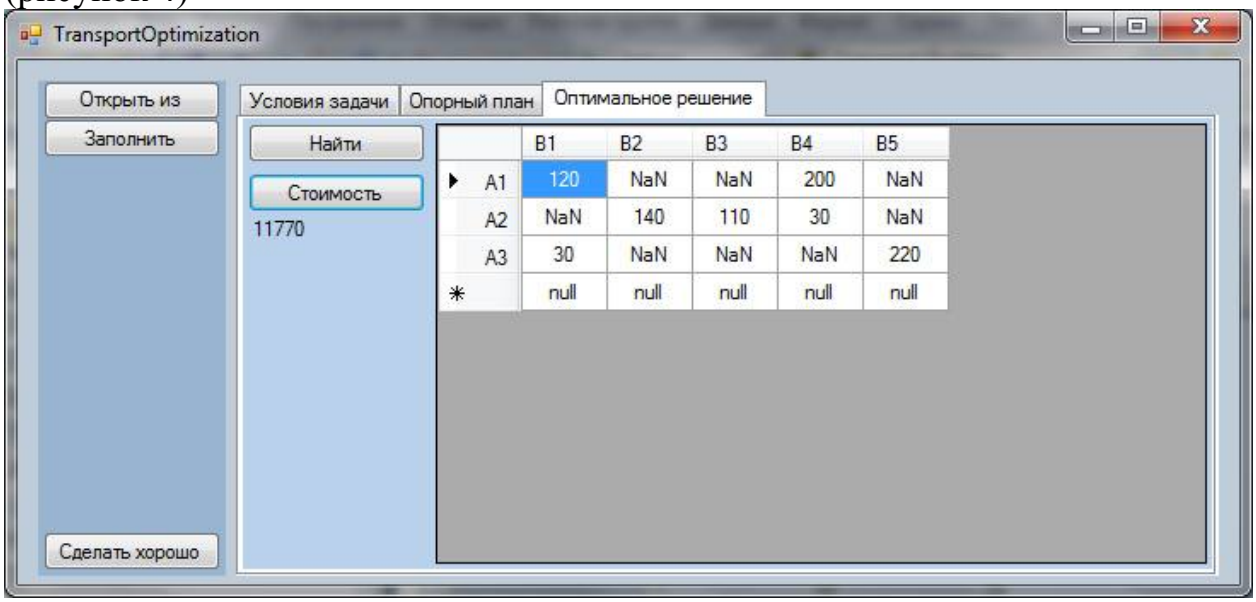


Рисунок 4 – Окно выбора оптимального решения

Листинг программы:

```
using System;
using System.IO;
using System.Windows.Forms;
using System.Drawing;
namespace lab2
{
    public partial class Form1 : Form
  {
    TransportProblem TP = null;
    float[,] SupportPlan = null;
```

```
float[,] Optimum;

public Form1()
{
   InitializeComponent();
}

private void gridA_RowsAdded(object sender, DataGridViewRowsAddedEventArgs e)
{
   gridC.RowCount++;
}

private void gridA_RowsRemoved(object sender, DataGridViewRowsRemovedEventArgs e)
{
   if (gridC.RowCount > 0) gridC.RowCount--;
}

private void CreateColumnsHeaders(DataGridView grid)
{
   grid.Columns.Clear();
   DataGridViewColumn[] newColumns = new DataGridViewColumn[gridB.Rows.Count - 1];
   for (int i = 0; i < newColumns.Length; i++)
   {
      DataGridViewTextBoxColumn ColC = new System.Windows.Forms.DataGridViewTextBoxColumn();
      grid.Columns.AddRange(new System.Windows.Forms.DataGridViewColumn[] { ColC });
      ColC.FillWeight = 50F;
      ColC.HeaderText = "B" + (grid.Columns.Count).ToString();
      ColC.Name = (grid.Columns.Count).ToString();
      ColC.Width = 50;
   }
}

private void gridB_RowsAdded(object sender, DataGridViewRowsAddedEventArgs e)
{
   CreateColumnsHeaders(gridC);
}

private void gridB_RowsRemoved(object sender, DataGridViewRowsRemovedEventArgs e)
{
   gridC.Columns.Remove((e.RowIndex + 1).ToString());
   int i = 1;
   foreach(DataGridViewTextBoxColumn Col in gridC.Columns)
   {
      Col.Name = i.ToString();
      Col.HeaderText = Col.Name;
      i++;
   }
}

private void btnOpen_Click(object sender, EventArgs e)
{
   gridA.RowCount = 1;
   gridB.RowCount = 1;
   gridC.ColumnCount = 1;
   gridC.RowCount = 1;
   Stream myStream = null;
   OpenFileDialog openFileDialog1 = new OpenFileDialog();

   openFileDialog1.InitialDirectory = "D:\\";
   openFileDialog1.Filter = "txt files (*.txt)|*.txt|All files (*.*)|*.*";
   openFileDialog1.FilterIndex = 1;
   openFileDialog1.RestoreDirectory = true;
```

```
    if (openFileDialog1.ShowDialog() == DialogResult.OK)
    {
      try
      {
        if ((myStream = openFileDialog1.OpenFile()) != null)
        {
          StreamReader SR = new StreamReader(myStream);
          String[] Sizes = SR.ReadLine().Split(' ');
          int Asize = 0, Bsize = 0;
          int.TryParse(Sizes[0], out Asize);
          int.TryParse(Sizes[1], out Bsize);
          String A = SR.ReadLine();
          String B = SR.ReadLine();
          String[] C = new String[Asize];
          for (int i = 0; i < Asize; i++) C[i] = SR.ReadLine();
          try
          {
            TP = new TransportProblem(Asize, Bsize, A, B, C);
          }
          catch(Exception exc)
          { MessageBox.Show(exc.Message); }
        }
        myStream.Close();
      }
      catch (Exception ex)
      {
        MessageBox.Show("Error: Could not read file from disk. Original error: " + ex.Message);
      }
    }
  }

  private void FillBigGrid(DataGridView grid, float[,] arr)
  {
    DataGridViewRow DataGridRow = new DataGridViewRow();
    for (int k = 0; k < TP.ASize; k++)
    {
      DataGridRow = new DataGridViewRow();
      DataGridRow.CreateCells(grid);
      for (int j = 0; j < TP.BSize; j++) DataGridRow.Cells[j].Value = arr[k, j].ToString();
      DataGridRow.HeaderCell.Value = "A" + (k + 1).ToString();
      grid.Rows.Insert(grid.Rows.Count - 1, DataGridRow);
    }
  }

  private void FillGrids()
  {
    for (int i = 0; i < TP.ASize; i++) gridA.Rows.Add(TP.mA[i].ToString());
    for (int i = 0; i < TP.BSize; i++) gridB.Rows.Add(TP.mB[i].ToString());
    FillBigGrid(gridC, TP.mC);
  }

  private void btnFillGrids_Click(object sender, EventArgs e)
  {
    gridA.Visible = true;
    gridB.Visible = true;
    gridC.Visible = true;
    FillGrids();
  }

  private void btnNordWest_Click(object sender, EventArgs e)
  {
```

```
    CreateColumnsHeaders(gridSupport);
    SupportPlan = TP.NordWest();
    FillBigGrid(gridSupport, SupportPlan);
}

private void btnMinEl_Click(object sender, EventArgs e)
{
    CreateColumnsHeaders(gridSupport);
    SupportPlan = TP.MinEl();
    FillBigGrid(gridSupport, SupportPlan);
}

private void btnCheck_Click(object sender, EventArgs e)
{
    int N = 0;
    for (int i = 0; i < SupportPlan.Length; i++)
    {
        int j = (i - i % TP.BSize) / TP.BSize;
        int k = i % TP.BSize;
        if (SupportPlan[j, k] == SupportPlan[j, k]) N++;
    }
    lblCheck1.Text = "N = " + TP.ASize.ToString() + " + " + TP.BSize.ToString() + " - 1";
    if (N == TP.ASize + TP.BSize - 1)
    {
        lblCheck1.Text += " = " + N.ToString();
        lblCheck2.Text = "==> OK";
    }
    else
    {
        lblCheck1.Text = " <> " + N.ToString();
        lblCheck2.Text = "==> не OK";
    }
}

private void btnFirstPay_Click(object sender, EventArgs e)
{
    float Sum = 0;
    for (int i = 0; i < SupportPlan.Length; i++)
    {
        int j = (i - i % TP.BSize) / TP.BSize;
        int k = i % TP.BSize;
        if (SupportPlan[j, k] == SupportPlan[j, k])
            Sum += SupportPlan[j, k] * TP.mC[j, k];
    }
    lblFirstPay.Text = Sum.ToString();
}

private void btnMakeGood_Click(object sender, EventArgs e)
{
    btnOpen_Click(null, null);
    btnFillGrids_Click(null, null);
    btnNordWest_Click(null, null);

}

private void button1_Click_1(object sender, EventArgs e)
{
    CreateColumnsHeaders(gridFinal);
    Optimum = TP.PotenMeth(SupportPlan);
    FillBigGrid(gridFinal, Optimum);
}
```

```
        private void btnStep2_Click(object sender, EventArgs e)
        {
            float Sum = 0;
            for (int i = 0; i < Optimum.Length; i++)
            {
                int j = (i - i % TP.BSize) / TP.BSize;
                int k = i % TP.BSize;
                if (Optimum[j, k] == Optimum[j, k])
                    Sum += Optimum[j, k] * TP.mC[j, k];
            }
            lblOptimum.Text = Sum.ToString();
        }

        private void gridC_CellContentClick(object sender, DataGridViewCellEventArgs e)
        {

        }

    }

    public class TransportProblem
    {
        class InvalidInpFormat : ApplicationException
        {
            public InvalidInpFormat() : base() { }
            public InvalidInpFormat(string str) : base(str) { }
            public override string ToString()
            {
                return Message;
            }
        }
        public float[] mA;
        public float[] mB;
        public float[,] mC;
        public int ASize;
        public int BSize;
        private Point[] cycle;

        public TransportProblem(float[] nA, float[] nB, float[,] nC)
        {
            if ((nA.Length != nC.GetLength(0))||(nB.Length != nC.GetLength(1)))
                throw new InvalidInpFormat("Размеры массива затрат не соответствуют размерам массивов
поставщиков и складов");
            this.mA = nA; this.mB = nB; this.mC = nC;
            this.ASize = nA.Length; this.BSize = nB.Length;
        }
        public TransportProblem(int _Asize, int _Bsize, string sA, string sB, string[] sC)
        {
            ASize = _Asize; BSize = _Bsize;
            float x = 0;
            string[] StrArr = sA.Split(' ');
            if(StrArr.Length != ASize)
                throw new InvalidInpFormat("Размеры массива A не соответствуют заявленным");
            mA = new float[ASize];
            for (int i = 0; i < mA.Length; i++) if (float.TryParse(StrArr[i], out x)) mA[i] = x;

            StrArr = sB.Split(' ');
            if (StrArr.Length != BSize)
                throw new InvalidInpFormat("Размеры массива B не соответствуют заявленным");
            mB = new float[BSize];
            for (int i = 0; i < mB.Length; i++) if (float.TryParse(StrArr[i], out x)) mB[i] = x;
```

```
            float sumA = 0;
            Array.ForEach(mA, delegate (float f) { sumA += f; });
            float sumB = 0;
            Array.ForEach(mB, delegate(float f) { sumB += f; });
            float dif = sumA - sumB;
            if (dif > 0)
            {
                float[] bufArr = mB;
                mB = new float[bufArr.Length + 1];
                bufArr.CopyTo(mB, 0);
                mB[mB.Length - 1] = Math.Abs(dif);
                BSize++;
            }
            else if (dif < 0)
            {
                float[] bufArr = mA;
                mA = new float[bufArr.Length + 1];
                bufArr.CopyTo(mA, 0);
                mA[mA.Length - 1] = Math.Abs(dif);
                ASize++;
            }

            mC = new float[ASize,BSize];
            for (int j = 0; j < sC.Length; j++)
            {
                StrArr = sC[j].Split(' ');
                if (StrArr.Length != _Bsize)
                    throw new InvalidInpFormat("Длина одной из строк входного файла не соответствует длине
массива B");
                for (int i = 0; i < _Bsize; i++) if (float.TryParse(StrArr[i], out x)) mC[j,i] = x;
            }
        }
        bool isEmpty(float[] arr)
        {
            return Array.TrueForAll(arr, delegate(float x) { return x == 0; });
        }

        private void NanToEmpty(float[,] outArr)
        {
            int i = 0, j = 0;
            for (i = 0; i < ASize; i++)
                for (j = 0; j < BSize; j++)
                    if (outArr[i, j] == 0) outArr[i, j] = float.NaN;
        }

        float findMin(float[,] Arr, bool[,] pr, out int indi, out int indj)
        {
            indi = -1; indj = -1;
            float min = float.MaxValue;
            for (int i = 0; i < ASize; i++)
                for (int j = 0; j < BSize; j++)
                    if ((pr[i, j]) && (Arr[i, j] < min))
                    {
                        min = Arr[i, j];
                        indi = i; indj = j;
                    }
            return min;
        }
        //
        public float[,] VolgelsMethod()
        {
            return null;
```

```
}

// Метод северо-западного угла
public float[,] NordWest()
{
   float[] Ahelp = mA;
   float[] Bhelp = mB;
   int i = 0, j = 0;
   float[,] outArr = new float[ASize,BSize];
   NanToEmpty(outArr);
   while (!(isEmpty(Ahelp) && isEmpty(Bhelp)))
   {
      float Dif = Math.Min(Ahelp[i],Bhelp[j]);
      outArr[i, j] = Dif;
      Ahelp[i] -= Dif; Bhelp[j] -= Dif;
      if ((Ahelp[i] == 0) && (Bhelp[j] == 0) && (j + 1 < BSize)) outArr[i, j + 1] = 0;
      if (Ahelp[i] == 0) i++;
      if (Bhelp[j] == 0) j++;
   }
   return outArr;
}

class FindWay
{
   FindWay Father;
   Point Root;
   FindWay[] Childrens;
   Point[] mAllowed;
   Point Begining;
   bool flag;
   public FindWay(int x, int y, bool _flag, Point[] _mAllowed, Point _Beg, FindWay _Father)
   {
      Begining = _Beg;
      flag = _flag;
      Root = new Point(x, y);
      mAllowed = _mAllowed;
      Father = _Father;
   }
   public Boolean BuildTree()
   {
      Point[] ps = new Point[mAllowed.Length];
      int Count = 0;
      for (int i = 0; i < mAllowed.Length; i++)
         if (flag)
         {
            if (Root.Y == mAllowed[i].Y)
            {
               Count++;
               ps[Count - 1] = mAllowed[i];
            }

         }
         else
            if (Root.X == mAllowed[i].X)
            {
               Count++;
               ps[Count - 1] = mAllowed[i];
            }

      FindWay fwu = this;
      Childrens = new FindWay[Count];
      int k = 0;
```

```
        for (int i = 0; i < Count; i++)
        {
            if (ps[i] == Root) continue;
            if (ps[i] == Begining)
            {
                while (fwu != null)
                {
                    mAllowed[k] = fwu.Root;
                    fwu = fwu.Father;
                    k++;
                };
                for (; k < mAllowed.Length; k++) mAllowed[k] = new Point(-1, -1);
                return true;
            }

            if(!Array.TrueForAll<Point>(ps,p => ((p.X == 0)&&(p.Y==0))))
                {
                    Childrens[i] = new FindWay(ps[i].X, ps[i].Y, !flag, mAllowed, Begining, this);
                    Boolean result = Childrens[i].BuildTree();
                    if (result) return true;
                }
        }
        return false;
    }

}

// Метод минимального элемента
public float[,] MinEl()
{
    float[] Ahelp = this.mA;
    float[] Bhelp = this.mB;
    int i = 0;
    int j = 0;
    float min = float.MaxValue;
    float[,] outArr = new float[this.ASize, this.BSize];
    bool[,] pArr = new bool[this.ASize, this.BSize];
    for (i = 0; i < this.ASize; i++)
    {
        for (j = 0; j < this.BSize; j++)
        {
            pArr[i, j] = true;
        }
    }
    i = 0;
    j = 0;
    int k;
    int count = 0;
    while (!this.isEmpty(Ahelp) || !this.isEmpty(Bhelp))
    {
        min = this.findMin(this.mC, pArr, out i, out j);
        float Dif = Math.Min(Ahelp[i], Bhelp[j]);
        outArr[i, j] += Dif; count++;
        Ahelp[i] -= Dif;
        Bhelp[j] -= Dif;
        if (Ahelp[i] == 0f)
        {
            k = 0;
            while (k < this.BSize)
            {
                pArr[i, k] = false;
                k++;
```

```
            }
        }
        if (Bhelp[j] == 0f)
        {
            for (k = 0; k < this.ASize; k++)
            {
                pArr[k, j] = false;
            }
        }
    }
    this.NanToEmpty(outArr);
    int difference = (ASize + BSize - 1) - count;
    for (int l = 0; l < difference; l++)
    {
        Allowed = new Point[count + 1];
        k = 0;
        for (i = 0; i < ASize; i++)
            for (j = 0; j < BSize; j++)
                if (outArr[i, j] == outArr[i, j])
                {
                    Allowed[k] = new Point(i, j);
                    k++;
                }

        Boolean p = true;
        Point Nl = new Point(0, 0);
        for (i = 0; (i < ASize) && p; i++)
            for (j = 0; (j < BSize) && p; j++)
            {
                Nl = Allowed[9] = new Point(i, j);
                FindWay fw = new FindWay(i, j, true, Allowed, new Point(i, j), null);
                p = fw.BuildTree();
            }
        if (!p) outArr[Nl.X, Nl.Y] = 0;
    }

    return outArr;
}

// Оптимизация методом потенциалов
private void FindUV(float[] U, float[] V, float[,] HelpMatr)
{
    bool[] U1 = new bool[ASize];
    bool[] U2 = new bool[ASize];
    bool[] V1 = new bool[BSize];
    bool[] V2 = new bool[BSize];
    while (!(AllTrue(V1) && AllTrue(U1)))
    {
        int i = -1;
        int j = -1;
        for (int i1 = BSize - 1; i1 >= 0; i1--)
            if (V1[i1] && !V2[i1]) i = i1;
        for (int j1 = ASize - 1; j1 >= 0; j1--)
            if (U1[j1] && !U2[j1]) j = j1;

        if ((j == -1) && (i == -1))
        for (int i1 = BSize - 1; i1 >= 0; i1--)
            if (!V1[i1] && !V2[i1])
            {
                i = i1;
                V[i] = 0;
                V1[i] = true;
```

```
            break;
          }
        if ((j == -1)&&(i == -1))
        for (int j1 = ASize - 1; j1 >= 0; j1--)
          if (!U1[j1] && !U2[j1])
          {
            j = j1;
            U[j] = 0;
            U1[j] = true;
            break;
          }

        if (i != -1)
        {
          for (int j1 = 0; j1 < ASize; j1++)
          {
            if (!U1[j1]) U[j1] = HelpMatr[j1, i] - V[i];
            if (U[j1] == U[j1]) U1[j1] = true;
          }
          V2[i] = true;
        }

        if (j != -1)
        {
          for (int i1 = 0; i1 < BSize; i1++)
          {
            if (!V1[i1]) V[i1] = HelpMatr[j, i1] - U[j];
            if (V[i1] == V[i1]) V1[i1] = true;
          }
          U2[j] = true;
        }

    }
    int rt = 0;
}

private Boolean AllPositive(float[,] m)
{
    Boolean p = true;
    for (int i = 0; (i < ASize) && p; i++)
      for (int j = 0; (j < BSize) && p; j++)
        if (m[i, j] < 0) p = false;
    return p;
}

private bool AllTrue(bool[] arr)
{
    return Array.TrueForAll(arr, delegate(bool x) { return x; });
}

private float[,] MakeSMatr(float[,] M, float[] U, float[] V)
{

    float[,] HM = new float[ASize, BSize];
    for (int i = 0; i < ASize; i++)
      for (int j = 0; j < BSize; j++)
      {
        HM[i, j] = M[i, j];
        if (HM[i, j] != HM[i, j])
          HM[i, j] = mC[i, j] - (U[i] + V[j]);
      }
    return HM;
```

```
}

private Point[] Allowed;

public int[] arra = new int[5];

private Point[] GetCycle(int x, int y)
{
   Point Beg = new Point(x, y);
   FindWay fw = new FindWay(x, y, true, Allowed, Beg, null);
   fw.BuildTree();
   Point[] Way = Array.FindAll<Point>(Allowed, delegate(Point p) { return (p.X != -1) && (p.Y != -1); });
   return Way;
}

private void Roll(float[,] m, float[,] sm)
{
   Point minInd = new Point();
   float min = float.MaxValue;
   int k = 0;
   Allowed = new Point[ASize+BSize];
   for (int i = 0; i < ASize; i++)
      for (int j = 0; j < BSize; j++)
      {
         if (m[i, j] == m[i, j])
         {
            Allowed[k].X = i;
            Allowed[k].Y = j;
            k++;
         }
         if (sm[i, j] < min)
         {
            min = sm[i, j];
            minInd.X = i;
            minInd.Y = j;
         }
      }
   Allowed[Allowed.Length - 1] = minInd;
   Point[] Cycle = GetCycle(minInd.X, minInd.Y);
   float[] Cycles = new float[Cycle.Length];
   Boolean[] bCycles = new Boolean[Cycle.Length];
   for (int i = 0; i < bCycles.Length; i++)
      bCycles[i] = i == bCycles.Length - 1 ? false : true;
   min = float.MaxValue;
   for (int i = 0; i < Cycle.Length; i++)
   {
      Cycles[i] = m[Cycle[i].X, Cycle[i].Y];
      if ((i % 2 == 0) && (Cycles[i] == Cycles[i]) && (Cycles[i] < min))
      {
         min = Cycles[i];
         minInd = Cycle[i];
      }
      if (Cycles[i] != Cycles[i]) Cycles[i] = 0;
   }
   int point1 = 0;

   for (int i = 0; i < Cycle.Length; i++)
   {
      if (i % 2 == 0)
      {
         Cycles[i] -= min;
         m[Cycle[i].X, Cycle[i].Y] -= min;
```

```
            }
            else
            {
                Cycles[i] += min;
                if (m[Cycle[i].X, Cycle[i].Y] != m[Cycle[i].X, Cycle[i].Y]) m[Cycle[i].X, Cycle[i].Y] = 0;
                m[Cycle[i].X, Cycle[i].Y] += min;
            }
        }
        m[minInd.X, minInd.Y] = float.NaN;
    }

    public float[,] PotenMeth(float[,] SupArr)
    {

        int i = 0, j = 0;
        float[,] HelpMatr = new float[ASize, BSize];
        for (i = 0; i < ASize; i++)
            for (j = 0; j < BSize; j++)
                if (SupArr[i, j] == SupArr[i, j]) HelpMatr[i, j] = mC[i, j];
                else HelpMatr[i, j] = float.NaN;

        float[] U = new float[ASize];
        float[] V = new float[BSize];
        FindUV(U, V, HelpMatr);
        float[,] SMatr = MakeSMatr(HelpMatr, U, V);
        while (!AllPositive(SMatr))
        {
            Roll(SupArr, SMatr);
            for (i = 0; i < ASize; i++)
                for (j = 0; j < BSize; j++)
                {
                    if (SupArr[i, j] == float.PositiveInfinity)
                        {
                            HelpMatr[i, j] = mC[i,j];
                            SupArr[i, j] = 0;
                            continue;
                        }
                    if(SupArr[i, j] == SupArr[i, j]) HelpMatr[i, j] = mC[i, j];
                    else HelpMatr[i, j] = float.NaN;
                }
            FindUV(U, V, HelpMatr);
            SMatr = MakeSMatr(HelpMatr, U, V);
        }

        return SupArr;
    }

  }
}
```

Задачи для самостоятельного решения

В лабораторной работе необходимо выполнить математическое решение и программную реализацию методами северо-западного угла, наименьшей стоимости и Фогеля следующей задачи: у поставщиков A1, A2, …, An определенное количество единиц однотипной продукции, которая должна быть доставлена потребителям B1, B2, …, Bn в нужном им

количестве. Стоимость доставки единицы продукции от поставщика А к указанным потребителям В, а также данные о грузоотправителях (А) и грузополучателях (В) приведены в таблице. Требуется найти оптимальное решение доставки продукции от поставщиков к потребителям, минимизирующие стоимость доставки.

Таблицы исходных значений распределены по вариантам.

Задача 1

Поставщик	Потребитель				Запас
	B1	B2	B3	B4	
A1	9	9	8	3	100
A2	2	6	4	4	400
A3	8	9	3	4	300
A4	3	3	5	2	600
A5	9	9	1	6	900
Потребность	500	600	500	700	

Задача 2

Поставщик	Потребитель					Запас
	B1	B2	B3	B4	B5	
A1	1	3	6	6	9	300
A2	2	5	7	2	7	700
A3	9	2	4	5	6	500
A4	5	6	6	7	5	500
A5	1	8	4	7	5	300
Потребность	400	600	400	600	300	

Задача 3

Поставщик	Потребитель					Запас
	B1	B2	B3	B4	B5	

A1	3	5	7	2	8	500
A2	6	5	6	8	7	400
A3	9	1	2	4	6	800
A4	3	5	8	9	6	500
Потребность	300	500	500	600	300	

Задача 4

Поставщик	Потребитель				Запас
	B1	B2	B3	B4	
A1	4	1	8	2	800
A2	4	1	3	2	900
A3	9	1	1	3	500
A4	2	6	5	8	400
A5	1	6	6	4	500
Потребность	900	600	900	700	

Задача 5

Поставщик	Потребитель					Запас
	B1	B2	B3	B4	B5	
A1	1	3	6	6	9	600
A2	2	5	7	2	7	300
A3	9	2	4	5	6	800
A4	5	6	6	7	5	300
A5	1	8	4	7	5	200
Потребность	400	400	800	500	100	

Задача 6

Поставщик	Потребитель					Запас
	B1	B2	B3	B4	B5	

A1	8	4	7	2	2	600
A2	6	3	5	6	2	400
A3	9	9	2	1	9	100
A4	8	4	7	5	6	900
Потребность	500	400	600	200	400	

Задача 7

Поставщик	Потребитель				Запас
	B1	B2	B3	B4	
A1	2	8	3	2	100
A2	6	3	6	7	500
A3	6	6	5	3	700
A4	2	6	5	8	500
A5	2	8	1	7	400
Потребность	500	400	600	700	

Задача 8

Поставщик	Потребитель					Запас
	B1	B2	B3	B4	B5	
A1	1	9	4	1	3	500
A2	2	6	8	2	8	900
A3	9	7	4	9	6	400
A4	6	7	5	3	7	100
A5	9	7	2	1	9	900
Потребность	700	900	500	400	300	

Задача 9

Поставщик	Потребитель					Запас
	B1	B2	B3	B4	B5	

A1	5	8	9	4	3	600
A2	4	5	5	7	7	300
A3	1	3	4	1	9	600
A4	3	4	5	3	5	900
Потребность	800	500	400	500	200	

Задача 10

Поставщик	Потребитель				Запас
	B1	B2	B3	B4	
A1	2	6	4	5	500
A2	4	8	2	9	800
A3	2	3	4	1	900
A4	4	6	8	3	100
A5	4	6	1	7	600
Потребность	500	700	900	700	

Задача 11

Поставщик	Потребитель					Запас
	B1	B2	B3	B4	B5	
A1	7	7	6	1	4	500
A2	2	7	5	3	9	700
A3	7	5	4	8	6	100
A4	6	7	5	5	4	900
A5	3	4	1	1	6	600
Потребность	500	700	900	400	300	

Задача 12

Поставщик	Потребитель					Запас
	B1	B2	B3	B4	B5	

A1	6	4	9	4	7	700
A2	3	6	5	4	9	500
A3	3	1	4	2	8	400
A4	4	3	2	7	5	300
Потребность	500	700	200	400	100	

Задача 13

Поставщик	Потребитель				Запас
	B1	B2	B3	B4	
A1	5	8	3	6	300
A2	4	6	8	2	700
A3	6	9	7	2	600
A4	1	3	4	7	900
A5	5	3	8	1	500
Потребность	600	800	700	900	

Задача 14

Поставщик	Потребитель					Запас
	B1	B2	B3	B4	B5	
A1	2	5	4	8	6	400
A2	5	6	4	8	3	400
A3	4	4	6	3	4	100
A4	3	5	4	2	4	200
A5	5	4	3	1	8	500
Потребность	300	400	400	200	300	

Задача 15

Поставщик	Потребитель					Запас
	B1	B2	B3	B4	B5	

A1	6	5	7	5	7	900
A2	3	3	5	7	2	900
A3	3	5	3	2	4	800
A4	4	1	6	4	2	900
Потребность	800	900	700	400	500	

Задача 16

Поставщик	Потребитель				Запас
	B1	B2	B3	B4	
A1	3	6	7	4	200
A2	2	6	4	7	400
A3	3	5	6	4	500
A4	3	6	4	1	300
A5	2	4	3	1	700
Потребность	300	900	300	600	

Задача 17

Поставщик	Потребитель					Запас
	B1	B2	B3	B4	B5	
A1	5	4	7	2	4	900
A2	7	4	2	1	5	100
A3	4	3	2	7	8	100
A4	5	4	7	1	3	600
A5	4	5	6	2	4	400
Потребность	500	300	400	500	400	

Задача 18

Поставщик	Потребитель					Запас
	B1	B2	B3	B4	B5	

A1	4	4	6	2	6	600
A2	6	7	9	3	1	800
A3	4	6	2	1	3	200
A4	7	4	3	5	2	500
Потребность	500	300	600	600	100	

Задача 19

Поставщик	Потребитель				Запас
	B1	B2	B3	B4	
A1	3	7	4	3	300
A2	7	5	3	9	800
A3	3	7	5	8	400
A4	7	6	5	9	200
A5	6	7	8	4	900
Потребность	900	700	800	200	

Задача 20

Поставщик	Потребитель					Запас
	B1	B2	B3	B4	B5	
A1	2	5	6	9	3	300
A2	5	3	4	9	7	600
A3	2	6	4	3	2	200
A4	4	7	5	3	2	100
A5	7	3	4	6	1	900
Потребность	500	400	300	600	300	

Лабораторная работа
«Сетевые модели»

Цель работы: изучение принципов решения задач, основанных на сетевых моделях, алгоритмов построения минимального остовного дерева, нахождения кратчайшего пути, нахождения максимального потока, потока наименьшей стоимости, а также применение изученных методов на практике.

Порядок выполнения:

1. Изучить теоретические сведения.
2. Выполнить математическое описание алгоритма решения задачи.
3. Разработать программу решающую поставленную задачу в общем виде.
4. Составить отчет по работе.

Теоретические сведения

В рамках теории исследования операций рассматривается большое количество практических задач, которые можно сформулировать и решить как сетевые модели. Недавние исследования показывают, что не менее 70% реальных задач математического программирования можно представить в виде сетевых моделей. Ниже несколько конкретных примеров:

1. Проектирование газопровода, соединяющего буровые скважины морского базирования с находящейся на берегу приемной станцией. Целевая функция соответствующей модели должна минимизировать стоимость строительства газопровода.
2. Поиск кратчайшего маршрута между двумя городами по существующей сети дорог.
3. Определение максимальной пропускной способности трубопровода для транспортировки угольной пульпы от угольных шахт к электростанциям.
4. Определение схемы транспортировки нефти от пунктов нефтедобычи к нефтеперерабатывающим заводам с минимальной стоимостью транспортировки.
5. Составление временного графика строительных работ (определение дат начала и завершения отдельных этапов работ).

Решение приведенных задач (как и многих аналогичных) требует применения различных сетевых оптимизационных алгоритмов:

1. Алгоритм нахождения минимального основного дерева

2. Алгоритм поиска кратчайшего пути

3. Алгоритм определения максимального потока

4. Алгоритм минимизации стоимости потока в сети с ограниченной пропускной способностью

5. Алгоритм определения критического пути

Сеть состоит из множества узлов, связанных дугами (или ребрами). Таким образом, сеть описывается парой множеств (N, A), где N— множество узлов, а A— множество ребер. Например, сеть, показанная на рис.1, описывается следующим образом.

N={1.2,3,4,5},

A={(1, 2), (1, 3), (2, 3), (2, 5), (3, 4), (3, 5), (4, 2), (4, 5)}.

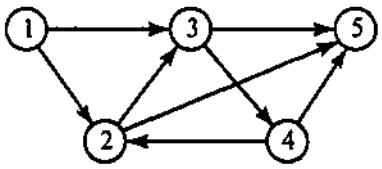


Рисунок 1 – Пример сети

С каждым типом сети связан определенный тип потоков (например, транспортный поток нефти в нефтепроводах или автомобильные потоки в сети городских дорог). В общем случае потоки в сети ограничены пропускной способностью ее ребер, которая может быть как конечной, так и бесконечной.

Ребро называется направленным, или ориентированным (и в этом случае ребро будем называть дугой), если в одном направлении возможен только положительный поток, а в противоположном — только нулевой. В ориентированной сети все ребра ориентированы.

Путем называется последовательность различных ребер, соединяющих два узла, независимо от направления потока в каждом ребре. Путь формирует цикл, если начальный и конечный узлы совпадают. Например, на рис.1 дуги (2,3), (3,4) и (4, 2) составляют цикл. Ориентированный цикл — это цикл, в котором все дуги ориентированы в определенном направлении.

Связная сеть — такая сеть, у которой любые два узла связаны по крайней мере одним путем. На рисунке1 показан именно такой тип сети. Деревом называется связная сеть, содержащая подмножество узлов исходной сети и не имеющая циклов. Остовное дерево — это дерево, содержащее все

узлы сети. На рисунке 2 показаны дерево и остовное дерево для сети из рисунок 1.

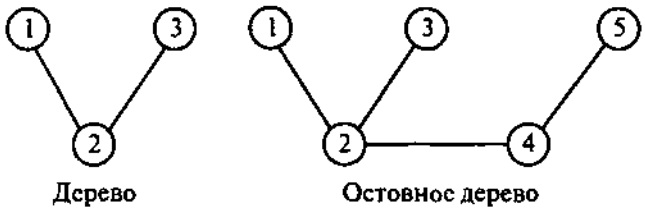


Рисунок 2 – Дерево и остовное дерево

Алгоритм нахождения минимального основного дерева

Он предполагает соединение всех узлов сети с помощью путей наименьшей длины. Типичной задачей, для решения которой необходим такой алгоритм, является создание (проектирование) сети дороге твердым покрытием, соединяющих населенные пункты в сельской местности, где дороги, соединяющие два каких-либо пункта, могут проходить через другие населенные пункты. Наиболее экономичный проект дорожной системы должен минимизировать общую длину дорог с твердым покрытием, при этом желаемый результат можно получить с помощью алгоритма построения минимального остовного дерева.

Опишем процедуру выполнения этого алгоритма. Обозначим через N={1, 2,..., n) множество узлов сети и введем новые обозначения:

C_k— множество узлов сети, соединенных алгоритмом после выполнения й-й итерации этого алгоритма,

$\bar{C}_k$ — множество узлов сети, не соединенных с узлами множества С, после выполнения к-й итерации этого алгоритма.

Этап 0. Пусть $C_0 = Ø$ и $\bar{C}_0 = N$.

Этап 1. Выбираем любой узел i из множества $\bar{C}_0$ и определяем $C_1 = \{i\}$, тогда $C_1 = N - \{i\}$. Полагаем k = 2.

Основной этап k. В множестве $\bar{C}_{k-1}$, выбираем узел j, который соединен самой короткой дугой с каким-либо узлом из множества C_{k-1}, Узел j присоединяется к множеству C_{k-1}, и удаляется из множества $\bar{C}_{k-1}$. Таким образом, $C_k = C_{k-1} + \{j\}$, $\bar{C}_k = \bar{C}_{k-1} - \{j\}$.

Если множество $\bar{C}_k$, пусто, то выполнение алгоритма заканчивается. В противном случае полагаем k = k + 1 и повторяем последний этап.

Алгоритм определения кратчайшего пути

Существуют два алгоритма для решения задачи поиска кратчайшего пути как в сетях, имеющих циклы, так и в сетях, не имеющих циклов.

1. Алгоритм Дейкстры
2. Алгоритм Флойда

Алгоритм Дейкстры разработан для поиска кратчайшего пути между заданным исходным узлом и любым другим узлом сети. Алгоритм Флойда более общий, поскольку он позволяет одновременно найти минимальные пути между любыми двумя узлами сети.

Алгоритм Дейкстры. В процессе выполнения этого алгоритма при переходе от узла i к следующему узлу j используется специальная процедура пометки ребер. Обозначим через u_1 кратчайшее расстояние от исходного узла 1 до узла i, через d_{ij} — длину ребра (i/j). Тогда для узла; определим метку [u_j, i] следующим образом.

$[u_j, i] = [u_i + d_{ij}, i], d_{ij} \geq 0$

Метки узлов в алгоритме Дейкстры могут быть двух типов: временные и постоянные. Временная метка впоследствии может быть заменена на другую временную, если будет найден более короткий путь к данному узлу. Когда же станет очевидным, что не существует более короткого пути от исходного узла к данному, статус временной метки изменяется на постоянный.

Вычислительная схема алгоритма состоит из следующих этапов.

Этап 0.

Исходному узлу (узел 1) присваивается постоянная метка [0, —]. Полагаем i = 1.

Этап 1.

a) Вычисляются временные метки $[u_i + d_{ij}, i]$ для всех узлов j, которые можно достичь непосредственно из узла i и которые не имеют постоянных ме ток. Если узел j уже имеет метку $[u_j, k]$, полученную от другого узла k, и, если $u_i + d_{ij} < u_j$, тогда метка $[u_j, k]$ заменяется на $[u_i + d_{ij}, i]$.

b) Если все узлы имеют постоянные метки, процесс вычислений заканчивается. В противном случае выбирается метка $[u_r, s]$ с наименьшим значением расстояния u_r среди всех временных меток (если таких меток несколько, то выбор произволен). Полагаем i = r и повторяем этап i.

Алгоритм Флойда. Этот алгоритм более общий по сравнению с алгоритмом Деикстры, так как он находит кратчайшие пути между любыми двумя узлами сети. В этом алгоритме сеть представлена в виде квадратной матрицы с n строками и n столбцами. Элемент (i, j) равен расстоянию d_{ij} (от узла i к узлу j, которое имеет конечное значение, если существует дуга (i, j), и равен бесконечности в противном случае.

Покажем сначала основную идею метода Флойда. Пусть есть три узла i, j в k и заданы расстояния между ними (рис.3). Если выполняется

неравенство $d_{ij} + d_{jk} < d_{ik}$ то целесообразно заменить путь i → k путем i → j → k. Такая замена (далее ее будем условно называть треугольным оператором) выполняется систематически в процессе выполнения алгоритма Флойда.

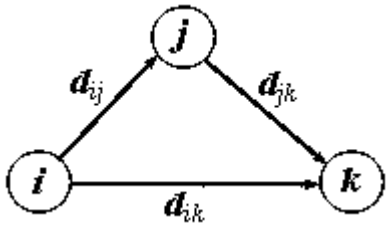


Рисунок 3 – Треугольный оператор Флойда

Определяем начальную матрицу расстояния D0 и матрицу последовательности узлов S0. Диагональные элементы обеих матриц помечаются знаком "-", показывающим, что эти элементы в вычислениях не участвуют. Полагаем k = 1:

$D_0 =$

	1	2	…	j	…	n
1	—	d_{12}	…	d_{1j}	…	d_{1n}
2	d_{21}	—	…	d_{2j}	…	d_{2n}
.	.	.	.	.	.	.
.	.	.	.	.	.	.
.	.	.	.	.	.	.
i	d_{i1}	d_{i2}	…	d_{ij}	…	d_{in}
.	.	.	.	.	.	.
.	.	.	.	.	.	.
.	.	.	.	.	.	.
n	d_{n1}	d_{n2}	…	d_{nj}	…	—

$S_0 =$

	1	2	…	j	…	n
1	—	2	…	j	…	n
2	1	—	…	j	…	n
.	.	.	.	.	.	.
.	.	.	.	.	.	.
.	.	.	.	.	.	.
i	1	2	…	j	…	n
.	.	.	.	.	.	.
.	.	.	.	.	.	.
.	.	.	.	.	.	.
n	1	2	…	j	…	—

Рисунок 4 – Начальная ситуация

Основной шаг k. Задаем строку k и столбец k как ведущую строку и ведущий столбец. Рассматриваем возможность применения треугольного оператора ко всем элементам dij матрицы Dk-1. Если выполняется неравенство dik + dkj < dij, (i <> k, j <> k, i <> j), тогда выполняем следующие действия:

- создаем матрицу Dk путем замены в матрице Dk-1 элемента dij на сумму dik + dkj,
- создаем матрицу Sk путем замены в матрице Sk-1 элемента sij на k. Полагаем k = k + 1 и повторяем шаг k.

Поясним действия, выполняемые на k-м шаге алгоритма, представив матрицу Dk-1 так, как она показана на рисунке 3. На этом рисунке строка k и столбец k являются ведущими. Строка i - любая строка с номером от 1 до k - 1, а строка p - произвольная строка с номером от k + 1 до n. Аналогично столбец j представляет любой столбец с номером от 1 до k - 1, столбец q - произвольный столбец с номером от k + 1 до n. Треугольный оператор выполняется следующим образом. Если сумма элементов ведущих строки и столбца (показанных в квадратах) меньше элементов, находящихся в пересечении столбца и строки (показанных в кружках), соответствующих рассматриваемым ведущим элементам, то расстояние (элемент в кружке) заменяется на сумму расстояний, представленных ведущими элементами:

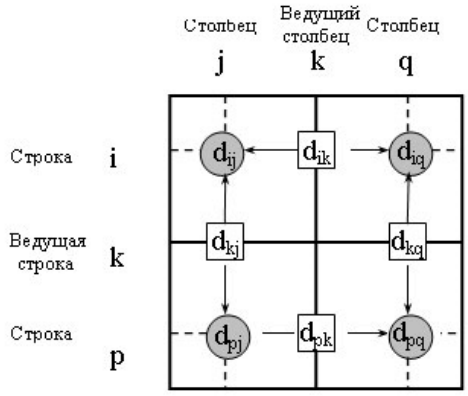


Рисунок 5 – Иллюстрация алгоритма Флойда

После реализации n шагов алгоритма определение по матрицам Dn и Sn кратчайшего пути между узлами i и j выполняется по следующим правилам.

Расстояние между узлами i и j равно элементу dij в матрице Dn.

Промежуточные узлы пути от узла i к узлу j определяем по матрице Sn. Пусть sij = k, тогда имеем путь i -> k -> j. Если далее sik = k и skj = j, тогда считаем, что весь путь определен, так как найдены все промежуточные узлы. В противном случае повторяем описанную процедуру для путей от узла i к узлу k и от узла k к узлу j.

Пример построения математической модели

Найдем для сети, показанной на рисунке 6, кратчайшие пути между любыми двумя узлами. Расстояние между узлами этой сети проставлены на рисунке возле соответствующих ребер. Ребро (3, 5) ориентированно, поэтому не допускается движение от узла 5 к узлу 3. Все остальные ребра допускают движение в обе стороны:

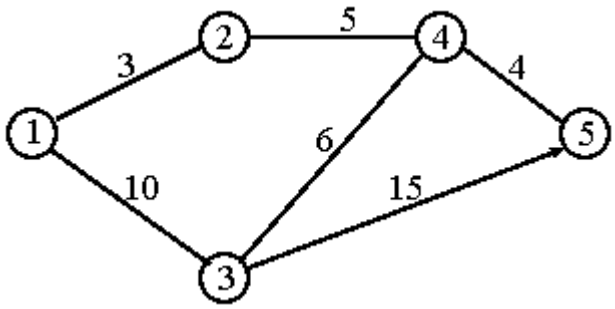


Рисунок 6 – Пример сети

Начальные матрицы D0 и S0 строятся непосредственно по заданной схеме сети. Матрица D0 симметрична, за исключением пары элементов d35 и d53, где d53 равно бесконечности, поскольку невозможен переход от узла 5 к узлу 3:

D_0

	1	2	3	4	5
1	—	3	10	∞	∞
2	3	—	∞	5	∞
3	10	∞	—	6	15
4	∞	5	6	—	4
5	∞	∞	∞	4	—

S_0

	1	2	3	4	5
1	—	2	3	4	5
2	1	—	3	4	5
3	1	2	—	4	5
4	1	2	3	—	5
5	1	2	3	4	—

Рисунок 7 – Начальное состояние

В матрице D0 выделены ведущие строка и столбец (k = 1). Двойной рамкой представлены элементы d23 и d32, единственные среди элементов матрицы D0, значения которых можно улучшить с помощью треугольного оператора. Таким образом, чтобы на основе матриц D0 и S0 получить матрицы D1 и S1, выполняем следующие действия.

Заменяем d23 на d21 + d13 = 3 + 10 = 13 и устанавливаем s23 = 1.

Заменяем d32 на d31 + d12 = 10 + 3 = 13 и устанавливаем s32 = 1.

Матрицы D1 и S1 имеют следующий вид:

D_1

	1	2	3	4	5
1	—	3	10	∞	∞
2	3	—	**13**	5	∞
3	10	**13**	—	6	15
4	∞	5	6	—	4
5	∞	∞	∞	4	—

S_1

	1	2	3	4	5
1	—	2	3	4	5
2	1	—	**1**	4	5
3	1	**1**	—	4	5
4	1	2	3	—	5
5	1	2	3	4	—

Рисунок 8 – Матрицы D1 и S1

Полагаем k = 2; в матрице D1 выделены ведущие строка и столбец. Треугольный оператор применяется к элементам матрицы D1 и S1, выделенным двойной рамкой.

В результате получаем матрицы D2 и S2:

D_2

	1	2	3	4	5
1	—	3	10	**8**	∞
2	3	—	13	5	∞
3	10	13	—	6	15
4	**8**	5	6	—	4
5	∞	∞	∞	4	—

S_2

	1	2	3	4	5
1	—	2	3	**2**	5
2	1	—	1	4	5
3	1	1	—	4	5
4	**2**	2	3	—	5
5	1	2	3	4	—

Рисунок 9 – Матрицы D2 и S2

Полагаем k = 3; в матрице D2 выделены ведущие строка и столбец. Треугольный оператор применяется к элементам матрицы D2 и S2, выделенным двойной рамкой. В результате получаем матрицы D3 и S3:

D_3

	1	2	3	4	5
1	—	3	10	8	**25**
2	3	—	13	5	**28**
3	10	13	—	6	15
4	8	5	6	—	4
5	∞	∞	∞	4	—

S_3

	1	2	3	4	5
1	—	2	3	2	**3**
2	1	—	1	4	**3**
3	1	1	—	4	5
4	2	2	3	—	5
5	1	2	3	4	—

Рисунок 10 – Матрицы D3 и S3

Полагаем k = 4, ведущие строка и столбец в матрице D3 выделены. Получаем новые матрицы D4 и S4:

D_4

	1	2	3	4	5
1	—	3	10	8	**12**
2	3	—	**11**	5	**9**
3	10	**11**	—	6	**10**
4	8	5	6	—	4
5	**12**	**9**	**10**	4	—

S_4

	1	2	3	4	5
1	—	2	3	2	**4**
2	1	—	**4**	4	**4**
3	1	**4**	—	4	**4**
4	**2**	2	3	—	5
5	**4**	**4**	**4**	4	—

Рисунок 11 – Матрицы D4 и S4

Полагаем k = 5, ведущие строка и столбец в матрице D4 выделены. Никаких действий на этом шаге не выполняем; вычисления закончены.

Конечные матрицы D4 и S4 содержат всю информацию, необходимую для определения кратчайших путей между любыми двумя узлами сети. Например, кратчайшее расстояние между узлами 1 и 5 равно d15 = 12.

Для нахождения соответствующих маршрутов напомним, что сегмент маршрута (i, j) состоит из ребра (i, j) только в том случае, когда sij = j. В противном случае узлы i и j связаны, по крайней мере, через один промежуточный узел. Например, поскольку s15 = 4 и s45 = 5, сначала кратчайший маршрут между узлами 1 и 5 будет иметь вид 1->4->5. Но так как s14 не равно 4, узлы 1 и 4 в определенном пути не связаны одним ребром (но в исходной сети они могут быть связаны непосредственно). Далее следует определить промежуточный узел (узлы) между первым и четвертым узлами. Имеем s14 = 2 и s24 = 4, поэтому маршрут 1->4 заменяем 1->2->4. Поскольку s12 = 2 и s24 = 4, других промежуточных узлов нет. Комбинируя определенные сегменты маршрута, окончательно получаем следующий кратчайший путь от узла 1 до узла 5: 1->2->4->5. Длина этого пути равна 12 километрам.

Пример разработанной программы, решающую поставленную задачу в общем виде

Пример №1.

Рассмотрим задачу, которая состоит в определении того, какие работы или операции из числа многих, составляющих проект, являются "критическими" по своему влиянию на общую календарную продолжительность проекта, и каким образом построить наилучший план проведения всех работ по данному проекту с тем, чтобы выдержать заданные сроки при минимальных затратах. В примере рассматривается задача нахождения резервов времени, критического пути для выполнения ряда работ и построение временного графика работ. Результатом выполнения является программная реализация алгоритма нахождения критического пути и его графическое представление.

Основное окно программы представлено на рисунке 12.

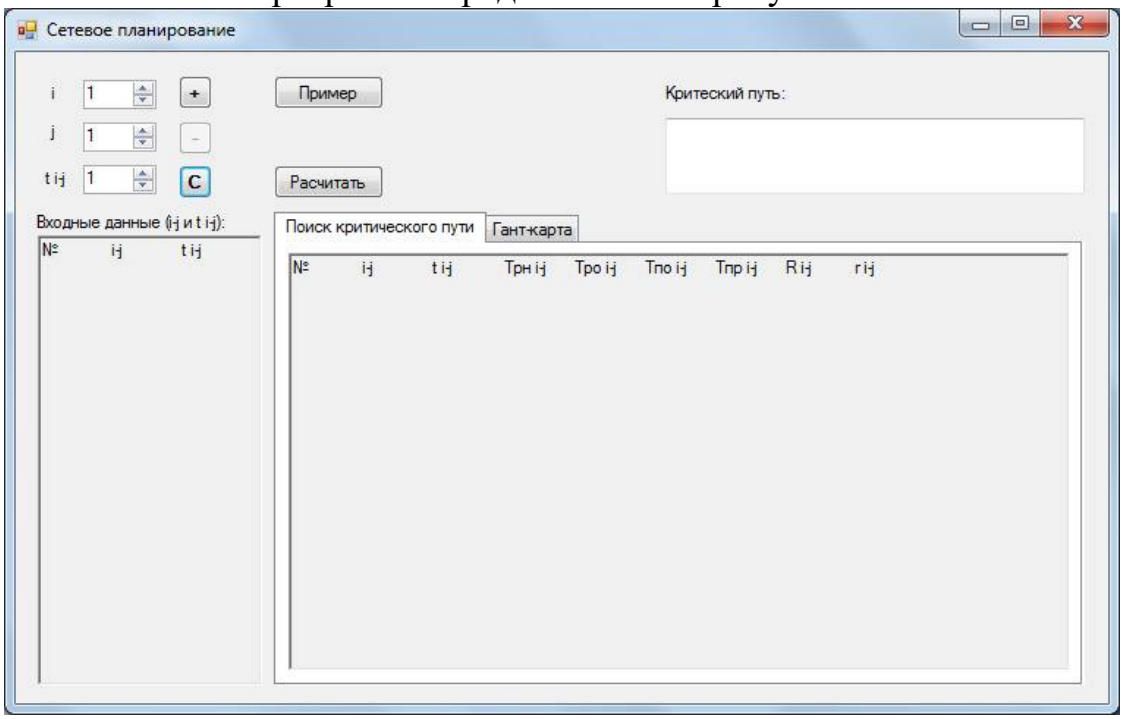


Рисунок 12 –Основное окно программы

Работа вкладки «Поиск критического пути» приведена на рисунках 13 и 14.

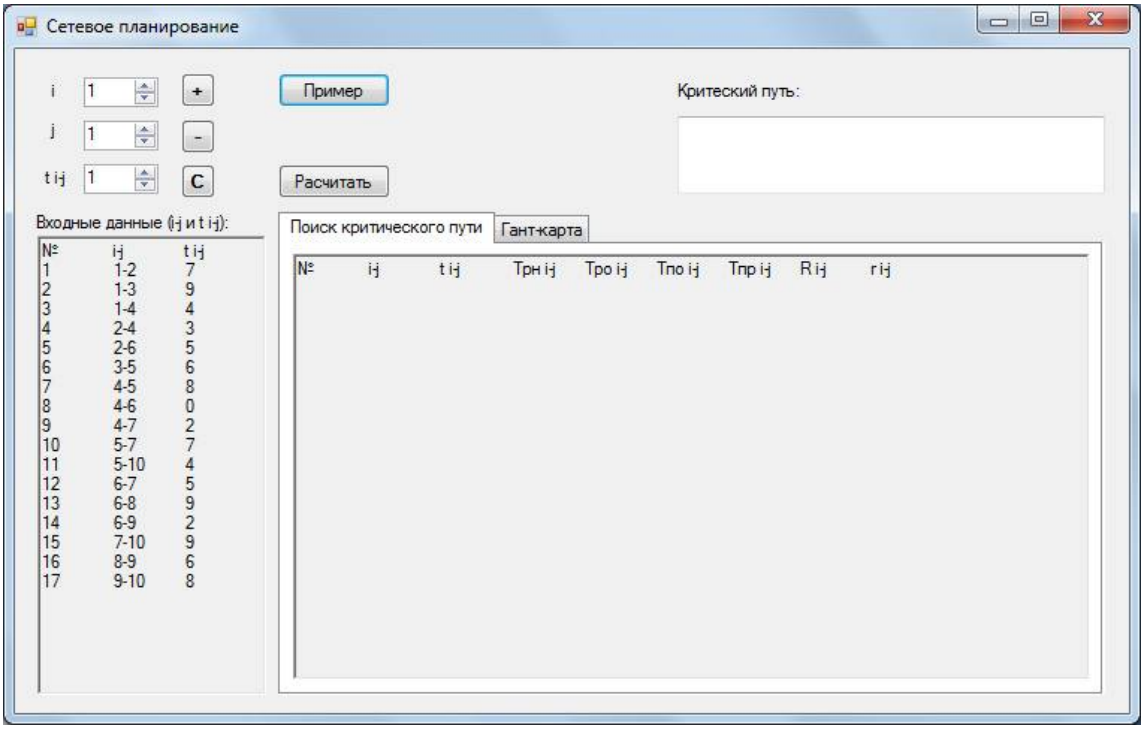


Рисунок 13 – Вкладка поиска критического пути – этап 1

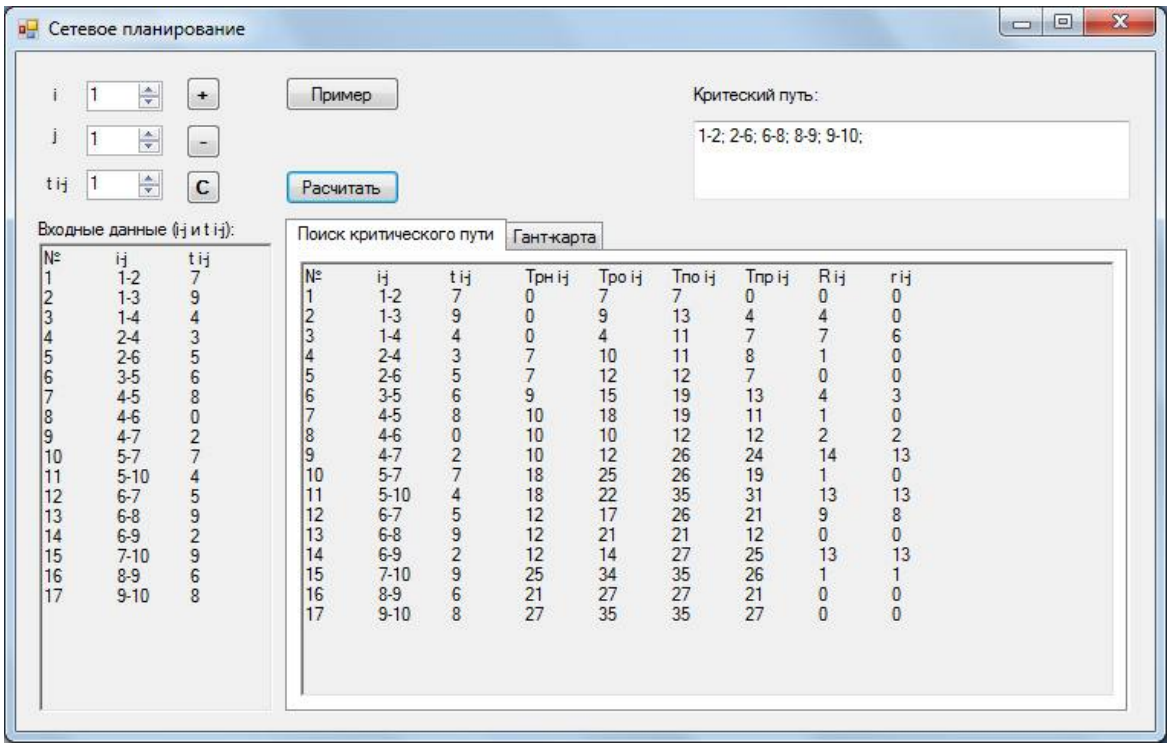


Рисунок 14 – Вкладка поиска критического пути – этап 2

Графический просмотр результатов вычислений в виде Гант-карты возможен на вкладке Гант- карта (рисунок 15).

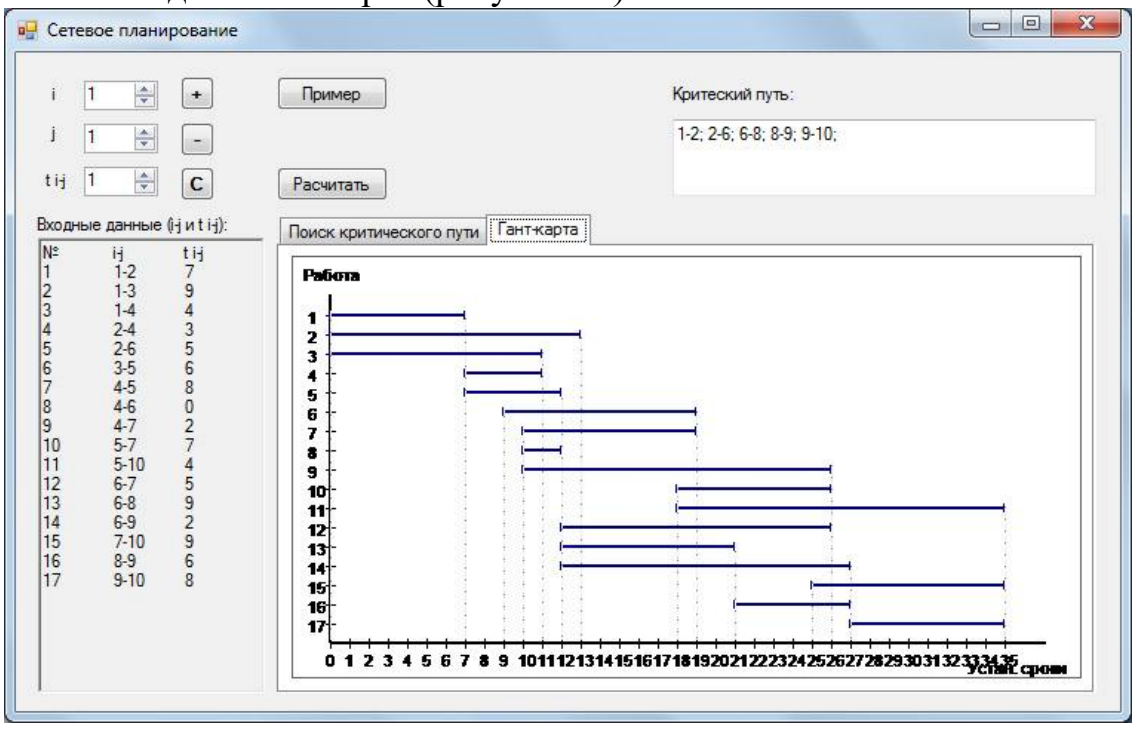


Рисунок 15 - Гант-карта

Листинг программы:

```
namespace WindowsFormsApplication1
{
   using System;
   using System.ComponentModel;
```

```
using System.Drawing;
using System.Drawing.Drawing2D;
using System.Windows.Forms;

public partial class Form1 : Form
{

    public Form1()
    {
        this.InitializeComponent();
    }

    private void button1_Click(object sender, EventArgs e)
    {
        int num;
        int num2;
        int num3;
        int num5 = 0;
        for (num = 0; num <= this.cou; num++) //начальные значения каждого события (события i-j и срок
выполнения)
        {
            if (this.mas[num, 0] > num5)
            {
                num5 = this.mas[num, 0];
            }
            if (this.mas[num, 1] > num5)
            {
                num5 = this.mas[num, 1];
            }
        }
        for (num = 1; num < num5; num++) //проход вперед для каждого события
        {
            num2 = 0;
            while (num2 <= this.cou)
            {
                if (this.mas[num2, 0] == 1)
                {
                    this.mas[num2, 3] = 0;
                    this.mas[num2, 4] = this.mas[num2, 2];
                }
                else if (this.mas[num2, 0] == num)
                {
                    this.max = 0;
                    num3 = 0;
                    while (num3 <= this.cou)
                    {
                        if ((this.mas[num3, 1] == num) && (this.mas[num3, 4] > this.max))
                        {
                            this.max = this.mas[num3, 4];
                        }
                        num3++;
                    }
                    this.mas[num2, 3] = this.max;
                    this.mas[num2, 4] = this.mas[num2, 2] + this.max;
                }
                num2++;
            }
        }
        for (num = num5; num >= 2; num--) //проход назад
        {
            this.max = 0;
            num3 = 0;
```

```
    while (num3 <= this.cou)
    {
      if ((this.mas[num3, 1] == num5) && (this.mas[num3, 4] > this.max))
      {
        this.max = this.mas[num3, 4];
      }
      num3++;
    }
    for (num2 = 0; num2 <= this.cou; num2++)
    {
      if (this.mas[num2, 1] == num5)
      {
        this.mas[num2, 5] = this.max;
        this.mas[num2, 6] = this.max - this.mas[num2, 2];
      }
      else if (this.mas[num2, 1] == num)
      {
        int num4 = this.max;
        for (num3 = 0; num3 <= this.cou; num3++)
        {
          if ((this.mas[num3, 0] == num) && (this.mas[num3, 6] < num4))
          {
            num4 = this.mas[num3, 6];
          }
        }
        this.mas[num2, 5] = num4;
        this.mas[num2, 6] = num4 - this.mas[num2, 2];
      }
    }
  }
  num3 = 0;
  while (num3 <= this.cou) //вычисление свободного резерва
  {
    this.mas[num3, 7] = this.mas[num3, 5] - this.mas[num3, 4];
    num3++;
  }
  for (num = 0; num <= this.cou; num++) //вычисление полного резерва
  {
    if (this.mas[num, 1] != num5)
    {
      num3 = 0;
      while (num3 <= this.cou)
      {
        if (this.mas[num3, 0] == this.mas[num, 1])
        {
          this.mas[num, 8] = this.mas[num3, 3] - this.mas[num, 4];
          break;
        }
        num3++;
      }
    }
    else
    {
      this.mas[num, 8] = this.max - this.mas[num, 4];
    }
  }
  string str = "";
  for (num3 = 0; num3 <= this.cou; num3++) //критический путь
  {
    if ((this.mas[num3, 7] == 0) & (this.mas[num3, 8] == 0))
    {
      str = str + Convert.ToString(this.mas[num3, 0]) + "-" + Convert.ToString(this.mas[num3, 1]) + "; ";
```

```
            }
        }
        this.textBox1.Text = str;
        this.richTextBox2.Text = "";
        this.richTextBox2.AppendText("№\ti-j\tt i-j\tТрн i-j\tТро i-j\tТпо i-j\tТпр i-j\tR i-j\tr i-j\n");
        for (num = 0; num <= this.cou; num++)
        {
            this.richTextBox2.AppendText(Convert.ToString((int)(num + 1)) + "\t" +
Convert.ToString(this.mas[num, 0]) + "-" + Convert.ToString(this.mas[num, 1]) + "\t" +
Convert.ToString(this.mas[num, 2]) + "\t" + Convert.ToString(this.mas[num, 3]) + "\t" +
Convert.ToString(this.mas[num, 4]) + "\t" + Convert.ToString(this.mas[num, 5]) + "\t" +
Convert.ToString(this.mas[num, 6]) + "\t" + Convert.ToString(this.mas[num, 7]) + "\t" +
Convert.ToString(this.mas[num, 8]) + "\n");
        }

        ////////////
        //прорисовка
        ////////////////////////////////

        int max = this.max;
        int num7 = this.cou + 1;
        int num8 = Convert.ToInt32((int)((this.pictureBox1.Width - 50) / max));
        int num9 = Convert.ToInt32((int)((this.pictureBox1.Height - 50) / num7));
        Bitmap image = new Bitmap(this.pictureBox1.Width, this.pictureBox1.Height);
        Graphics graphics = Graphics.FromImage(image);
        graphics.DrawLine(new Pen(Color.Black, 2f), 0x19, 0x19, 0x19, this.pictureBox1.Height - 0x19);
        graphics.DrawLine(new Pen(Color.Black, 2f), 0x19, this.pictureBox1.Height - 0x19, this.pictureBox1.Width
- 0x19, this.pictureBox1.Height - 0x19);
        for (num = 0; num <= max; num++)
        {
            graphics.DrawString(Convert.ToString(num), new Font("Arial", 8f), new SolidBrush(Color.Black),
(float)((num * num8) + 20), (float)(this.pictureBox1.Height - 20));
            graphics.DrawLine(new Pen(Color.Black, 1f), (int)((num * num8) + 0x19), (int)(this.pictureBox1.Height -
0x1c), (int)((num * num8) + 0x19), (int)(this.pictureBox1.Height - 0x16));
        }
        for (num = num7; num > 0; num--)
        {
            graphics.DrawString(Convert.ToString((int)((num7 + 1) - num)), new Font("Arial", 8f), new
SolidBrush(Color.Black), 8f, (float)((this.pictureBox1.Height - (num * num9)) - 30));
            graphics.DrawLine(new Pen(Color.Black, 1f), 0x1c, (this.pictureBox1.Height - (num * num9)) - 0x19,
0x16, (this.pictureBox1.Height - (num * num9)) - 0x19);
        }
        for (num = 0; num < num7; num++)
        {
            graphics.DrawLine(new Pen(Color.Navy, 2f), (int)((this.mas[num, 3] * num8) + 0x19),
(int)((this.pictureBox1.Height - ((num7 - num) * num9)) - 0x19), (int)((this.mas[num, 5] * num8) + 0x19),
(int)((this.pictureBox1.Height - ((num7 - num) * num9)) - 0x19));
            graphics.DrawLine(new Pen(Color.Navy, 2f), (int)((this.mas[num, 3] * num8) + 0x19),
(int)((this.pictureBox1.Height - ((num7 - num) * num9)) - 0x1c), (int)((this.mas[num, 3] * num8) + 0x19),
(int)((this.pictureBox1.Height - ((num7 - num) * num9)) - 0x16));
            graphics.DrawLine(new Pen(Color.Navy, 2f), (int)((this.mas[num, 5] * num8) + 0x19),
(int)((this.pictureBox1.Height - ((num7 - num) * num9)) - 0x1c), (int)((this.mas[num, 5] * num8) + 0x19),
(int)((this.pictureBox1.Height - ((num7 - num) * num9)) - 0x16));
            graphics.DrawLine(new Pen(new HatchBrush(HatchStyle.SmallConfetti, Color.Gray,
Form.ActiveForm.BackColor), 1f), (int)((this.mas[num, 3] * num8) + 0x19), (int)((this.pictureBox1.Height - ((num7
- num) * num9)) - 0x1c), (int)((this.mas[num, 3] * num8) + 0x19), (int)(this.pictureBox1.Height - 0x1c));
            graphics.DrawLine(new Pen(new HatchBrush(HatchStyle.SmallConfetti, Color.Gray,
Form.ActiveForm.BackColor), 1f), (int)((this.mas[num, 5] * num8) + 0x19), (int)((this.pictureBox1.Height - ((num7
- num) * num9)) - 0x1c), (int)((this.mas[num, 5] * num8) + 0x19), (int)(this.pictureBox1.Height - 0x1c));
        }
        graphics.DrawString("Работа", new Font("Arial", 8f), new SolidBrush(Color.Black), (float)5f, (float)5f);
```

```
            graphics.DrawString("Устан. сроки", new Font("Arial", 8f), new SolidBrush(Color.Black),
(float)(this.pictureBox1.Width - 80), (float)(this.pictureBox1.Height - 15));
            this.pictureBox1.BackgroundImage = image;
            graphics.Dispose();
        }

        private void button2_Click(object sender, EventArgs e)
        {
            this.cou = 0x10;
            this.mas[0, 0] = 1;
            this.mas[0, 1] = 2;
            this.mas[0, 2] = 7;
            this.mas[1, 0] = 1;
            this.mas[1, 1] = 3;
            this.mas[1, 2] = 9;
            this.mas[2, 0] = 1;
            this.mas[2, 1] = 4;
            this.mas[2, 2] = 4;
            this.mas[3, 0] = 2;
            this.mas[3, 1] = 4;
            this.mas[3, 2] = 3;
            this.mas[4, 0] = 2;
            this.mas[4, 1] = 6;
            this.mas[4, 2] = 5;
            this.mas[5, 0] = 3;
            this.mas[5, 1] = 5;
            this.mas[5, 2] = 6;
            this.mas[6, 0] = 4;
            this.mas[6, 1] = 5;
            this.mas[6, 2] = 8;
            this.mas[7, 0] = 4;
            this.mas[7, 1] = 6;
            this.mas[7, 2] = 0;
            this.mas[8, 0] = 4;
            this.mas[8, 1] = 7;
            this.mas[8, 2] = 2;
            this.mas[9, 0] = 5;
            this.mas[9, 1] = 7;
            this.mas[9, 2] = 7;
            this.mas[10, 0] = 5;
            this.mas[10, 1] = 10;
            this.mas[10, 2] = 4;
            this.mas[11, 0] = 6;
            this.mas[11, 1] = 7;
            this.mas[11, 2] = 5;
            this.mas[12, 0] = 6;
            this.mas[12, 1] = 8;
            this.mas[12, 2] = 9;
            this.mas[13, 0] = 6;
            this.mas[13, 1] = 9;
            this.mas[13, 2] = 2;
            this.mas[14, 0] = 7;
            this.mas[14, 1] = 10;
            this.mas[14, 2] = 9;
            this.mas[15, 0] = 8;
            this.mas[15, 1] = 9;
            this.mas[15, 2] = 6;
            this.mas[0x10, 0] = 9;
            this.mas[0x10, 1] = 10;
            this.mas[0x10, 2] = 8;
            this.richTextBox1.Text = "";
            this.richTextBox1.AppendText("№\ti-j\tt i-j\n");
```

```
            for (int i = 0; i <= this.cou; i++)
            {
                this.richTextBox1.AppendText(Convert.ToString((int)(i + 1)) + "\t" + Convert.ToString(this.mas[i, 0]) +
"-" + Convert.ToString(this.mas[i, 1]) + "\t" + Convert.ToString(this.mas[i, 2]) + "\n");
            }
            this.button5.Enabled = true;
        }

        private void button3_Click(object sender, EventArgs e) //добавление события i
        {
            this.cou++;
            this.mas[this.cou, 0] = Convert.ToInt32(this.numericUpDown1.Value);
            this.mas[this.cou, 1] = Convert.ToInt32(this.numericUpDown2.Value);
            this.mas[this.cou, 2] = Convert.ToInt32(this.numericUpDown3.Value);
            this.richTextBox1.Text = "";
            this.richTextBox1.AppendText("№\ti-j\tt i-j\n");
            for (int i = 0; i <= this.cou; i++)
            {
                this.richTextBox1.AppendText(Convert.ToString((int)(i + 1)) + "\t" + Convert.ToString(this.mas[i, 0]) +
"-" + Convert.ToString(this.mas[i, 1]) + "\t" + Convert.ToString(this.mas[i, 2]) + "\n");
            }
            this.button5.Enabled = true;
        }

        private void button5_Click(object sender, EventArgs e) //удаление данных
        {
            this.cou--;
            if (this.cou < 0)
            {
                this.button5.Enabled = false;
            }
            this.richTextBox1.Text = "";
            this.richTextBox1.AppendText("№\ti-j\tt i-j\n");
            this.richTextBox2.Text = "";
            this.richTextBox2.AppendText("№\ti-j\tt i-j\tTрн i-j\tTро i-j\tTпо i-j\tTпр i-j\tR i-j\tr i-j\n");
            for (int i = 0; i <= this.cou; i++)
            {
                this.richTextBox1.AppendText(Convert.ToString((int)(i + 1)) + "\t" + Convert.ToString(this.mas[i, 0]) +
"-" + Convert.ToString(this.mas[i, 1]) + "\t" + Convert.ToString(this.mas[i, 2]) + "\n");
            }
        }

        private void button6_Click(object sender, EventArgs e) //убираем событие i-j
        {
            this.numericUpDown1.Value = 1M;
            this.numericUpDown2.Value = 1M;
            this.numericUpDown3.Value = 1M;
            this.cou = -1;
            this.button5.Enabled = false;
            this.richTextBox1.Text = "";
            this.richTextBox1.AppendText("№\ti-j\tt i-j\n");
            this.richTextBox2.Text = "";
            this.richTextBox2.AppendText("№\ti-j\tt i-j\tTрн i-j\tTро i-j\tTпо i-j\tTпр i-j\tR i-j\tr i-j\n");
            this.textBox1.Text = "";
            Bitmap image = new Bitmap(this.pictureBox1.Width, this.pictureBox1.Height);
            Graphics graphics = Graphics.FromImage(image);
            this.pictureBox1.BackgroundImage = image;
            graphics.Dispose();
        }

        protected override void Dispose(bool disposing) //для прорисовки формы, стандартный метод
        {
```

```
            if (disposing && (this.components != null))
            {
                this.components.Dispose();
            }
            base.Dispose(disposing);
        }

        private void Form1_Load(object sender, EventArgs e)
        {
            this.richTextBox1.AppendText("№\ti-j\tt i-j");
            this.richTextBox2.AppendText("№\ti-j\tt i-j\tTрн i-j\tTро i-j\tTпо i-j\tTпр i-j\tR i-j\tr i-j\n");
            this.cou = -1;
        }


        private void numericUpDown1_ValueChanged(object sender, EventArgs e)
        {
            this.numericUpDown2.Minimum = this.numericUpDown1.Value;
        }

        private void tabControl1_Selected(object sender, TabControlEventArgs e)
        {
        }

        private void tabPage2_Resize(object sender, EventArgs e)
        {
            if (this.cou > -1)
            {
                this.button1_Click(new object(), new EventArgs());
            }
        }
    }
}
```

Пример №2

Задача о максимальном потоке является классической и имеет множество применений. Дан взвешенный ориентированный граф с неотрицательными весами (пропускными способностями). Выделены две вершины: исток S и сток T такие, что любая другая вершина лежит на пути из S в T. Потоком назовем функцию F: V x V с такими свойствами

1. Ограничение пропускной способности. Поток по ребру не может быть больше его (ребра) пропускной способности.

2.Антисимметричность. Для каждого ребра (u, v): F(u, v) = -F(v, u).

3. Сохранение потока. Для каждой вершины (кроме S и T), количество входящего потока (отрицательного) равен количеству исходящего потока (положительного). То есть, алгебраическая сумма потоков для каждой вершины (кроме S и T) равна нулю.

Программа «NetworkFlow» является программой реализующей алгоритм поиска наиболее выгодной модели управления запасами для поставленной задачи. Основное окно программы представлено на рисунке 16.

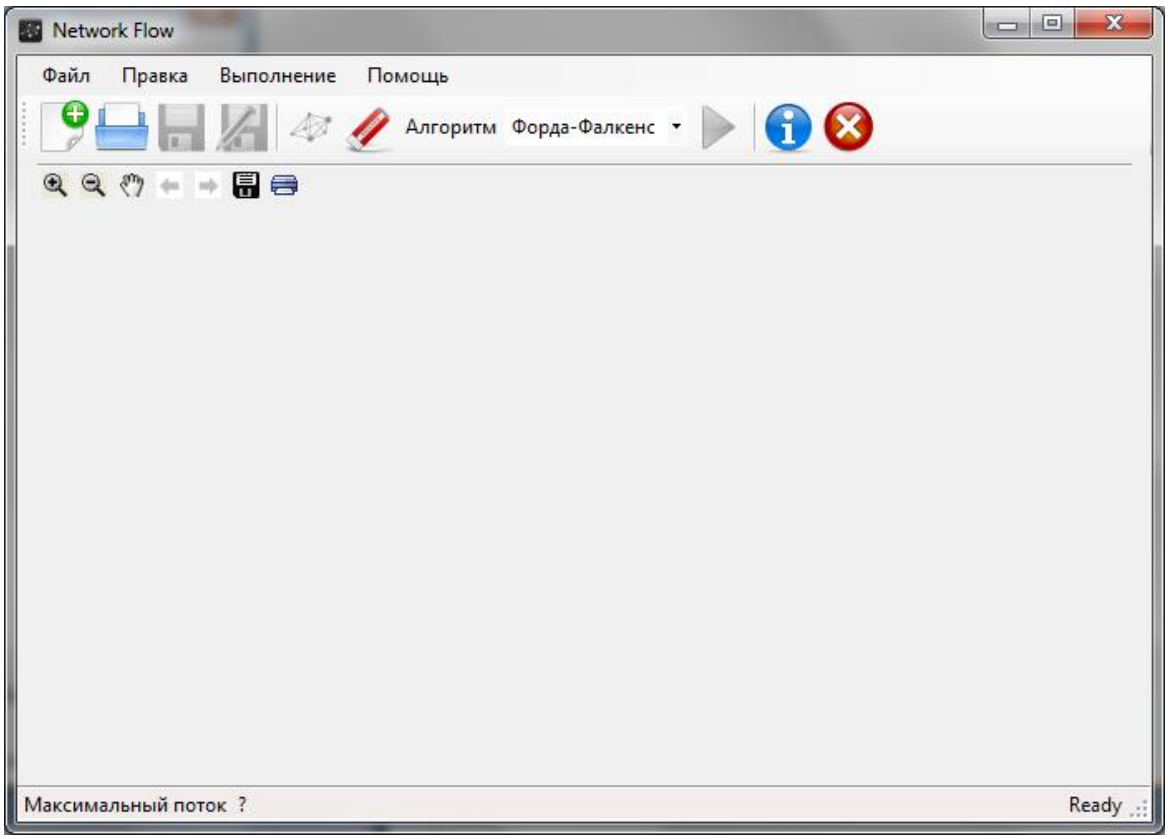


Рисунок 16 – Окно «NetworkFlow»

Чтобы начать работу с программой пользователь должен грузить схему сети (рисунок 16), либо создать её вручную, пользуясь графическим интерфейсом и выбрать один алгоритмов решения задачи

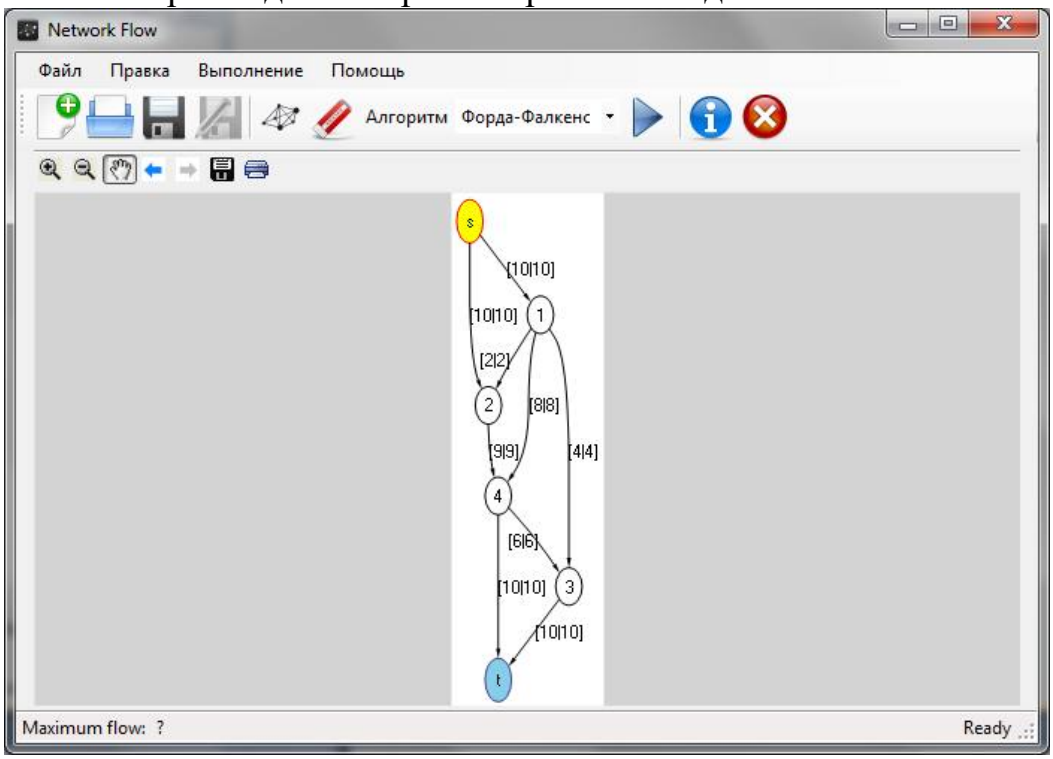


Рисунок 17 – Окно «NetworkFlow»графическое отображение схемы

Подобная реализация является эталонной в плане выполнения лабораторной работы и приводится с целью ориентации студентов в плане реализации интерфейсов решаемых задач. Листинг программы не приводится в связи с тем, что должен реализовываться студентами самостоятельно.

Задачи для самостоятельного решения

В лабораторной работе необходимо выполнить решения комплекса задач. В отчете по каждой задаче должны быть представлены математическое решение и программная реализация.

Задание №1. Во входном файле `input.txt` задан неориентированный взвешенный граф в виде списка ребер и вершин. Построить минимальное остовное дерево.

Задание №2. В входных файлах `заданы три графа, для которых решить` соответственно задачу нахождения кратчайшего пути, задачу о максимальном потоке и задачу нахождение потока наименьшей стоимости в общем виде.

Задание №3

Решить задачу согласно варианту.

1. Граф, заданный матрицей смежности, проверить на связность; в случае несвязности вывести подмножества вершин каждого связного подграфа.

2. Заданный орграф проверить на наличие циклов и при их наличии вывести каждый цикл в виде последовательности вершин циклического пути.

3. В орграфе без циклов, заданном матрицей смежности с весами (длинами) дуг, найти критический путь, то есть путь наибольшей длины, идущий из первой вершины до последней.

4. В орграфе найти все стоки, то есть вершины, в которые только входят дуги, и истоки, из которых только выходят дуги.

5. В заданном неориентированном графе найти кратчайший путь, связывающий две заданные вершины.

6. Матрицу смежности несвязного графа перенумерацией вершин (перестановкой строк и столбцов) превратить в блочную, состоящую из нескольких подматриц, расположенных на главной диагонали. При этом каждая подматрица будет задавать связный подграф.

7. Известно, что плоская фигура может быть обведена за один прием, «не отрывая карандаша от бумаги», если она содержит две или ни одной точки, в которой сходится нечетное число линий. Фигура задана множеством пар номеров вершин <i,j>, соединенных линиями. Найти путь обхода фигуры или показать, что его не существует.

8. Массив *K(n)* содержит значения (номиналы) денежных знаков (купюр и монет) некоторой валютной системы; *L(n)* — количество знаков каждого
достоинства в кассе. Массив *S(m)* — ведомость выдачи зарплаты; известно,

что $\sum_{i=1}^{m} s_i \le \sum_{j=1}^{n} k_j l_j$, то есть касса платежеспособна. Реализовать выдачу зарплаты, то есть найти количество знаков каждого достоинства для каждого работника или показать, что без сдачи это сделать невозможно.

9. Из n предметов, обладающих каждый весом v_i и стоимостью p_i, i=1, 2,..., n, выбрать такие, что при суммарном весе не более Vих суммарная стоимость максимальна.

10. Пусть n красных и n синих точек на плоскости заданы своими координатами. Построить n отрезков с разноцветными концами, суммарная длина которых минимальна (каждая точка является концом только одного отрезка).

11. На плоскости своими координатами задано n точек. Построить связный граф с вершинами во всех этих точках так, чтобы суммарная длина его ребер была наименьшей. В массиве $Z(n)$ найти наибольшую по количеству элементов арифметическую прогрессию (элементы прогрессии стоят в массиве в произвольном порядке).

12. Из прямоугольника размером $a \times b$ требуется вырезать возможно большее число прямоугольников размером $c \times d$. Найти оптимальный вариант раскроя (возможно, таких вариантов несколько).

13. Из элементов массива $A(2n)$ сформировать два «почти ортогональных» массива $B(n)$ и C(n), то есть таких, чтобы модуль скалярного произведения их были минимален.

14. Для каждого жителя города задано множество (возможно, пустое) имен его детей; каждый житель города имеет уникальное имя. Жители x и y называются родственниками, если либо x — ребенок y, либо y — ребенок x, либо существует некий z, такой, что x является родственником z, а z — родственником y. Получить все подмножества родственников. Вместо имен можно использовать шифры (номера) жителей.

15. В условиях предыдущей задачи найти жителя, имеющего наибольшее количество потомков (детей, внуков и так далее). В терминах теории графов это значит, что нужно в несвязном графе найти связный подграф с наибольшим числом вершин. Вывести также весь найденный родовой клан.

16. Найти все вершины графа, к которым существует путь заданной длины (не обязательно кратчайший) от его первой вершины.

17. Клеточное поле размером $m \times n$ есть результат игры в крестики-нолики на «бесконечном» (неограниченном) поле. Проверить, является ли конфигурация предвыигрышной для одного из игроков, то есть нельзя ли за один ход достичь победы. Считается, что, например, «крестики» выиграли, если на поле найдется по горизонтали, вертикали или диагонали цепочка, состоящая из пяти крестиков.

18. В заданном орграфе перенумеровать вершины так, чтобы всякая дуга вела от вершины с меньшим номером к вершине с большим номером или показать, что это невозможно.

19. Определить, является ли заданный граф двудольным, то есть можно ли разбить множество его вершин так, чтобы каждое ребро соединяло вершины двух разных подмножеств.

20. Треугольником в графе называется всякая тройка различных и попарно смежных вершин этого графа. Склеиванием треугольника называется замена треугольника одной вершиной с сохранением связности его с остальным графом. Последовательно применяя склеивание, преобразовать данный граф в граф, в котором нет треугольников.

Задание №4.

Задание связано с реализацией методов сетевого планирования. Пусть для некоторого комплекса работ установлены оценки для каждой работы на уровне нормативных продолжительностей и срочного режима, а также даны стоимости. Информация представлена в таблице. В задание необходимо предусмотреть возможность ручного ввода данных.

.

	Нормативный режим		Срочный режим	
	Продолжительность, дни	Стоимость, м/р	Продолжительность, дни	Стоимость, м/р
(1,2)	3	6	2	11
(1,3)	5	8	3	12
(1,4)	4	7	8	9
(2,5)	10	25	8	30
(3,5)	8	20	6	24
(3,6)	15	26	12	30
(4,6)	13	24	10	30
(5,7)	3	15	6	25
(6,7)	4	10	3	15

1. Построить график данного комплекса работ.
2. Рассчитать: четные варианта вычисляют временные характеристики сетевого графика при нормальном режиме работ, критический путь, полные резервы времени; нечетные варианты рассчитывают временные характеристики сетевого графика при срочном режиме работ, критический путь и полные резервы времени.
3. Определить стоимость работ.

Лабораторная работа
«Детерминированные модели динамического программирования»

Цель работы: изучение принципов решения задач с помощью детерминированных моделей динамического программирования и применение данного метода на практике.

Порядок выполнения:

1. Решить математическую модель для своего варианта;
2. Разработать программу решающую поставленную задачу в общем виде;
3. Составить отчет по работе.

Теоритические сведения

Динамическое программирование (ДП) пределяет оптимальное решение n-мерной задачи путем ее декомпозиции на n этапов, каждый из которых представляет подзадачу относительно одной переменной. Вычислительное преимущество такого подхода состойт в том, что мы занимаемся решением одномерных оптимизационных подзадач, вместо большое n-мерной задачи. Фундаментальным принципом ДП, состовляющим основу декомпозиции задачи, является оптимальность. Так как природа каждого этапа решения зависит от конкретной оптимизационной задачи, ДП не предлагает вычислительных алгоритмов непосредственно для каждого этапа. Вычислительные аспекты решения оптимизационных подзадач на каждом этапе проектируются и реализуются по отдельности (что, конечно, не исключает применения единого алгоритма для всех этапов).

Вычисления в ДП выполняются рекуррентно в том смысле, что оптимальное решение одной подзадачи используется в качестве исходных данных для следующей. Решив последнюю подзадачу, мы получим оптимальное решение исходной задачи. Способ выполнения рекуррентных вычислений зависит от того, как производится декомпозиция исходной задачи. В частности, подзадачи обычно связаны между собой некоторыми общими ограничениями. Если осуществляется переход от одной подзадачи к другой, то должны учитываться эти ограничения.

Принцип оптимальности заключается в том, что на каждом этапе оптимальная стратегия определяется независимо от стратегий, использованных на предыдущих этапах.

Вычисления могут производиться последовательно: от начального этапа до последнего. Такая последовательность вычисления известна как

алгоритм прямой прогонки. Так же задача может быть решена с помощью алгоритма обратной прогонки, в соответствии с которым вычисления проводятся от последнего этапа к первому.

Алгоритмы прямой и обратной прогонки приводят к одному и тому же решению. Несмотря на то что алгоритм прямой прогонки представляется более логичным, в специальной литературе, посвященной динамическому программированию, неизменно используется алгоритм обратной прогонки. Причина этого в том, что в общем случае алгоритм обратной прогонки может быть более эффективным с вычислительной точки зрения.

Мы рассмотрим несколько примеров, каждый из которых выбран для демонстрации методов динамического программирования. При рассмотрении каждого примера особое внимание обратите на три основных элемента моделей динамического программирования.

1. Определение этапов.
2. Определение на каждом этапе вариантов решения (альтернатив).
3. Определение состояний на каждом этапе.

Из перечисленных выше элементов понятие состояния, как правило, представляется весьма сложным для восприятия. Рассмотренные примеры последовательно показывают, что определение состояния меняется в зависимости от моделируемой ситуации. При рассмотрении каждого примера полезно ответить на следующие вопросы.

1. Какие соотношения связывают этапы вместе?
2. Какая информация необходима для того, чтобы получить допустимые решения на текущем этапе без повторной проверки решений, принятых на предыдущих этапах?

Задача о загрузке

Задача о загрузке — это задача о рациональной загрузке судна (самолета, автомашины и т.п.), которое имеет ограничения по объему или грузоподъемности. Каждый помещенный на судно груз приносит определенную прибыль. Задача состоит в определении загрузки судна такими грузами, которые приносят наибольшую суммарную прибыль.

Перед тем как представить соотношения динамического программирования, заметим, что рассматриваемая здесь задача известна также как задача о снаряжении, где пилот реактивного самолета должен определить наиболее ценные (необходимые) предметы, которые следует взять на борт самолета, или как задача о рюкзаке, в которой солдат (или турист) должен определить наиболее ценные предметы, подлежащие загрузке в ранец (рюкзак). Кажется, что три упомянутых названия для одной

и той же задачи были выбраны для того, чтобы гарантировать равное представительство военно-морского флота, воздушных сил и армии!

Рекуррентное уравнение процедуры обратной прогонки выводится для общей задачи загрузки судна грузоподъемностью W предметов (грузов) n наименований. Пусть m_i — количество предметов i-го наименования, подлежащих загрузке, r_i — прибыль, которую приносит один загруженный предмет i-го наименования, w_i,— вес одного предмета i-го наименования. Общая задача имеет вид следующей целочисленной задачи линейного программирования.

Максизимизировать $z=r_1m_1+r_2m_2+\ldots+r_nm_n$

При условии, что

$w_1m_1+w_2m_2+\ldots+w_nm_n \square W$

$m_1, m_2, \ldots, m_n \square 0$ и целые.

Три элемента модели динамического программирования определяется следующим образом.

1. Этап i ставится в соответствие предмету i-го наименования, i = 1, 2,..., n.
2. Варианты решения на этапе i описываются количеством m_i, предметов i-го наименования, подлежащих загрузке. Соответствующая прибыль равна r_im_i. Значение m_i заключено в пределах от 0 до $[W/w_i]$, где $[W/w_i]$ — целая часть числа W/w_i.
3. Состояние x_i на этапе i выражает суммарный вес предметов, решения о погрузке которых приняты на этапах i, i + 1,..., n. Это определение отражает тот факт, что ограничение по весу является единственным, которое связывает n этапов вместе.

Пусть $f_i(x_i)$— максимальная суммарная прибыль от этапов i, i + 1,..., n при заданном состоянии x_i, Проще всего рекуррентное уравнение определяется с помощью следующей двухшаговой процедуры.

ШАГ 1

Выразим функцию $f_i(x_i)$ как функцию $f_{i+1}(x_{i+1})$ в виде

$$f_i(x_i) = \max_{\substack{x_i = 0,1,\ldots,W \\ m_i = 0,1,\ldots,[W/w_i]}} \{r_i m_i + f_{i+1}(x_{i+1})\}, i = 1,2,\ldots,n$$

ШАГ 2

Выразим x_{i+1} как функцию x_i для гарантии того, что левая часть последнего уравнения является функцией лишь x_i. По определению x_i - x_{i+1} представляет собой вес, загруженный на этапе i, т.е. x_i - x_{i+1} = w_im_i. Следовательно, рекуррентное уравнение приобретает следующий вид:

$$f_i(x_i) = \max_{\substack{m_i=0,1,\dots,[W/w_i] \\ x_i=0,1,\dots,W}} \{r_i m_i + f_{i+1}(x_i - w_i m_i)\}, i = 1,2,\dots,n$$

Задача планирования рабочей силы

При выполнении некоторых проектов число рабочих, необходимых для реализации какого-либо проекта, регулируется путем их найма и увольнения. Поскольку как наем, так и увольнение рабочих связано с дополнительными затратами, необходимо определить, каким образом должна регулироваться численность рабочих в период реализации проекта.

Предположим, что проект будет выполняться в течение n недель и минимальная потребность в рабочей силе на протяжении i-й недели составит b_i, рабочих. При идеальных условиях хотелось бы на протяжении i-й недели иметь ровно b_i рабочих. Однако в зависимости от стоимостных показателей может быть более выгодным отклонение численности рабочей силы как в одну, так и в другую сторону от минимальных потребностей. Если x_i — количество работающих на протяжении i-й недели, то возможны затраты двух видов: 1) $C_1(x_i - b_i)$— затраты, связанные с необходимостью содержать избыток $x_i - b_i$ рабочей силы и 2) $C_2(x_i - x_{i-1})$ — затраты, связанные с необходимостью дополнительного найма $x_i - x_{i-1}$ рабочих.

Элементы модели динамического программирования определяются следующим образом.

1. Этап i представляется порядковым номером недели i, i = 1, 2, ..., n.
2. Вариантами решения на i-м этапе являются значения x_i — количество работающих на протяжении i-й недели.
3. Состоянием на i-м этапе является x_{i-1} — количество работающих на протяжении (i - 1)-й недели (этапа).

Рекуррентное уравнение динамического программирования представляется в виде

$$f_i(x_{i-1}) = \min_{x_i \geq b_i} \{C_1(x_i - b_i) + C_2(x_i - x_{i-1}) + f_{i+1}(x_i)\}, i = 1,2,\dots,n$$

где $f_{n+1}(x_n) \equiv 0$. Вычисления начинаются с этапа n при $x_n = b_n$ и заканчиваются на этапе 1.

Задача замены оборудования

Чем дольше механизм эксплуатируется, тем выше затраты на его обслуживание и ниже его производительность. Когда срок эксплуатации механизма достигает определенного уровня, может оказаться более выгодной его замена. Задача замены оборудования, таким образом, сводится к определению оптимального срока эксплуатации механизма.

Предположим, что мы занимаемся заменой механизмов на протяжении n лет. В начале каждого года принимается решение либо об эксплуатации механизма еще один год, либо о замене его новым. Обозначим через r(t) и c(t) прибыль от эксплуатации г-летнего механизма на протяжении года и затраты на его обслуживание за этот же период. Далее пусть s(t) — стоимость продажи механизма, который эксплуатировался t лет. Стоимость приобретения нового механизма остается неизменной на протяжении всех лет и равна I.

Элементы модели динамического программирования таковы.

1. Этап i представляется порядковым номером года i, i = 1, 2, ..., n.
2. Вариантами решения на i-м этапе (т.е. для i-го года) являются альтернативы: продолжить эксплуатацию или заменить механизм в начале i-го года.
3. Состоянием на i-м этапе является срок эксплуатации t (возраст) механизма к началу i-го года.

Пусть $f_i(t)$ — максимальная прибыль, получаемая за годы от i до n при условии» что в начале i-го года имеется механизм t-летнего возраста. Рекуррентное уравнение имеет следующий вид:

$$f_i(t) = \max \begin{Bmatrix} r(t) - c(t) + f_{i+1}(t+1), \text{если эксплуатировать механизм} \\ r(0) + s(t) - I - c(0) + f_{i+1}(1), \text{если заменить механизм} \end{Bmatrix}$$

где $f_{n+1} \equiv 0$.

Задача инвестирования

Предположим, что в начале каждого из следующих n лет необходимо сделать инвестиции P_1, P_2, ..., P_n, Вы имеете возможность вложить капитал в два банка: первый банк выплачивает годовой сложный процент r_1, а второй — r_2. Для поощрения депозитов оба банка выплачивают новым инвесторам премии в виде процента от вложенной суммы. Премиальные меняются от года к году, и для i-го года равны q_{i1} и q_{i2} в первом и втором банках соответственно. Они выплачиваются в конце года, на протяжении которого сделан вклад, и могут быть инвестированы в один из двух банков на следующий год. Это значит, что лишь указанные проценты и новые деньги могут быть инвестированы в один из двух банков. Размещенный в банке вклад должен находиться там до конца рассматриваемого периода. Необходимо разработать стратегию инвестиций на следующие n лет.

Элементы модели динамического программирования таковы.

1. Этап i представляется порядковым номером года i, i = 1, 2, ..., n.
2. Вариантами решения на i-м этапе (для i-го года) являются суммы I_i, и $\overline{I_i}$ инвестиций в первый и второй банк соответственно.

3. Состоянием x_i на i-м этапе является сумма денег на начало i-го года, которые могут быть инвестированы.

Заметим, что по определению $\overline{I_\iota} = x_i - I_i$. Следовательно,

$x_1 = P_i$

$x_i = P_i + q_{i-1.1}I_{i-1} + q_{i-1.2}(x_{i-1} - I_{i-1}) =$

$= P_i + (q_{i-1.1} - q_{i-1.2})I_{i-1} + q_{i-1.2}x_{i-1}$

где i = 2,3, ..., n. Сумма денег x_i, которые могут быть инвестированы, включает лишь новые деньги и премиальные проценты за инвестиции, сделанные на протяжении (i - 1)-го года.

Пусть $f_i(x_i)$— оптимальная сумма инвестиций для интервала от i-го до n-го года при условии, что в начале i-го года имеется денежная сумма x_i. Далее обозначим через s_i, накопленную сумму к концу i-го года при условии, что I_i, и (x_i-I_i) — объемы инвестиций на протяжении i-го года в первый и второй банк соответственно. Обозначая α_i = (1 + r_i), i = 1,2, мы можем сформулировать задачу в следующем виде.

Максимизировать z = $s_1 + s_2 + ... + s_n$, где

$s_i = I_i\alpha_1^{n+1-i} + (x_i - I_i)\alpha_2^{n+1-i} = \left(\alpha_1^{n+1-i} - \alpha_2^{n+1-i}\right)I_i + \alpha_1^{n+1-i}x_i,$

$i = 1,2, \dots, n-1$

$s_n = (\alpha_1 + q_{n1} - \alpha_2 - q_{n2})I_n + (\alpha_2 + q_{n2})x_n$

Поскольку премиальные за n-й год являются частью накопленной денежной суммы от инвестиций, в выражения для s_n добавлены q_{n1} и q_{n2}.

Итак, в данном случае рекуррентное уравнение для обратной прогонки в алгоритме динамического программирования имеет вид:

$$f_1(x_1) = \max_{0<Ii\leq xi}\{s_1 + f_{i+1}(x_{i+1})\}\ , i = 1,2, \dots, n-1$$

где x_i выражается через x_i в соответствии с приведенной выше формулой, а $f_{n+1}(x_{n+1}) \equiv 0$.

Пример построения математической модели

В качестве примера возьмем задачу о кратчайшем пути.

Предположим, необходимо выбрать кратчайший путь между двумя городами. Сеть дорог, показанная на рис. 1, представляет возможные маршруты между исходным городом, находящимся в узле 1, и конечным пунктом, который находится в узле 7. Маршруты проходят через промежуточные города, обозначенные на сети узлами с номерами 2-6,

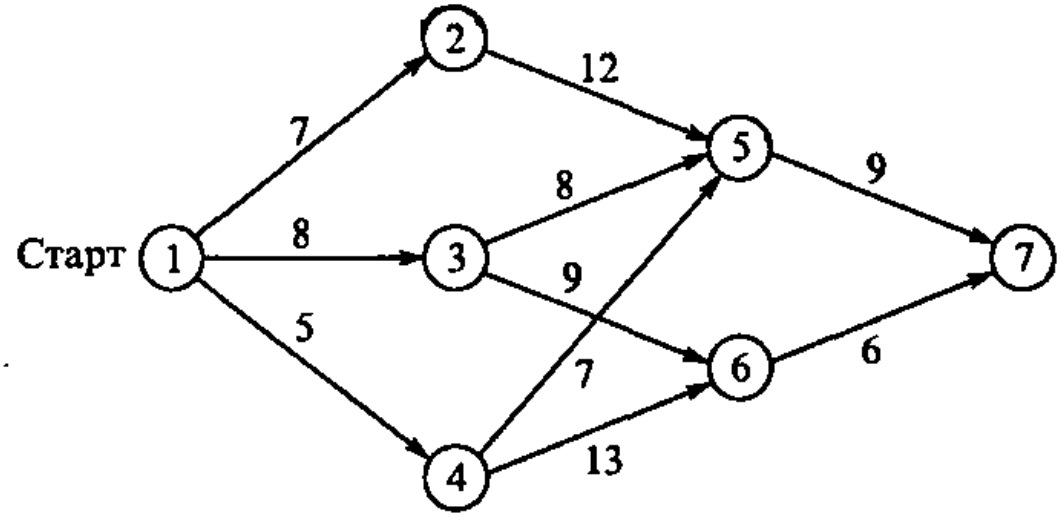


Рисунок 1 –Сеть дорог

Мы можем решить эту задачу посредством полного перебора всех маршрутов между узлами 1 и 7 (имеется пять таких маршрутов). Однако в большой сети полный перебор является неэффективным с вычислительной точки зрения.

Чтобы решить эту задачу с помощью методов динамического программирования, сначала разделим ее на этапы. Вертикальные пунктирные линии на рис.2 очерчивают три этапа задачи. Далее выполняются вычисления для каждого этапа в отдельности.

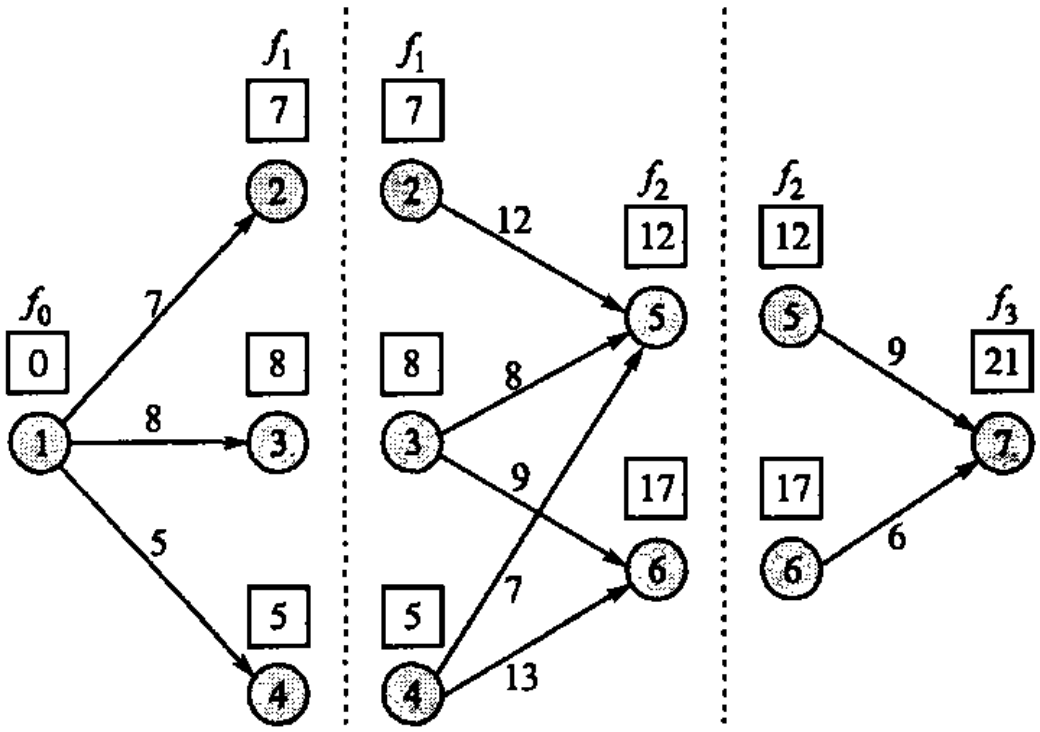


Рисунок 2 –Декомпозиция задачи на 3 этапа

Общая задача состоит в вычислении кратчайших (постепенно накапливаемых) расстояний ко всем вершинам этапа с последующим использованием этих расстояний в качестве исходных данных для следующего этапа. Рассматривая узлы, относящиеся к первому этапу, замечаем, что каждый из узлов 2, 3 и 4 связан с начальным узлом 1 единственной дугой (рис.2). Следовательно, для первого этапа имеем следующее.

Этап 1. Итоговые результаты.

Кратчайший путь к узлу 2 равен 7 миль (из узла 1).

Кратчайший путь к узлу 3 равен 8 миль (из узла 1).

Кратчайший путь к узлу 4 равен 5 миль (из узла 1).

Далее переходим ко второму этапу для вычисления кратчайших (накопленных) расстояний к узлам 5 и 6. Рассматривая узел 5 первым, из рис.2 замечаем, что есть три возможных маршрута, по которым можно достичь узла 5, а именно (2, 5), (3, 5) и (4, 5). Эта информация вместе с кратчайшими расстояниями к узлам 2, 3, и 4 определяет кратчайшее (накопленное) расстояние к узлу 5 следующим образом.

$$\begin{pmatrix}\text{Кратчайщий}\\ \text{путь к узлу 5}\end{pmatrix} = \min_{i=2,3,4}\left\{\begin{pmatrix}\text{Кратчайший}\\ \text{путь к узлу } i\end{pmatrix} + \begin{pmatrix}\text{Расстояние от}\\ \text{узла } i \text{ к узлу 5}\end{pmatrix}\right\} =$$

$$= min\begin{Bmatrix}7+12=19\\ 8+8=16\\ 5+7=12\end{Bmatrix} = 12 \text{ (из узла 4)}$$

Аналогично для узла 6 имеем следующее:

$$\begin{pmatrix}\text{Кратчайщий}\\ \text{путь к узлу 6}\end{pmatrix} = \min_{i=2,3,4}\left\{\begin{pmatrix}\text{Кратчайший}\\ \text{путь к узлу } i\end{pmatrix} + \begin{pmatrix}\text{Расстояние от}\\ \text{узла } i \text{ к узлу 6}\end{pmatrix}\right\} =$$

$$= min\begin{Bmatrix}8+9=17\\ 5+13=18\end{Bmatrix} = 17 \text{ (из узла 3)}$$

Этап 2. Итоговые результаты.

Кратчайший путь к узлу 5 равен 12 миль (из узла 4).

Кратчайший путь к узлу 6 равен 17 миль (из узла 3).

Последним шагом является третий этап. Конечный узел 7 можно достичь как из узла 5, так и 6. Используя итоговые результаты этапа 2 и расстояния от узлов 5 и 6 к узлу 7, получаем следующее.

$$\begin{pmatrix}\text{Кратчайщий}\\ \text{путь к узлу 7}\end{pmatrix} = min\begin{Bmatrix}12+9=21\\ 17+6=23\end{Bmatrix} = 21 \text{ (из узла 5)}$$

Этап 3. Итоговые результаты.

Кратчайший путь к узлу 7 равен 21 миле (из узла 5).

Приведенные вычисления показывают, что кратчайшее расстояние между узлами 1 и 7 равно 21 миле. Города, через которые проходит кратчайший маршрут, определяются следующим образом. Из итоговых

результатов третьего этапа следует, что узел 7 связывается с узлом 5. Далее из итоговых результатов второго этапа следует, что узел 4 связывается с узлом 5. Наконец, из итоговых результатов первого этапа следует, что узел 4 связывается с узлом 1. Следовательно, оптимальным маршрутом является последовательность $1 \rightarrow 4 \rightarrow 5 \rightarrow 7$.

Теперь покажем, как рекуррентные вычисления динамического программирования можно выразить математически. Пусть $f_i(x_i)$ — кратчайшее расстояние до узла x_i на этапе i, $d(x_{i-1},x_i)$ — расстояние от узла x_{i-1} до узла x_i. Тогда f_i, вычисляется на основе значений f_{i-1} с помощью следующего рекуррентного уравнения.

$$f_i(x_i) = \min_{\text{все допустимые маршруты}} \{d(x_{i-1}, x_i) + f_{i+1}(x_{i-1})\}, i = 1,2,3$$

При i = 1 полагаем $f_0(x_0) \equiv 0$. Это уравнение показывает, что кратчайшие расстояния $f_i(x_i)$ на этапе i должны быть выражены как функции следующего узла x_i. В терминологии динамического программирования x_i именуется состоянием системы на этапе i.

В действительности состояние системы на этапе i — это информация, связывающая этапы между собой, при этом оптимальные решения для оставшихся этапов могут приниматься без повторной проверки того, как были получены решения на предыдущих этапах. Такое определение состояния системы позволяет рассматривать каждый этап отдельно и гарантирует, что решение является допустимым на каждом этапе.

Определение состояния системы приводит к следующему унифицированному положению.

Рекуррентное уравнение для алгоритма обратной прогонки вимеет вид:

$$f_i(x_i) = \min_{\text{все допустимые маршруты}} \{d(x_i, x_{i+1}) + f_{i+1}(x_{i+1})\}, i = 1,2,3$$

где $f_4x_4 \equiv 0$ для $x_4=7$. Соответствующей последовательность вычислений будет $f_3 \rightarrow f_2 \rightarrow f_1$.

Этап 3. Поскольку узел 7 ($x_4 = 7$) связан с узлами 5 и 6 ($x_3=5$ и 6) только одним маршрутом, альтернативы для выбора отсутствуют, а результаты третьего этапа можно подытожить следующим образом.

	$d(x_3,x_4)$	Оптимально решение	
x_3	$x_4=7$	$f_3(x_3)$	x'_4
5	9	9	7
6	6	6	7

Этап 2. Так как маршрута (2, 6) не существует, соответствующая альтернатива не рассматривается. Используя значения $f_3(x_3)$, полученные на

третьем этапе, мы можем сравнить допустимые альтернативные решения, как показано в следующей таблице.

x_2	$d(x_2,x_3)+f_3(x_3)$		Оптимально решение	
	$x_3=5$	$x_3=6$	$f_3(x_3)$	x'_4
2	12+9=21	-	21	5
3	8+9=17	9+6=15	15	6
4	7+9=16	13+6=19	16	5

Оптимальное решение второго этапа означает следующее. Если вы находитесь в узле (городе) 2 или 4, кратчайший путь к узлу 7 проходит через узел 5, а если в узле 3 — через узел 6.

Этап 1. Из узла 1 начинаются три альтернативных маршрута: (1, 2), (1, 3) и (1, 4).

Оптимальное решение на первом этапе показывает, что кратчайший путь проходит через город 4. Далее из оптимального решения на втором этапе следует, что из города 4 необходимо двигаться в город 5. Наконец, из оптимального решения на третьем этапе следует, что город 5 связан с городом 7. Следовательно, полным маршрутом, имеющим кратчайшую длину, является 1 → 4 → 5 → 7, и его длина равна 21 миле.

Пример разработанной программы, решающую поставленную задачу в общем виде

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

namespace Лабораторная_9
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }


        private void button1_Click(object sender, EventArgs e)
        {
            int x1=0;
            int x2=0;
            int x3=0;
            int name1=0;
            int name2=0;
            int answ=0;
            int x1_2 = Convert.ToInt32(x12.Text);
```

```
int x1_3 = Convert.ToInt32(x13.Text);
int x1_4 = Convert.ToInt32(x14.Text);
int x2_5 = Convert.ToInt32(x25.Text);
int x3_5 = Convert.ToInt32(x35.Text);
int x4_5 = Convert.ToInt32(x45.Text);
int x3_6 = Convert.ToInt32(x36.Text);
int x4_6 = Convert.ToInt32(x46.Text);
int x5_7 = Convert.ToInt32(x57.Text);
int x6_7 = Convert.ToInt32(x67.Text);


if (x1_2 < x1_3)
{
    x1 = x1_2;
    name1 = 2;
}
else
{
    x1 = x1_3;
    name1 = 3;
}
if (x1_4 < x1)
{
    x1 = x1_4;
    name1 = 4;
}

switch (name1)
{
    case 2:
        x2 = x2_5 + x5_7;
        name2 = 5;
        break;
    case 3:
        if ((x3_5 + x5_7) < (x3_6 + x6_7))
        {
            x2 = x3_5;
            name2 = 5;
        }
        else
        {
            x2 = x3_6;
            name2 = 6;
        }
        break;
    case 4:
        if ((x4_5 + x5_7) < (x4_6 + x6_7))
        {
            x2 = x4_5;
            name2 = 5;
        }
        else
        {
            x2 = x4_6;
            name2 = 6;
        }
        break;
    default:
        break;
}

if (x5_7 > x6_7)
    x3 = x5_7;
else
```

```
                x3 = x6_7;
            answ = x1 + x2 + x3;
            label6.Text = name1.ToString();
            label7.Text = name2.ToString();
            label9.Text = answ.ToString();
        }
    }
}
```

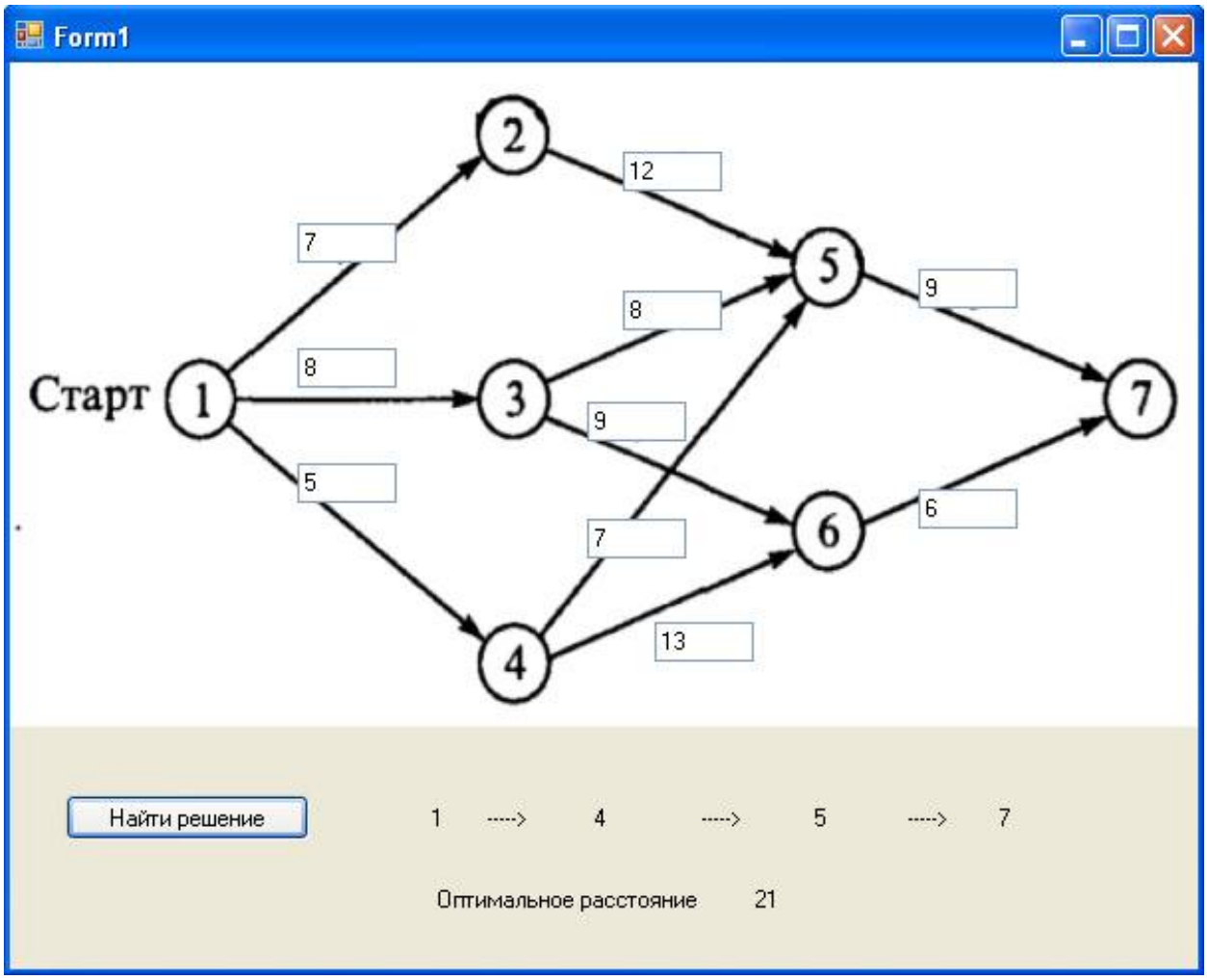


Рисунок 18- Пример работы программы

Задачи для самостоятельного решения

Задача 1

Предположим, вы хотите инвестировать 5000 долл. сейчас и 4000 долл. в начале 2-го года, 3000 долл. в начале 3-го года и 2000 долл. в начале 4-го. Первый банк выплачивает годовой сложный процент 8% и премиальные на протяжении следующих четырех лет в размере 1,8, 1,7, 2,1 и 2,5% соответственно. Годовой сложный процент, предлагаемый вторым банком, на 0,2% ниже, чем предлагает первый банк, но его премиальные на 0,5% выше. Задача состоит в максимизации накопленного капитала к концу четвертого года.

Задача 2

Некий инвестор с начальным капиталом в 10 000 долл. должен решить в конце каждого года, сколько денег истратить и сколько инвестировать. Каждый инвестированный доллар возвращает α = 1,09 долл. в конце года. Истраченные y долларов на протяжении каждого года приносят удовлетворение, определяемое количественно как эквивалент получения $g(y)= \sqrt{y}$ долларов. Решите задачу с помощью методов динамического программирования для периода n = 5 лет.

Задача 3

Фермер имеет k овец. В конце каждого года он принимает решение, сколько овец продать и сколько оставить. Прибыль от продажи одной овцы в i-й год равна p_i. Количество овец в конце i-го года удваивается к концу (i + 1)-го года. Фермер планирует в конце n-го года полностью продать овец.

Решите задачу при следующих данных: n = 3 года, k = 2 овцы, p_1 = 100 долл., p_2 - 130 долл., p_3= 120 долл..

Задача 4

Компания планирует определить оптимальную политику замены используемого в настоящее время двкхлетнего механизма на протяжении следующих 4 лет (n = 4), т.е. вплоть до начала пятого года. Приведенная таблица содержит относящиеся к задаче данные. Компания требует обязательной замены механизма, который находится в эксплуатации 6 лет. Стоимость нового механизма равна 100 000 долл.

Возраст t, (года)	Прибыль r(t),(долл)	Стоимость обслуживания c(t), (долл)	Остаточная стоимость s(t), (долл)
0	20 000	200	-
1	19 000	600	80 000
2	18 500	1200	60 000
3	17 200	1500	50 000
4	15 500	1700	30 000
5	14 000	1800	10 000
6	12 200	2200	5 000

Задача 5

Найдите оптимальное решение задачи 4, при условии, что в начале года имеется механизм, находящийся в эксплуатации 1 год.

Задача 6

Найдите оптимальное решение задачи 4, при условии, что в начале года куплен новый механизм.

Задача 7

Группа ферм владеет трактором двухлетней давности и планирует разработать стратегию его замены на следующие пять лет. Трактор должен эксплуатироваться не менее двух и не более пяти лет. В настоящее время новый трактор стоит 40 000 долларов, и эта цена за год увеличивается на 10%. Текущая годичная стоимость эксплуатации трактора составляет 1300 долларов и, как ожидается, будет увеличиваться на 10% в год.

Определите оптимальную стратегию замены трактора на следующие пять лет.

Задача 8

Рассмотрим задачу замены оборудования на протяжении n лет. Цена новой единицы оборудования равна c долларов, а стоимость продажи после t лет эксплуатации равна $s(t) = n - t$ при $n > t$ и нулю — в противном случае. Годичная прибыль от эксплуатации является функцией возраста оборудования t и равна $r(t) = n^2 - t^2$ при $n > t$ и нулю — в противном случае.

Определите оптимальную стратегию замены оборудования двухгодичной давности при c = 10 000 долл., считая, что n = 5.

Задача 9

Решите задачу из предыдущего упражнения, предполагая, что возраст оборудования составляет 1 год и n = 4, c = 6000 долл., $r(t) = n/(n + 1)$.

Задача 10

Строительный подрядчик оценивает минимальные потребности в рабочей силе на каждую из последующих пяти недель следующим образом: 6, 5, 3, 6 и 8 рабочих соответственно. Содержание избытка рабочей силы обходится подрядчику в 300 долл. за одного рабочего в неделю, а наем рабочей силы на протяжении одной недели обходится в 400 долл. плюс 200 долл. за одного рабочего в неделю.

Задача 11

Решите задачу из задачи 10, предполагая, что подрядчик оценивает минимальные потребности в рабочей силе на каждую из последующих пяти недель следующим образом: 8, 4, 7, 8 и 2 рабочих соответственно.

Задача 12

Решите задачу из задачи 10, предполагая, что каждому уволенному рабочему выплачивается выходное пособие в размере 100 долл. Найдите оптимальное решение

Задача 13

Туристическое агентство организовывает недельные поездки в Египет. В соответствии с договором на ближайшие четыре недели агентство должно обеспечить туристические группы арендными автомобилями в количестве семь, четыре, семь и восемь штук соответственно. Агентство заключает договор с местным дилером по прокату автомобилей. Дилер назначает арендную плату за один автомобиль 220 долл. в неделю плюс 500 долл. за любую арендную сделку. Агентство, однако, может не возвращать арендованные автомобили в конце Недели, и в этом случае оно должно будет платить только арендную плату в 220 долл. Каково оптимальное решение проблемы, связанной с арендой автомобилей?

Задача 14

Компания на следующие четыре года заключила контракт на поставку авиационных двигателей, по 4 двигателя в год. Доступные производственные мощности и стоимость производства меняются от года к году. Компания может изготовить пять двигателей за 1-й год, шесть — за 2-й, три — за 3-й и пять — за 4-й. Стоимость производства одного двигателя на протяжении следующих четырех лет равна соответственно 300 000, 330 000, 350 000 и 420 000 долл. В течение года компания может произвести больше двигателей, чем необходимо, Но в этом случае двигатели должны надлежащим образом храниться до их отгрузки потребителю. Стоимость хранения одного двигателя также меняется от года к году и оценивается в 20 000 долл. для первого года, 30 000 долл. — для второго, 40 000 долл. — для третьего и 50 000 — для четвертого. В начале первого года компания имеет один двигатель, готовый к отгрузке. Разработайте оптимальный план Производства двигателей.

Задача 15

Студент должен выбрать 10 факультативных курсов на четырех различных факультетах, причем на каждом факультете должен быть выбран по меньшей мере один курс. Эти курсы распределяются между факультетами таким образом, чтобы максимизировать объем "знаний". Студент оценивает

знания по шкале в сто баллов и приходит к выводам, представленным в следующей таблице. Какие курсы следует выбрать студенту?

Факультет	Номер курса						
	1	2	3	4	5	6	≥7
I	25	50	60	80	100	100	100
II	20	70	90	100	100	100	100
III	40	60	80	100	100	100	100
IV	10	20	30	40	50	60	70

Задача 16

Шериф округа Вашингтон баллотируется на следующий срок. Денежные средства на предвыборную кампанию составляют примерно 10 000 долларов. Хотя комитет по переизбранию хотел бы провести кампанию во всех пяти избирательных участках округа, ограниченность денежных средств предписывает действовать по-другому. Приведенная ниже таблица содержит данные о числе избирателей и денежных средствах, необходимых для проведения успешной кампании по каждому избирательному участку. Каждый участок может либо использовать все предназначенные деньги, либо вовсе их не использовать. Как следует распределить денежные средства?

Участок	Число избирателей	Необходимые средства (долл)
1	3100	3500
2	2600	2500
3	3500	4000
4	2800	3000
5	2400	2000

Задача 17

Решите следующую задачу с помощью метода динамического программирования.

Максимизировать: $z=y_1^2+ y_2^2+\ldots +y_n^2$

При условиях:

$y_1y_2\ldots y_n=c$,

$y_i\geq 0$, i=1,2, …, n.

Задача 18

Решите следующую задачу с помощью метода динамического программирования.

Минимизировать: $z=(y_1+2)^2+y_2\ y_3+(y_4-5)^2$

При условиях:

$y_1+y_2+\ y_3+y_4=c$,

$y_i\geq 0$, i=1,2, …, n.

Задача 19

Решите следующую задачу с помощью метода динамического программирования.

Максимизировать: $z=(y_1+2)^2+y_2y_3+(y_4-5)^2$

При условиях:

$y_1+y_2+\ y_3+y_4\leq 5$,

$y_i\geq 0$ и целые, i=1,2, …, n.

Задача 20

Решите следующую задачу с помощью метода динамического программирования.

Минимизировать: $z=\max\{f(y_1),f(y_2),\ldots,f(y_n)\}$

При условиях:

$y_1+y_2+\ y_3+y_4=c$,

$y_i\geq 0$, i=1,2, …, n.

Найдите решение задачи при условии, что n=3, c=10, $f(y_1)=\ y_1+5$, $f(y_2)=5\ y_2+3$, $f(y_3)=\ y_3-2$.

Лабораторная работа
«Параметрическое линейное программирование»

Цель работы: изучение принципов решения математической модели для параметрического линейного программирования и применение данного метода на практике.

Порядок выполнения:

1. Решить математическую модель для своего варианта;
2. Разработать программу решающую поставленную задачу в общем виде;
3. Составить отчет по работе.

Теоритические сведения

Параметрическое программирование представляет собой один из разделов математического программирования, изучающий задачи, в которых целевая функция или ограничения зависят от одного или нескольких параметров.

Необходимость рассмотрения подобных задач обусловлена различными причинами. Одной из основных является та, что исходные данные для численного решения любой реальной задачи оптимизации в большинстве случаев определяются приближенно или могут изменяться под влиянием каких-то факторов, что может существенно сказаться на оптимальности выбираемой программы (плана) действий. Соответственно, разумно указывать не конкретные данные, а диапазон возможного изменения данных, чтобы в результате решения иметь наилучшие планы для любого варианта исходных данных.

С математической точки зрения параметрическое программирование выступает как одно из средств анализа чувствительности решения к вариации исходных данных, оценки устойчивости решения.

Заметим, что существуют различные подходы к подобному анализу (например, на основе постановки двойственной задачи). Здесь мы, не ссылаясь на двойственные оценки, рассмотрим самые простейшие варианты решения для самых простейших параметрических программ.

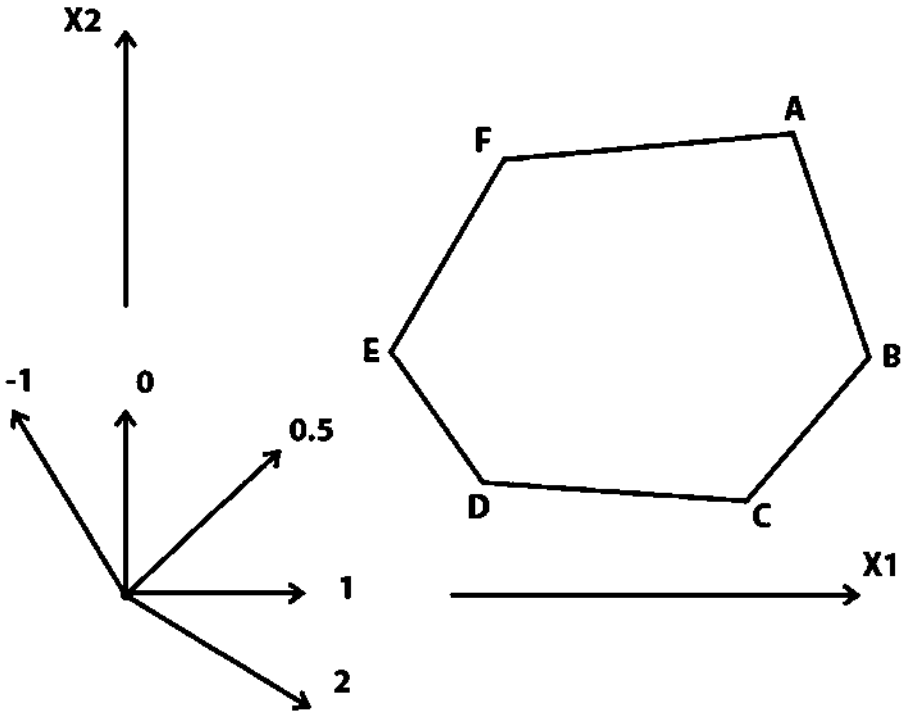
Рассмотрим задачу параметрического линейного программирования, в которой только коэффициенты целевой функции линейно зависят от некоторого единственного параметра λ (времени, температуры и т. п.):

Отыскать максимум (или минимум) функции:

$$\mathrm{L}(\mathrm{X},\lambda) = \sum_{j=1}^{n} (C_j + D_j\lambda)X_j$$

при условиях

$$\sum_{j=1}^{n} A_j X_j = B, \qquad X_j \geq 0, \qquad \lambda_1 \leq \lambda \leq \lambda_2$$



Если обратиться к геометрической интерпретации задачи, то можно заметить, что вектор-градиент линейной формы определяется её параметром. Например, для целевой функции $L(X, \lambda) = \lambda X_1 + (1-\lambda)X_2$ при различных значениях параметра λ градиент определяет различные направления роста функции.

Нетрудно видеть, что, если при некотором значении параметра максимум достигается в вершине А, то небольшая вариация этого значения несколько изменит направление градиента, но не изменит положение точки максимума. Отсюда напрашивается вывод, что некоторый план, оптимальный при $\lambda = \lambda_0$ оптимален и в окрестности λ_0, т.е. при $\alpha \leq \lambda \leq \beta$ где $\lambda_0 \in [\alpha, \beta]$.

Можно заметить, что при градиенте, ставшем перпендикулярным некоторой стороне многоугольника планов, имеем два разных оптимальных опорных плана с одним и тем же значением линейной формы, откуда можно утверждать непрерывность экстремума линейной формы по λ

В случае неограниченности множества планов можно утверждать, что если линейная форма не ограничена при $\lambda = \lambda 0$, то она не ограничена при всех λ, больших или меньших $\lambda 0$.

Алгоритм для решения задач параметрического линейного программирования в случае зависимости от параметра коэффициентов целевой функции незначительно отличается от обычного симплексного метода (примеры решения подобных задач приведены ниже).

В случае зависимости от параметра компонент вектора правых частей ограничений, т. е. решения задачи поиска экстремума функции

$$L(X) = \sum_{j=1}^{n} C_j X_j$$

при условиях

$$\sum_{j=1}^{n} A_j X_j = B + \lambda D, \qquad X_j \geq 0, \qquad j = 1 \ldots n; \quad \lambda_1 \leq \lambda \leq \lambda_2$$

Во избежание сложностей, связанных с требованием сохранения неотрицательности компонент плана при любых λ (сохранения неотрицательности правой части системы уравнений при всех ее тождественных преобразованиях), достаточно постановить двойственную задачу, воспользоваться вышеупомянутым алгоритмом решения задач параметрического линейного программирования при зависимости от параметра коэффициентов целевой функции и с помощью известных двойственных соотношений находить решение исходной задачи.

Пример построения математической модели

Рассмотрим задачу минимизации

$L(X, \lambda) = \lambda X_1 - \lambda X_2 - \lambda X_3 - \lambda X_4$

при условиях

$3X_1 - 3X_2 - X_3 + X_4 \geq 5$

$2X_1 - 2X_2 + X_3 - X_4 \leq 3$

$X_k \geq 0, k = 1 \ldots 4$

$-\infty < \lambda < \infty$

Как обычно, приводим задачу к канонической форме и с использованием метода искусственного базиса отыскиваем начальный опорный план X_0= (0, 0, 0, 0, 0, 3, 5) с $L(X_0, \lambda) = 5M$.

С баз	Базис 1	План $X_{баз}$	λ	-λ	-1	1	0	0	M
			A_1	A_2	A_3	A_4	A_5	A_6	A_7
M	A_7	**5**	3	-3	-1	**1**	-1	**0**	**1**
0	A_6	**3**	2	-2	1	-1	0	**1**	**0**
Δ_k		5M	3M- λ	-3M+λ	-M+1	**M-1**	- M	0	0

Так как определяющую роль на этом шаге решения играет величина *M*, превышающая все величины задачи, то не обращаем внимания на λ и, обнаружив невыполнение критерия оптимальности для X_0, вводим в базис A_4 вместо A_7 (переходим к следующему опорному плану):

С баз	Базис 2	План $X_{баз}$	λ	-λ	-1	1	0	0
			A_1	A_2	A_3	A_4	A_5	A_6
1	A_4	**5**	3	-3	-1	**1**	-1	**0**
0	A_6	**8**	**5**	-5	0	**0**	-1	**1**
Δ_k		5	**3 - λ**	-3+λ	0	0	- 1	0

Полученный опорный план $X_1 = (0, 0, 0, 5, 0, 8)$ с $L(X_1, \lambda) = 5$ будет оптимальным, если все значения Δ_k неположительны, т. е.

$\Delta_1 \equiv 3 - \lambda - \lambda \leq 0$

$\Delta_2 \equiv -3 + \lambda + \lambda \leq 0$

$\lambda \geq 3$

$\lambda \leq 3$

Решаем систему двух линейных неравенств и обнаруживаем, что найденный план X_1 оптимален при $\lambda = 3$.

Исследуем оставшиеся из заданного диапазона значения λ.

Пусть $\lambda > 3$. Тогда $\Delta_2 > 0$ и вектор A_2 подлежит вводу в базис, но в силу неположительности его компонент приходим к выводу, что при $\lambda > 3$ линейная форма задачи не ограничена снизу.

Пусть $\lambda < 3$. Тогда $\Delta_1 > 0$ и в базис вводится вектор A_1:

C баз	Базис 3	План $X_{баз}$	λ	$-\lambda$	-1	1	0	0
			A_1	A_2	A_3	A_4	A_5	A_6
1	A_4	**1/5**	**0**	0	-1	**1**	-2/5	-3/5
λ	A_1	**8/5**	**1**	-1	0	0	-1/5	1/5
	Δ_k	$(8\lambda+1)/5$	0	0	0	0	**-(λ+2)/5**	$(\lambda-3)/5$

Полученный опорный план является оптимальным, если все значения Δ_k неположительны, т. е.

$\Delta_5 = -(\lambda + 2)/5 \leq 0$

$\Delta_6 = (\lambda - 3)/5 \leq 0$

$\lambda \geq -2$

$\lambda \leq 3$

Решая эту систему линейных неравенств, обнаруживаем, что найденный план $X=(8/5, 0, 0, 1/5)$ с $L(X, \lambda) = (8\lambda + 1)/5$ оптимален при $-2 \leq \lambda \leq 3$.

Пусть $\lambda < -2$. Тогда $\Delta_5 > 0$ и вектор A_5 подлежит вводу в базис; в силу неположительности его компонент приходим к выводу, что при $\lambda < -2$ линейная форма задачи не ограничена снизу.

Таким образом, мы получили решение задачи:

$$L(X) = \begin{cases} \to -\infty, \lambda < -2; \\ \dfrac{8\lambda + 1}{5}, -2 \leq \lambda \leq 3; X_{opt} = (\dfrac{8}{5}, 0, 0, \dfrac{1}{5}) \\ 5, \lambda = 3; X_{opt} = (0, 0, 0, 5) \\ \to -\infty, \lambda > 3. \end{cases}$$

Задачи для самостоятельного решения

Выполнить программное решение задачи параметрического линейного программирования согласно варианту:

Задача 1

Максимизировать
$L(x, \lambda) = (1 - \lambda)x1 - (2 + 3\lambda)x2 + x3$
при условиях
$x1 + 4x2 + x3 = 5$
$x1 - 2x2 + x3 = -1$
$x1, x2, x3 \geq 0$
$-\infty < \lambda < +\infty$

Задача 2

Максимизировать
$L(x, \lambda) = x1 - (5 + \lambda)x2 - (1 + \lambda)x3 + x4$
при условиях
$x1 + 3x2 + 3x3 + x4 = 3$
$2x1 + 3x3 - x4 = 4$
$x1, x2, x3, x4 \geq 0$
$-\infty < \lambda < +\infty$

Задача 3

Максимизировать
$L(x, \lambda) = (1 - \lambda)x1 + (1 + \lambda)x2 - x3$
при условиях
$x1 - x2 - x3 = 4$
$x1 + 15x2 + x3 = 2$
$x1, x2, x3 \geq 0$
$-\infty < \lambda < +\infty$

Задача 4

Минимизировать
$L(x, \lambda) = -x1 - (4 - \lambda)x2 - (1 + 2\lambda)x3$
при условиях
$4x1 + 11x2 + 3x3 = 7$
$x1 + x2 - x3 = 0$
$x1, x2, x3 \geq 0$
$-\infty < \lambda < +\infty$

Задача 5

Максимизировать
$L(x, \lambda) = (1 - \lambda)x1 + (3 - \lambda)x2 + 5x3$
при условиях

$x1 + 2x2 - x3 = 2$
$x1 - x2 = 6$
$x1, x2, x3 \geq 0$
$-\infty < \lambda < +\infty$

Задача 6

Минимизировать
$L(x, \lambda) = (-1 + \lambda)x1 - (4 + \lambda)x2 - (1 + 5\lambda)x3$
при условиях
$x1 - x2 + x3 = 3$
$2x1 - 5x2 - x3 = 0$
$x1, x2, x3 \geq 0$
$-\infty < \lambda < +\infty$

Задача 7

Максимизировать
$L(x, \lambda) = x1 + (2 - 10\lambda)x2 - (1 + \lambda)x3$
при условиях
$x1 + 7x2 + 9x3 = 8$
$x1 + 3x2 + 5x3 = 4$
$x1, x2, x3 \geq 0$
$-\infty < \lambda < +\infty$

Задача 8

Максимизировать
$L(x, \lambda) = (1 - \lambda)x1 + (8 - 2\lambda)x2 + (10 + \lambda)x3$
при условиях
$x1 - x2 + 4x3 = 2$
$x1 - x2 + 2x3 = 0$
$x1, x2, x3 \geq 0$
$-\infty < \lambda < +\infty$

Задача 9

Минимизировать
$L(x, \lambda) = -(1 + 2\lambda)x1 - (1 - \lambda)x2 - x3$
при условиях
$-x1 + x2 + x3 = 2$
$3x1 - x2 + x3 = 0$
$x1, x2, x3 \geq 0$
$-\infty < \lambda < +\infty$

Задача 10

Максимизировать
$L(x, \lambda) = (1 - \lambda)x1 - 3x2 - (5 + \lambda)x3 - (1 + 2\lambda)x4$

при условиях
$x1 + 4x2 + 4x3 + x4 = 5$
$x1 + 7x2 + 8x3 + 2x4 = 9$
$x1, x2, x3, x4 \geq 0$
$-\infty < \lambda < +\infty$

Задача 11

Минимизировать
$L(x, \lambda) = -(1 + \lambda)x1 - x2 - (3 - 2\lambda)x3 + 2x4$
при условиях
$x1 + 2x2 + 5x3 - x4 = 3$
$3x1 - x2 + x3 - 10x4 = 2$
$x1, x2, x3, x4 \geq 0$
$-\infty < \lambda < +\infty$

Задача 12

Максимизировать
$L(x, \lambda) = (1 - 4\lambda)x1 + (2 + \lambda)x2 + (\lambda - 1)x3 + (1 + 4\lambda)x4$
при условиях
$x1 + x2 - 2x3 + 3x4 = 1$
$2x1 - x2 - x3 + x4 = 2$
$x1, x2, x3, x4 \geq 0$
$-\infty < \lambda < +\infty$

Задача 13

Минимизировать
$L(x, \lambda) = (\lambda - 1)x1 - (1 + 2\lambda)x2 - x3 - x4$
при условиях
$x1 + 3x2 + 7x3 - x4 = 6$
$x1 - x2 - x3 + 3x4 = 2$
$x1, x2, x3, x4 \geq 0$
$-\infty < \lambda < +\infty$

Задача 14

Максимизировать
$L(x, \lambda) = (1 + 2\lambda)x1 + (1 - \lambda)x2 + x3 + x4$
при условиях
$x1 + 3x2 - x3 + 2x4 = 5$
$2x1 - x3 = 1$
$x1, x2, x3, x4 \geq 0$
$-\infty < \lambda < +\infty$

Задача 15

Максимизировать

$L(x) = x1 - 2x2 + x3$
при условиях
$x1 + 4x2 + x3 = 5 + 2\lambda$
$x1 - 2x2 - x3 = 4\lambda - 1$
$x1, x2, x3 \geq 0$
$-\infty < \lambda < +\infty$

Задача 16

Максимизировать
$L(x) = x1 + 7x2 - x3$
при условиях
$x1 - x2 - 2x3 = 2\lambda - 1$
$x1 + 2x2 + 13x3 = 14 - \lambda$
$x1, x2, x3 \geq 0$
$-\infty < \lambda < +\infty$

Задача 17

Максимизировать
$L(x) = x1 + 2x2 - x3$
при условиях
$x1 + 7x2 + 9x3 = 8 + 5\lambda$
$x1 + 3x2 + 5x3 = 4 + \lambda$
$x1, x2, x3, x4 \geq 0$
$-\infty < \lambda < +\infty$

Задача 18

Максимизировать
$L(x) = x1 + 3x2 + 5x3$
при условиях
$x1 + 2x2 - x3 = 2 - \lambda$
$x1 - x2 = 3\lambda$
$x1, x2, x3 \geq 0$
$-\infty < \lambda < +\infty$

Задача 19

Максимизировать
$L(x) = x1 + 4x2 + x3$
при условиях
$x1 - x2 + x3 = 3 + \lambda$
$2x1 - 5x2 - x3 = -5\lambda$
$x1, x2, x3 \geq 0$
$-\infty < \lambda < +\infty$

Задача 20

Максимизировать
$L(x) = x1 + 8x2 + 10x3$
при условиях
$x1 + x2 + 4x3 = 2 – \lambda$
$x1 - x2 + 2x3 = 10\lambda$
$x1, x2, x3 \geq 0$
$-\infty < \lambda < +\infty$

Лабораторная работа
«Теория игр и принятия решений»

Цель работы: изучение принципов решения математической модели для теории игр и принятия решений и применение данного метода на практике.

Порядок выполнения:

1. Решить математическую модель для своего варианта;
2. Разработать программу решающую поставленную задачу в общем виде;
3. Составить отчет по работе.

Теоритические сведения

Классическими задачами исследования операций являются игровые задачи принятия решений в условиях риска и неопределенности.

Неопределенными могут быть как цели операции, условия выполнения операции, так и сознательные действия противников или других лиц, от которых зависит успех операции.

Разработаны специальные математические методы, предназначенные для обоснования решений в условиях риска и неопределенности. В некоторых, наиболее простых случаях эти методы дают возможность фактически найти и выбрать оптимальное решение. В более сложных случаях эти методы доставляют вспомогательный материал, позволяющий глубже разобраться в сложной ситуации и оценить каждое из возможных решений с различных точек зрения, и принять решений с учетом его возможных последствий. Одним из важных условий принятия решений в этом случае является минимизация риска.

При решении ряда практических задач исследования операций (в области экологии, обеспечения безопасности жизнедеятельности и т. д.) приходится анализировать ситуации, в которых сталкиваются две (или более) враждующие стороны, преследующие различные цели, причем результат любого мероприятия каждой из сторон зависит от того, какой образ действий выберет противник. Такие ситуации мы можно отнести к конфликтным ситуациям.

Теория игр является математической теорией конфликтных ситуаций, при помощи которой можно выработать рекомендации по рациональному образу действий участников конфликта. Чтобы сделать возможным

математический анализ ситуации без учета второстепенных факторов, строят упрощенную, схематизированную модель ситуации, которая называется игрой. игра ведется по вполне определенным правилам, под которыми понимается система условий, регламентирующая возможные варианты действий игроков; объем информации каждой стороны о поведении другой; результат игры, к которому приводит каждая данная совокупность ходов.

Результат игры (выигрыш или проигрыш) вообще не всегда имеет количественное выражение, но обычно можно, хотя бы условно, выразить его числовым значением.

Ход — выбор одного из предусмотренных правилами игры действий и его осуществление. Ходы делятся на личные и случайные. Личным ходом называется сознательный выбор игроком одного из возможных вариантов действий и его осуществление. Случайным ходом называется выбор из ряда возможностей, осуществляемый не решением игрока, а каким-либо механизмом случайного выбора (бросание монеты, выбор карты из перетасованной колоды и т. п.). Для каждого случайного хода правила игры определяют распределение вероятностей возможных исходов. Игра может состоять только их личных или только из случайных ходов, или из их комбинации. Следующим основным понятием теории игр является понятие стратегии. Стратеги — это априори принятая игроком система решений (вида «если — то»), которых он придерживается во время ведения игры, которая может быть представлена в виде алгоритма и выполняться автоматически.

Целью теории игр является выработка рекомендаций для разумного поведения игроков в конфликтной ситуации, т. е. определение «оптимальной стратегии» для каждого из них. Стратегия, оптимальная по одному показателю, необязательно будет оптимальной по другим. Сознавая эти ограничения и поэтому не придерживаясь слепо рекомендаций, полученных игровыми методами, можно все же разумно использовать математический аппарат теории игр для выработки, если не в точности оптимальной, то, во всяком случае «приемлемой» стратегии.

Игры можно классифицировать: по количеству игроков, количеству стратегий, характеру взаимодействия игроков, характеру выигрыша, количеству ходов, состоянию информации и т.д..

В зависимости от количества игроков различают игры двух и n игроков. Первые из них наиболее изучены. Игры трех и более игроков менее исследованы из-за возникающих принципиальных трудностей и технических возможностей получения решения.

В зависимости от числа возможных стратегий игры делятся на «конечные» и «бесконечные».

Игра называется конечной, если у каждого игрока имеется только конечное число стратегий, и бесконечной, если хотя бы у одного из игроков имеется бесконечное число стратегий.

По характеру взаимодействия игры делятся на бескоалиционные: игроки не имеют права вступать в соглашения, образовывать коалиции; коалиционные (кооперативные) — могут вступать в коалиции.

В кооперативных играх коалиции заранее определены.

По характеру выигрышей игры делятся на: игры с нулевой суммой (общий капитал всех игроков не меняется, а перераспределяется между игроками; сумма выигрышей всех игроков равна нулю) и игры с ненулевой суммой.

По виду функций выигрыша игры делятся на: матричные, биматричные, непрерывные, выпуклые и др.

Матричная игра — это конечная игра двух игроков с нулевой суммой, в которой задается выигрыш игрока 1 в виде матрицы (строка матрицы соответствует номеру применяемой стратегии игрока 1, столбец — номеру применяемой стратегии игрока на пересечении строки и столбца матрицы находится выигрыш игрока 1, соответствующий применяемым стратегиям).

Для матричных игр доказано, что любая из них имеет решение и оно может быть легко найдено путем сведения игры к задаче линейного программирования.

Биматричная игра — это конечная игра двух игроков с ненулевой суммой, в которой выигрыши каждого игрока задаются матрицами отдельно для соответствующего игрока (в каждой матрице строка соответствует стратегии игрока 1, столбец — стратегии игрока 2, на пересечении строки и столбца в первой матрице находится выигрыш игрока 1, во второй матрице — выигрыш игрока)

Непрерывной считается игра, в которой функция выигрышей каждого игрока является непрерывной. Доказано, что игры этого класса имеют решения, однако не разработано практически приемлемых методов их нахождения.

Если функция выигрышей является выпуклой, то такая игра называется выпуклой. Для них разработаны приемлемые методы решения, состоящие в отыскании чистой оптимальной стратегии (определенного числа) для одного игрока и вероятностей применения чистых оптимальных стратегий другого игрока. Такая задача решается сравнительно легко.

Принятие решений в условиях определенности — метод анализа иерархий

Модели линейного программирования являются примером принятия решений в условиях определенности. Эти модели применимы лишь в тех случаях, когда альтернативные решения можно связать между собой точными линейными функциями. В этом разделе рассматривается иной подход к принятию решений в ситуациях, когда, например, для идей, чувств, эмоций определяются некоторые количественные показатели, обеспечивающие числовую шкалу предпочтений для возможных альтернативных решений. Этот подход известен как метод анализа иерархий.

Перед тем как изложить детали данного метода, рассмотрим пример, демонстрирующий способ, с помощью которого оцениваются различные альтернативные решения.

Общая структура метода анализа иерархий может включать несколько иерархических уровней со своими критериями. Определение весовых коэффициентов. Сложность метода анализа иерархий за ключается в определении относительных весовых коэффициентов для оценки альтернативных решений. Если имеется n критериев на заданном уровне иерархии, соответствующая процедура создает матрицу A размерности n × n, именуемую матрицей парных сравнений, которая отражает суждение лица, принимающего решение, относительно важности разных критериев. Парное сравнение выполняется таким образом, что критерий в строке i (i = 1, 2, ..., n) оценивается относительно каждого из критериев, представленных n столбцами. Обозначим через aij элемент матрицы A, находящийся на пересечении i-й строки и j-го столбца. В соответствии с методом анализа иерархий для описания упомянутых оценок используются целые числа от 1 до 9. При этом aij = 1 означает, что i-й и j-й критерии одинаково важны, aij = 5 отражает мнение, что i-й критерий значительно важнее, чем j-й, а aij = 9 указывает, что i-й критерий чрезвычайно важнее j-го. Другие промежуточные значения между 1 и 9 интерпретируются аналогично. Согласованность таких обозначений обеспечивается следующим условием: если aij = k, то автоматически aji = 1/k. Кроме того, все диагональные элементы aii матрицы A должны быть равны 1, так как они выражают оценку критерия относительно самих себя.

Согласованность матрицы сравнений.

Согласованность означает, что решение будет согласовано с определениями парных сравнений критериев или альтернатив. С математической точки зрения согласованность матрицы A означает, что $a_{ij}a_{jk}=a_{ik}$ для всех i, j и k. Свойство согласованности требует линейной

зависимости столбцов (истрок) матрицыA. В частности, столбцы любой матрицы сравнений размерностью 2×2 являются зависимыми, и, следовательно, такая матрица всегда является согласованной. Не все матрицы сравнений являются согласованными. Действительно, принимая во внимание, что такие матрицы строятся на основе человеческих суждений, можно ожидать некоторую степень несогласованности, и к ней следует относиться терпимо при условии, что она не выходит за определенные ‘‘допустимые’’ рамки.

Чтобы выяснить, является ли уровень согласованности ‘‘допустимым’’, необходимо определить соответствующую количественную меру для матрицы сравнений A.

$$\mathbf{N} = \begin{pmatrix} w_1 & w_1 & \cdots & w_1 \\ w_2 & w_2 & \cdots & w_2 \\ \vdots & \vdots & \vdots & \vdots \\ w_n & w_n & \cdots & w_n \end{pmatrix}.$$

Отсюда следует, что матрица сравнений A может быть получена из матрицы N путем деления элементов i-го столбца на w_i (это процесс, обратный к нахождению матрицы N из A). Итак, получаем следующее.

$$\mathbf{A} = \begin{pmatrix} 1 & \frac{w_1}{w_2} & \cdots & \frac{w_1}{w_n} \\ \frac{w_2}{w_1} & 1 & \cdots & \frac{w_2}{w_n} \\ \vdots & \vdots & \vdots & \vdots \\ \frac{w_n}{w_1} & \frac{w_n}{w_2} & \cdots & 1 \end{pmatrix}.$$

Используя приведенное определение матрицы A, имеем

$$\begin{pmatrix} 1 & \frac{w_1}{w_2} & \cdots & \frac{w_1}{w_n} \\ \frac{w_2}{w_1} & 1 & \cdots & \frac{w_2}{w_n} \\ \vdots & \vdots & \vdots & \vdots \\ \frac{w_n}{w_1} & \frac{w_n}{w_2} & \cdots & 1 \end{pmatrix} \begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{pmatrix} = \begin{pmatrix} nw_1 \\ nw_2 \\ \vdots \\ nw_n \end{pmatrix} = n \begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{pmatrix}.$$

В компактной форме условие согласованности матрицы A формулируется следующим образом. Матрица A будет согласованной тогда и только тогда, когда

Aw = nw,

где w – вектор столбец относительных весов w_i, i = 1, 2, ..., n.

Когда матрица A не является согласованной, относительный вес wi аппроксимируется средним значением n элементов i-й строки нормализованной матрицы N. Обозначив через w вычисленную оценку (среднее значение), можно показать, что

$$A\overline{w} = n_{max}\overline{w}$$

где nmax ≥ n. В этом случае, чем ближе nmax к n, тем более согласованной является матрица сравнения A. В результате в соответствии с методом анализа иерархий вычисляется коэффициент согласованности в виде

$$CR = \frac{CI}{RI}$$

где

$CI = \frac{n_{max}-n}{n-1}$ – коэффициент согласованности матрицы A,

$RI = \frac{1{,}98(n-2)}{n}$ – стохастический коэффициент согласованности матрицы A.

Стохастический коэффициент согласованности RI определяется эмпирическим путем как среднее значение коэффициента CI для большой выборки генерированных случайным образом матриц сравнения A.

Коэффициент согласованности CR используется для проверки согласованности матрицы сравнения A следующим образом. Если CR ≤ 0,1, уровень несогласованности является приемлемым. В противном случае уровень несогласованности матрицы сравнения A является высоким, и лицу, принимающему решение, рекомендуется проверить элементы парного сравнения a_{ij} матрицы A в целях получения более согласованной матрицы.

Значение nmax вычисляется на основе матричного уравнения $A\overline{w} = n_{max}\overline{w}$, при этом нетрудно заметить, что i-е уравнение этой системы имеет вид:

$$\sum_{j=1}^{n} a_{ij}\overline{w_j} = n_{max}\overline{w_i}, i = 1,2,\dots,n$$

Поскольку $\sum_{i=1}^{n}\overline{w_i} = 1$, легко проверить, что

$$\sum_{i=1}^{n}\left(\sum_{j=1}^{n} a_{ij}\overline{w_j}\right) = n_{max}\sum_{i=1}^{n}\overline{w_i} = n_{max}$$

Это значит, что величину nmax можно определить путем вычисления вектор-столбца $A\overline{w}$ с последующим суммированием его элементов.

Принятие решений в условиях риска

Если решение принимается в условиях риска, то стоимости альтернативных решений обычно описываются вероятностными распределениями. По этой причине принимаемое решение основывается на использовании критерия ожидаемого значения, в соответствии с которым альтернативные решения сравниваются с точки зрения максимизации ожидаемой прибыли или минимизации ожидаемых затрат. Такой подход имеет свои недостатки, которые не позволяют использовать его в некоторых ситуациях. Для них разработаны модификации упомянутого критерия. В этой главе рассматриваются часто используемые подходы к принятию решений в условиях риска.

Критерий ожидаемого значения

Критерий ожидаемого значения сводится либо к максимизации ожидаемой (средней) прибыли, либо к минимизации ожидаемых затрат. В данном случае предполагается, что прибыль (затраты), связанная с каждым альтернативным решением, является случайной величиной.

В общем случае задача принятия решений может включать n состояний природы и m альтернатив. Если p_j – вероятность j-го состояния природы, а a_{ij} – платеж, связанный с принятием решения i при состоянии природы j (i = 1, 2, ..., m, j = 1, 2, ..., n), тогда ожидаемый платеж для решения i вычисляется в виде

$$MV_i = a_{i1}p_1 + a_{i2}p_2 + \cdots + a_{in}p_n, i = 1,2, \dots, n$$

где по определению $p_1 + p_2 + ... + p_n = 1$.

Наилучшим решением будет то, которое соответствует $MV_i' = max_i\{MV_i\}$ или $MV_i' = min_i\{MV_i\}$, в зависимости от того, является ли платеж в задаче доходом (прибылью) или убытком (затратами).

Пример построения математической модели

Предположим, что вы хотите вложить на фондовой бирже 10 000 долл. в акции одной из двух компаний: А или В. Акции компании А являются рискованными, но могут принести 50 % прибыли от суммы инвестиции на протяжении следующего года. Если условия фондовой биржи будут неблагоприятны, сумма инвестиции может обесцениться на 20 %. Компания В обеспечивает безопасность инвестиций с 15 % прибыли в условиях повышения котировок на бирже и только 5 % - в условиях понижения котировок. Все аналитические публикации, с которыми можно познакомиться (а они всегда есть в изобилии в конце года), с вероятностью 60 % прогнозируют повышение котировок и с вероятностью 40 % - понижение котировок. В какую компанию следует вложить деньги?

Информация, связанная с принятием решения, суммирована в следующей таблице.

Альтернативные решения	Прибыль а один год от инвестиции 10 000 долл	
	При повышении котировок	При понижении котировок
Акции компании А	5 000	-2 000
Акции компании В	1 500	500
Вероятность события	0,6	0,4

Эта задача может быть представлена в виде дерева решений. На рисунке используется два типа вершин: квадратик представляет ‘решающую’’ вершину, а кружок - ‘‘случайную’’. Таким образом, из вершины 1 (‘‘решающая’’) выходят две ветви, представляющие альтернативы, связанные с покупкой акций компании А или В. Далее две ветви, выходящие из ‘‘случайных’’ вершин 2 и 3, соответствуют случаям повышения и понижения котировок на бирже с вероятностями их появления и соответствующими платежами.

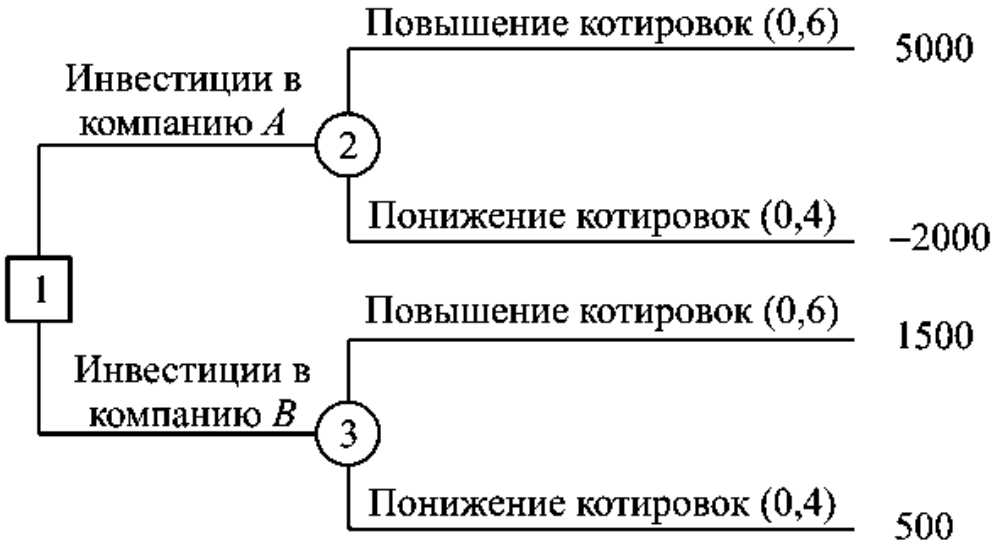


Рисунок 1 - Дерево решений для задачи инвестирования

Исходя из схемы рисунка получаем ожидаемую прибыль за год для каждой из двух альтернатив.

Для акций компании А: 5000 × 0,6 + (-2000) × 0,4 = 2 200 (долл.).

Для акций компании В: 1500 × 0,6 + 500 × 0,4 = 1 100 (долл.).

Вашим решением, основанным на этих вычислениях, является покупка акций компании А.

Пример разработанной программы, решающую поставленную задачу в общем виде

Пример №1

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

namespace Лабораторная_12
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            double A, B = 0;
            A = Convert.ToDouble(P1.Text) * Convert.ToDouble(bet1.Text) +
Convert.ToDouble(P2.Text) * Convert.ToDouble(bet2.Text);
            B = Convert.ToDouble(P3.Text) * Convert.ToDouble(bet3.Text) +
Convert.ToDouble(P4.Text) * Convert.ToDouble(bet4.Text);

            if (A>B)
            {
                label1.Text = "выгодно вложиться в компанию А";
            }
            else
            {
                label1.Text = "выгодно вложиться в компанию В";
            }
        }
    }
}
```

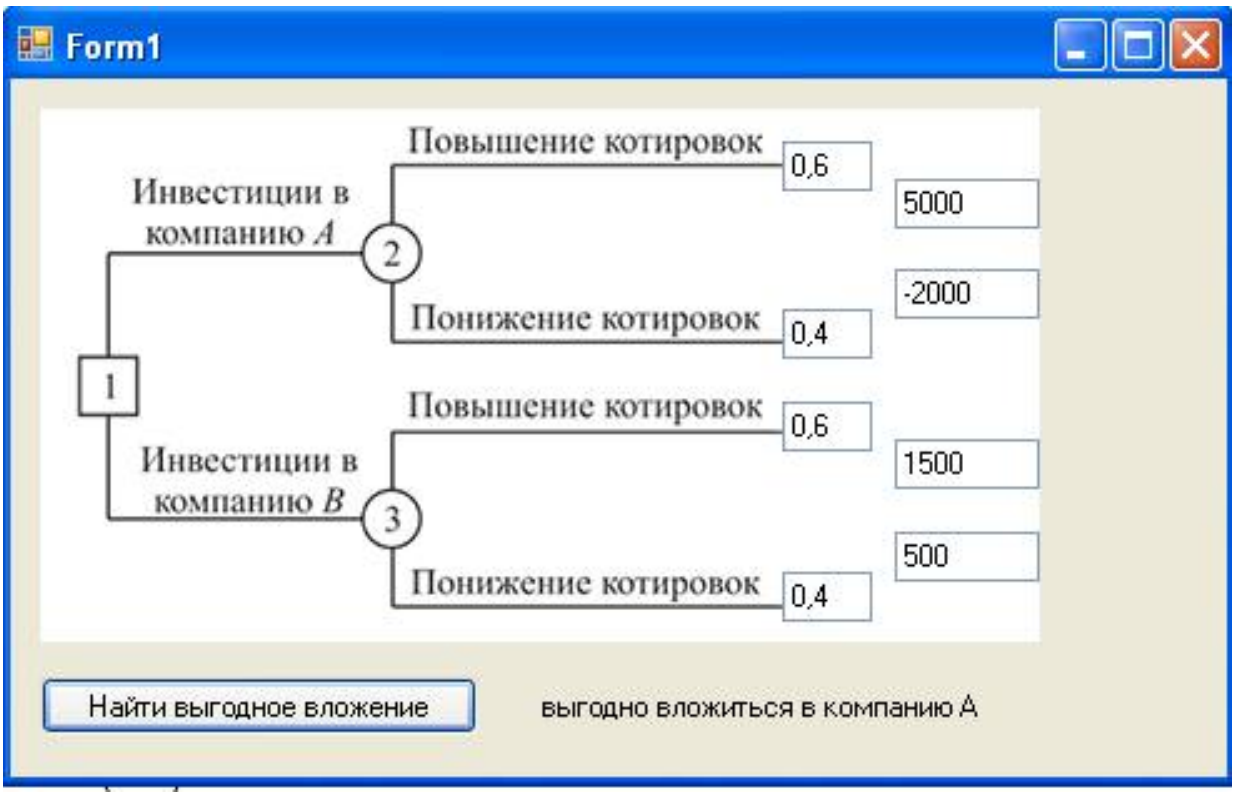


Рисунок 19 – Приложение к примеру №1

Пример №2

Решение еще одной задачи приведено на рисунке 20.

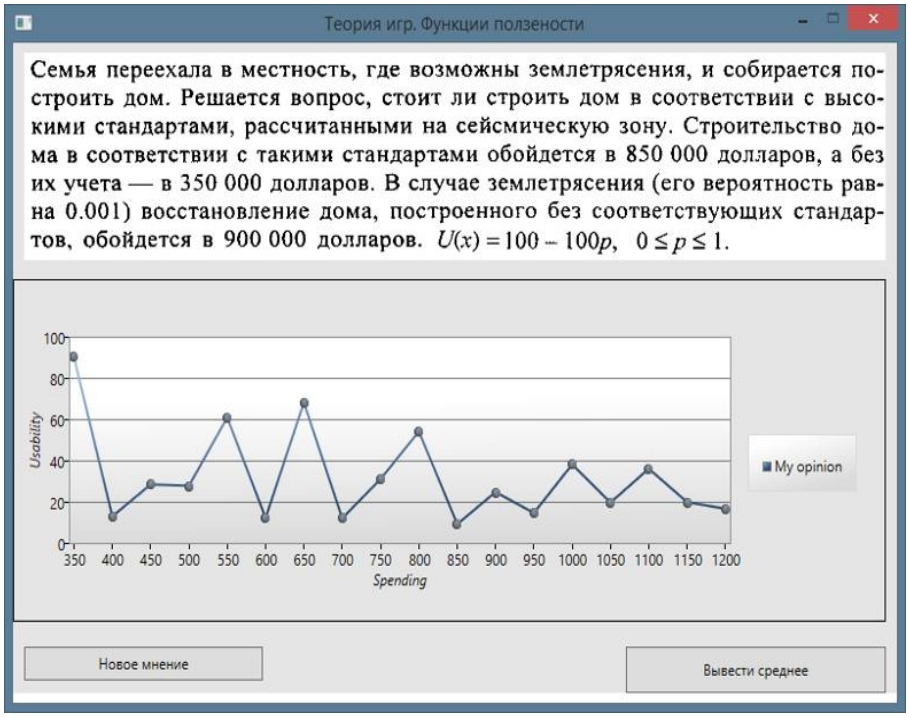


Рисунок 20 – Программа после запуска

По нажатию на кнопку «Новое мнение» генерируется новое мнение. Для примера, нажмем три раза, появится результат работы программы – три новых мнения по этому вопросу (рисунок 21).

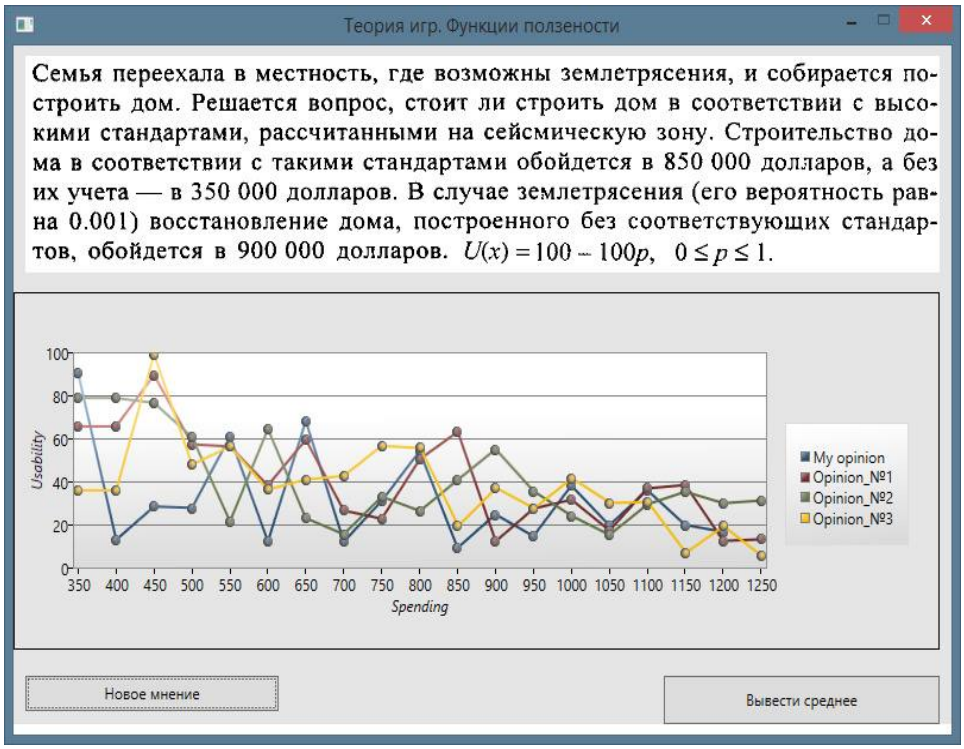


Рисунок 21– Работа программы

Нажмем еще 35 раз (выбрано относительно большое число) увидим, что различить что-либо на графике затруднительно (рисунок 22).

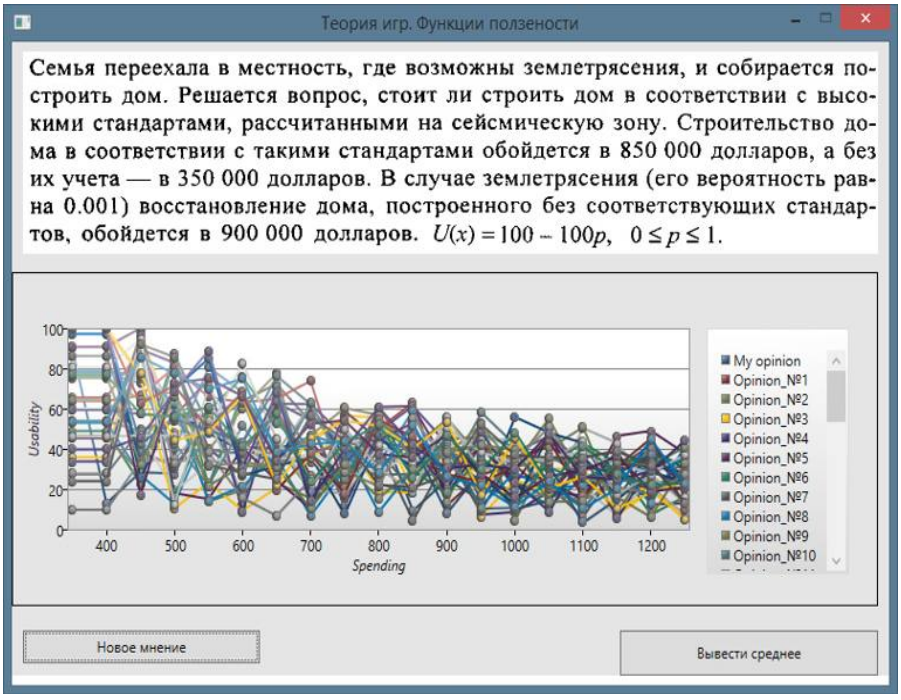


Рисунок 5 – График всех мнений

Для решения этой проблемы и была создана кнопка «Вывести среднее». Результатом ее нажатия является график функции полезности, построенные по средним значениям по каждой сумме денег (рисунок 23).

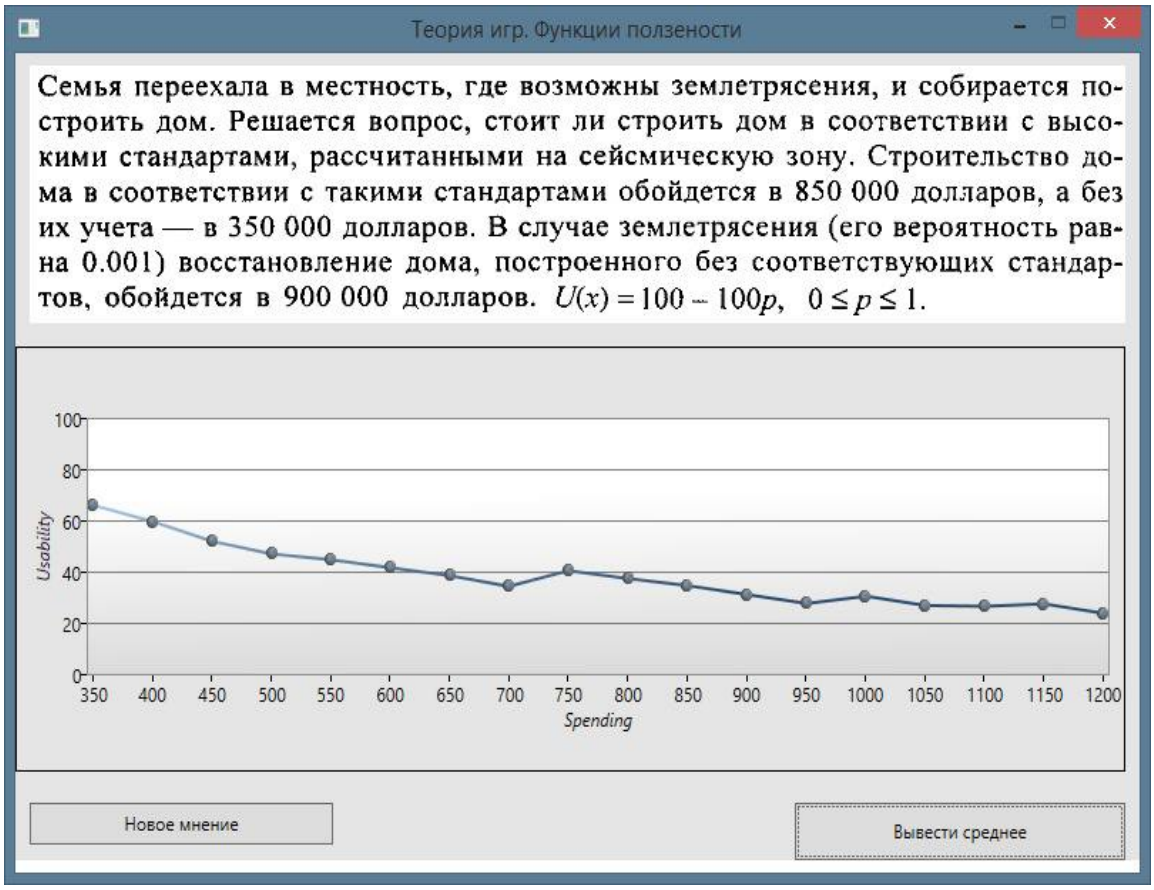


Рисунок 23 – График по среднему значению

Задачи для самостоятельного решения

Задача 1

Отдел кадров фирмы сузил поиск будущего сотрудника до трех кандидатур: Стив (S), Джейн (J) и Майса (M). Конечный отбор основан на трех критериях: собеседование (C), опыт работы (O) и рекомендации (P). Отдел кадров использует матрицу A (приведенную ниже) для сравнения трех критериев. После проведенного собеседования с тремя претендентами, сбора данных, относящихся к опыту их работы и рекомендациям, построены матрицы A_C, A_O и A_P. Какого из трех кандидатов следует принять на работу? Оцените согласованность данных.

$$\mathbf{A}=\begin{array}{c} \\ C \\ O \\ P \end{array}\begin{array}{c} \begin{array}{ccc} C & O & P \end{array} \\ \begin{pmatrix} 1 & 2 & \frac{1}{4} \\ \frac{1}{2} & 1 & \frac{1}{5} \\ 4 & 5 & 1 \end{pmatrix} \end{array},\quad \mathbf{A}_C=\begin{array}{c} \\ S \\ J \\ M \end{array}\begin{array}{c} \begin{array}{ccc} S & J & M \end{array} \\ \begin{pmatrix} 1 & 3 & 4 \\ \frac{1}{3} & 1 & \frac{1}{5} \\ \frac{1}{4} & 5 & 1 \end{pmatrix} \end{array},$$

$$\mathbf{A}_O=\begin{array}{c} \\ S \\ J \\ M \end{array}\begin{array}{c} \begin{array}{ccc} S & J & M \end{array} \\ \begin{pmatrix} 1 & \frac{1}{3} & 2 \\ 3 & 1 & \frac{1}{2} \\ \frac{1}{2} & 2 & 1 \end{pmatrix} \end{array},\quad \mathbf{A}_P=\begin{array}{c} \\ S \\ J \\ M \end{array}\begin{array}{c} \begin{array}{ccc} S & J & M \end{array} \\ \begin{pmatrix} 1 & \frac{1}{2} & 1 \\ 2 & 1 & \frac{1}{2} \\ 1 & 2 & 1 \end{pmatrix} \end{array}.$$

Задача 2

Кевин и Джун Парки (К и Д) покупают новый дом. Рассматриваются три варианта - A, B и C. Парки согласовали два критерия для выбора дома: площадь зеленой лужайки (Л) и близость к месту работы (Б), а также разработали матрицы сравнений, приведенные ниже. Необходимо оценить три дома в порядке их приоритета и вычислить коэффициент согласованности каждой матрицы.

$$\mathbf{A}=\begin{array}{c} \\ K \\ Д \end{array}\begin{array}{c} \begin{array}{cc} K & Д \end{array} \\ \begin{pmatrix} 1 & 2 \\ \frac{1}{2} & 1 \end{pmatrix} \end{array},\quad \mathbf{A}_K=\begin{array}{c} \\ Л \\ Б \end{array}\begin{array}{c} \begin{array}{cc} Л & Б \end{array} \\ \begin{pmatrix} 1 & \frac{1}{3} \\ 3 & 1 \end{pmatrix} \end{array},\quad \mathbf{A}_Д=\begin{array}{c} \\ Л \\ Б \end{array}\begin{array}{c} \begin{array}{cc} Л & Б \end{array} \\ \begin{pmatrix} 1 & 4 \\ \frac{1}{4} & 1 \end{pmatrix} \end{array},$$

$$\mathbf{A}_{КЛ}=\begin{array}{c} \\ A \\ B \\ C \end{array}\begin{array}{c} \begin{array}{ccc} A & B & C \end{array} \\ \begin{pmatrix} 1 & 2 & 3 \\ \frac{1}{2} & 1 & 2 \\ \frac{1}{3} & \frac{1}{2} & 1 \end{pmatrix} \end{array},\quad \mathbf{A}_{КБ}=\begin{array}{c} \\ A \\ B \\ C \end{array}\begin{array}{c} \begin{array}{ccc} A & B & C \end{array} \\ \begin{pmatrix} 1 & 2 & \frac{1}{2} \\ \frac{1}{2} & 1 & \frac{1}{3} \\ 2 & 3 & 1 \end{pmatrix} \end{array},$$

$$\mathbf{A}_{ДЛ}=\begin{array}{c} \\ A \\ B \\ C \end{array}\begin{array}{c} \begin{array}{ccc} A & B & C \end{array} \\ \begin{pmatrix} 1 & 4 & 2 \\ \frac{1}{4} & 1 & 3 \\ \frac{1}{2} & \frac{1}{3} & 1 \end{pmatrix} \end{array},\quad \mathbf{A}_{ДБ}=\begin{array}{c} \\ A \\ B \\ C \end{array}\begin{array}{c} \begin{array}{ccc} A & B & C \end{array} \\ \begin{pmatrix} 1 & \frac{1}{2} & 4 \\ 2 & 1 & 3 \\ \frac{1}{4} & \frac{1}{3} & 1 \end{pmatrix} \end{array}.$$

Задача 3

Автор книги по исследованию операций определил три критерия для выбора издательства, которое будет печатать его книгу: процент авторского гонорара (R), уровень маркетинга (M) и размер аванса (A). Издательства H и P проявили интерес к изданию книги. Используя приведенные ниже матрицы сравнения, необходимо дать оценку двум издательствам и оценить согласованность решения.

$$\mathbf{A}=\begin{array}{c} \\ R \\ M \\ A \end{array}\begin{array}{c} \begin{array}{ccc} R & M & A \end{array} \\ \begin{pmatrix} 1 & 1 & \frac{1}{4} \\ 1 & 1 & \frac{1}{5} \\ 4 & 5 & 1 \end{pmatrix} \end{array}, \quad \mathbf{A}_R=\begin{array}{c} \\ H \\ P \end{array}\begin{array}{c} \begin{array}{cc} H & P \end{array} \\ \begin{pmatrix} 1 & 2 \\ \frac{1}{2} & 1 \end{pmatrix} \end{array}, \quad \mathbf{A}_M=\begin{array}{c} \\ H \\ P \end{array}\begin{array}{c} \begin{array}{cc} H & P \end{array} \\ \begin{pmatrix} 1 & \frac{1}{2} \\ 2 & 1 \end{pmatrix} \end{array}, \quad \mathbf{A}_A=\begin{array}{c} \\ H \\ P \end{array}\begin{array}{c} \begin{array}{cc} H & P \end{array} \\ \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix} \end{array}$$

Задача 4

Профессор политологии планирует предсказать исход выборов в местный школьный совет. Кандидаты I, B и S баллотируются на одно место. Профессор делит всех избирателей на три категории: левые (L), центристы (C) и правые (R). Оценка кандидатов основывается на трех факторах: педагогический опыт (О), отношение к детям (Д) и характер (Х). Ниже приведены матрицы сравнения для первого иерархического уровня, связанного с градацией избирателей (левые, центристы и правые).

$$\mathbf{A}=\begin{array}{c} \\ L \\ C \\ R \end{array}\begin{array}{c} \begin{array}{ccc} L & C & R \end{array} \\ \begin{pmatrix} 1 & 2 & \frac{1}{2} \\ \frac{1}{2} & 1 & \frac{1}{5} \\ 2 & 5 & 1 \end{pmatrix} \end{array}, \quad \mathbf{A}_L=\begin{array}{c} \\ О \\ Д \\ Х \end{array}\begin{array}{c} \begin{array}{ccc} О & Д & Х \end{array} \\ \begin{pmatrix} 1 & 3 & \frac{1}{2} \\ \frac{1}{3} & 1 & \frac{1}{3} \\ 2 & 3 & 1 \end{pmatrix} \end{array},$$

$$\mathbf{A}_C=\begin{array}{c} \\ О \\ Д \\ Х \end{array}\begin{array}{c} \begin{array}{ccc} О & Д & Х \end{array} \\ \begin{pmatrix} 1 & 2 & 2 \\ \frac{1}{2} & 1 & 1 \\ \frac{1}{2} & 1 & 1 \end{pmatrix} \end{array}, \quad \mathbf{A}_R=\begin{array}{c} \\ О \\ Д \\ Х \end{array}\begin{array}{c} \begin{array}{ccc} О & Д & Х \end{array} \\ \begin{pmatrix} 1 & 1 & 9 \\ 1 & 1 & 8 \\ \frac{1}{9} & \frac{1}{8} & 1 \end{pmatrix} \end{array}$$

Профессор сгенерировал еще девять матриц сравнения для трех кандидатов на втором иерархическом уровне, связанном с педагогическим опытом, отношением к детям и характером. Затем был использован метод анализа иерархий для сведения этих матриц к следующим относительным весам.

Кандидат	Левые			Центристы			Правые		
	О	Д	Х	О	Д	Х	О	Д	Х
I	0,1	0,2	0,3	0,3	0,5	0,2	0,7	0,1	0,3
B	0,5	0,4	0,2	0,4	0,2	0,4	0,1	0,4	0,2
S	0,4	0,4	0,2	0,5	0,3	0,4	0,2	0,5	0,5

Используя эту информацию, необходимо определить, кто из кандидатов выиграет выборы, и оценить согласованность решения.

Задача 5

Школьный округ крайне заинтересован в сокращении своих расходов, что вызвано очередным уменьшением бюджетного финансирования начальных школ. Есть две возможности решить эту проблему: ликвидировать программу физического воспитания (Ф) или программу музыкального образования (М). Управляющий округа сформировал комитет с равным представительством от местного школьного совета (С) и ассоциации родителей и учителей (Р) для изучения ситуации и выработки предложения. Комитет принял решение изучить ситуацию с точки зрения ограничения бюджета (Б) и потребностей учеников (П). Проведенный анализ дал следующие матрицы сравнения.

$$\mathbf{A}_C = \begin{matrix} \\ Б \\ П \end{matrix}\begin{matrix} \begin{matrix} Б & П \end{matrix} \\ \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix} \end{matrix}, \quad \mathbf{A}_P = \begin{matrix} \\ Б \\ П \end{matrix}\begin{matrix} \begin{matrix} Б & П \end{matrix} \\ \begin{pmatrix} 1 & 2 \\ \frac{1}{2} & 1 \end{pmatrix} \end{matrix},$$

$$\mathbf{A}_{СБ} = \begin{matrix} \\ Ф \\ М \end{matrix}\begin{matrix} \begin{matrix} Ф & М \end{matrix} \\ \begin{pmatrix} 1 & \frac{1}{2} \\ 2 & 1 \end{pmatrix} \end{matrix}, \quad \mathbf{A}_{СП} = \begin{matrix} \\ Ф \\ М \end{matrix}\begin{matrix} \begin{matrix} Ф & М \end{matrix} \\ \begin{pmatrix} 1 & \frac{1}{3} \\ 3 & 1 \end{pmatrix} \end{matrix},$$

$$\mathbf{A}_{РБ} = \begin{matrix} \\ Ф \\ М \end{matrix}\begin{matrix} \begin{matrix} Ф & М \end{matrix} \\ \begin{pmatrix} 1 & \frac{1}{3} \\ 3 & 1 \end{pmatrix} \end{matrix}, \quad \mathbf{A}_{РП} = \begin{matrix} \\ Ф \\ М \end{matrix}\begin{matrix} \begin{matrix} Ф & М \end{matrix} \\ \begin{pmatrix} 1 & 2 \\ \frac{1}{2} & 1 \end{pmatrix} \end{matrix}.$$

Проанализируйте ситуацию, связанную с принятием решения, и выработайте соответствующее предложение.

Задача 6

Решив купить автомобиль, человек сузил свой выбор до трех моделей: М1, М2 и М3. Факторами, влияющими на его решение, являются: стоимость

автомобиля (С), стоимость обслуживания (О), стоимость поездки по городу (Г) и сельской местности (М). Следующая таблица содержит необходимые данные, соответствующие трехгодичному сроку эксплуатации автомобиля.

Модель автомобиля	С (долл)	О (долл)	Г (долл)	М (долл)
М1	6 000	1 800	4 500	1 500
М2	8 000	1 200	2 250	750
М3	10 000	600	1 125	600

Используйте указанные стоимости для построения матриц сравнений. Оцените согласованность матриц и определите модель автомобиля, которую следует выбрать.

Задача 7

Фермер Мак-Кой может выращивать либо кукурузу, либо соевые бобы. Вероятность того, что цены на будущий урожай этих культур повысятся, останутся на том же уровне или понизятся, равна соответственно 0,25, 0,30 и 0,45. Если цены возрастут, урожай кукурузы даст 30 000 долл. чистого дохода, а урожай соевых бобов – 10000 долл. Если цены останутся неизменными, Мак-Кой лишь покроет расходы. Но если цены станут ниже, урожай кукурузы и соевых бобов приведет к потерям в 35 000 и 5 000 долл. соответственно.

a) Представьте данную задачу в виде дерева решений.

b) КакуюкультуруследуетвыращиватьМак-Кою?

Задача 8

Допустим, у вас имеется возможность вложить деньги в три инвестиционных фонда открытого типа: простой, специальный (обеспечивающий максимальную долгосрочную прибыль от акций мелких компаний) и глобальный. Прибыль от инвестиции может измениться в зависимости от условий рынка. Существует 10%-ная вероятность, что ситуация на рынке ценных бумаг ухудшится, 50%-ная – что рынок останется умеренным и 40%-ная – рынок будет возрастать. Следующая таблица содержит значения процентов прибыли от суммы инвестиции при трех возможностях развития рынка.

Альтернатива (фонды)	Процент прибыли от инвестиции		
	Ухудшающийся рынок	Умеренный рынок	Растущий рынок
Простой	+5	+7	+8

Специальный	-10	+5	+30
Глобальный	+2	+7	+20

a) Представьте задачу в виде дерева решений.

b) Какой фонд открытого типа вам следует выбрать?

Задача 9

Предположим, у вас имеется возможность вложить деньги либо в 7,5%-ные облигации, которые продаются по номинальной цене, либо в специальный фонд, который выплачивает лишь 1% дивидендов. Если существует вероятность инфляции, процентная ставка возрастет до 8%, и в этом случае номинальная стоимость облигаций увеличится на 10%, а цена акций фонда – на 20%. Если прогнозируется спад, то процентная ставка понизится до 6%. При этих условиях ожидается, что номинальная стоимость облигаций поднимется на 5%, а цена акций фонда увеличится на 20%. Если состояние экономики останется неизменным, цена акций фонда увеличится на 8%, а номинальная стоимость облигаций не изменится. Экономисты оценивают в 20% шансы наступления инфляции и в 15% - наступление спада. Ваше решение относительно инвестиций принимается с учетом экономических условий следующего года.

a) Представьтезадачуввидедереварешений.

b) Будете ли вы покупать акции фонда или облигации?

Задача 10

Фирма планирует производство новой продукции быстрого питания в национальном масштабе. Исследовательский отдел убежден в большом успехе новой продукции и хочет внедрить ее немедленно, без рекламной кампании на рынках сбыта фирмы. Отдел маркетинга положение вещей оценивает иначе и предлагает провести интенсивную рекламную кампанию. Такая кампания обойдется в 100 000 долл., а в случае успеха принесет 950 000 долл. годового дохода. В случае провала рекламной кампании (вероятность этого составляет 30%) годовой доход оценивается лишь в 200 000 долл. Если рекламная кампания не проводится вовсе, годовой доход оценивается в 400 000 долл. при условии, что покупателям понравится новая продукция (вероятность этого равна 0,8), и в 200 000 долл. с вероятностью 0,2, если покупатели останутся равнодушными к новой продукции.

a) Постройтесоответствующеедереворешений.

b)

Какдолжнапоступитьфирмавсвязиспроизводствомновойпродукции?

Задача 11

Симметричная монета подбрасывается три раза. Вы получаете один доллар за каждое выпадение герба (Г) и дополнительно 0,25 доллара за каждые два последовательных выпадения герба (заметим, что выпадение ГГГ состоит из двух последовательностей ГГ). Однако вам приходится платить 1,1 долл. за каждое выпадение решки (Р). Вашим решением является участие или неучастие в игре.

a) Постройтесоответствующеедеревореш енийдляописаннойигры.

b) Будете ли вы играть в эту игру?

Задача 12

Предположим, у вас имеется возможность сыграть в игру следующего содержания. Симметричная игральная кость бросается два раза, при этом возможны четыре исхода: 1) выпадает два четных числа, 2) выпадает два нечетных числа, 3) выпадает сначала четное, затем нечетное число, 4) выпадает сначала нечетное, затем четное число. Вы можете делать одинаковые ставки на два исхода. Например, вы можете поставить на два четных числа (исход 1) и два нечетных (исход 2). Выигрыш на каждый доллар, поставленный на первый исход, равен 2 доллара, на второй и третий исходы – 1,95 доллара, на четвертый - 1,50 доллара.

a) Постройтедереворешенийдляописаннойигры.

b) Накакиеисходыследуетделатьставки?

c) Можно ли иметь стабильный выигрыш в этой игре?

Задача 13

Фирмапроизводитпартииипродукциис0,8, 1, 1,2и 1,4% бракованных изделий с вероятностями 0,4, 0,3, 0,25 и 0,05 соответственно. Три потребителя A, B и C заключили контракт на получение партий изделий с процентом некачественных изделий не выше 0,8, 1,2 и 1,4 % соответственно. Фирма штрафуется в сумме 1000 долл. за каждый пункт процента (это одна десятая процента) в случае, если процент некачественных изделий выше указанного. Наоборот, поставка партий изделий с меньшим процентом бракованных изделий, чем оговорено в контракте, приносит фирме прибыль в 500 долл. за каждый пункт процента. Предполагается, что партии изделий перед отправкой не проверяются.

a) Постройтесоответствующеедеревореш ений.

b) Какой из потребителей должен иметь наивысший приоритет при получении своего заказа?

Задача 14

Фирма планирует открыть новое предприятие в Арканзасе. В настоящее время имеется возможность построить либо крупное предприятие, либо небольшое, которое через два года можно будет расширить при условии высокого спроса на выпускаемую им продукцию. Рассматривается задача принятия решений на десятилетний период. Фирма оценивает, что на протяжении этих 10 лет вероятность высокого и низкого спроса на производимую продукцию будет равна 0,75 и 0,25 соответственно. Стоимость немедленного строительства крупного предприятия равна 5 млн. долл., а небольшого – 1 млн. долл. Расширение малого предприятия через два года обойдется фирме в 4,2 млн. долл. Прибыль, получаемая от функционирования производственных мощностей на протяжении 10 лет, приводится в следующей таблице.

Альтернатива	Ожидаемый доход за год (тыс. долл.)	
	Высокий спрос	Низкий спрос
Крупное предприятие сейчас	1 000	300
Небольшое предприятие сейчас	250	200
Расширенное предприятие через 2 года	900	200

a) Постройте соответствующее дерево решений, принимая во внимание, что через два года фирма может либо расширить небольшое предприятие, либо не расширять его.

b) Сформулируйте стратегию строительства для фирмы на планируемый 10-летний период. (Для простоты не принимайте во внимание возможную инфляцию.)

Задача 15

Решите предыдущее упражнение, предположив, что ежегодная учетная ставка равна 10 % и что решение принимается с учетом инфляции.

Задача 16

Решите упражнение 14, предположив, что спрос может быть высоким, средним и низким с вероятностями 0,7, 0,2 и 0,1 соответственно. Расширение небольшого предприятия будет проведено лишь в том случае, если на протяжении первых двух лет спрос будет высоким. Следующая таблица содержит данные о прибылях за год.

Альтернатива	Ожидаемый доход за год (тыс. долл.)

	Высокий спрос	Средний спрос	Низкий спрос
Крупное предприятие сейчас	1 000	500	300
Небольшое предприятие сейчас	400	280	150
Расширенное предприятие через 2 года	900	600	200

Задача 17

Электроэнергетическая компания использует парк из 20 грузовых автомобилей для обслуживания электрической сети. Компания планирует периодический профилактический ремонт автомобилей. Вероятность поломки автомобиля в первый месяц равна нулю, во второй месяц – 0,03 и увеличивается на 0,01 для каждого последующего месяца, по десятый включительно. Начиная с одиннадцатого месяца и далее, вероятность поломки сохраняется постоянной на уровне 0,13. Случайная поломка одного грузового автомобиля обходится компании в 200 долл., а планируемый профилактический ремонт в 75 долл.

Компания хочет определить оптимальный период (в месяцах) между планируемыми профилактическими ремонтами.

a) Постройтесоответствующеедеревореше ний.

b) Определите оптимальную длину цикла для профилактического ремонта.

Задача 18

Ежедневный спрос на булочки в продовольственном магазине задается следующим распределением вероятностей.

n	100	150	200	250	300
p_n	0,20	0,25	0,30	0,15	0,10

Магазин покупает булочку по 55 центов, а продает по 1,20 долл. Если булочка не продана в тот же день, то к концу дня она может быть реализована за 25 центов. Величина запаса булочек может принимать одно из возможных значений спроса, которые перечислены выше.

a) Постройте соответствующее дерево решений.

b) Сколько булочек необходимо заказывать ежедневно?

Задача 19

Пусть в предыдущем упражнении временной интервал, для которого необходимо решить задачу принятия решений, составляет два дня. Альтернативы для второго дня зависят от объема реализации булочек в первый день. Если реализован в точности весь запас первого дня, магазин закажет такое же количество булочек и на второй день. Если потребность в булочках в первый день превышает имеющийся запас, то для второго дня магазин может заказать любой из объемов спроса на булочки, который превышает запас первого дня. И наконец, если в первый день реализовано меньше булочек, чем было закуплено, то для второго дня магазин может заказать любой из объемов спроса на булочки, который меньше запаса первого дня. Постройте соответствующее дерево решений и определите оптимальную стратегию заказа.

Задача 20

Наружный диаметр d цилиндра, производимого автоматом, имеет верхнее инижнее допустимые значения $d+t_U$ и $d-t_L$ соответственно. Производственный процесс настроен так, что величина диаметра является нормально распределенной случайной величиной с математическим ожиданием μ и стандартным отклонением σ. Каждый цилиндр со значением диаметра, превышающим верхнее допустимое значение, доводится до нужных размеров за c_1 долл. Цилиндр, диаметр которого меньше установленной нижней нормы, реализуется с убытком c_2 долл. Определите оптимальное значение настройки для автомата.

Лабораторная работа
«Целочисленное линейное программирование»

Цель работы: изучение принципов решения задач целочисленного программирования и реализация изученных методов на практике в виде программной реализации.

Порядок выполнения:

1. Разработать математическую модель решения задачи согласно варианту;
2. Разработать программу решающую поставленную задачу в общем виде;
3. Составить отчет по работе.

Теоритические сведения

Специфика задач целочисленного программирования заключается в том, что на переменные x_i и функцию цели F(x) налагается дополнительное ограничение – условие целочисленности. Целочисленное программирование иногда называютдискретным программированием. Если требование целочисленности распространяется не на все переменные, а только на часть из них, то задача называется частично целочисленной.

Методы решения задач целочисленного линейного программирования основаны на использовании вычислительных возможностей методов линейного программирования. Обычно алгоритмы целочисленного программирования включают три шага.

Шаг 1. "Ослабление" пространства допустимых решений задачи целочисленного линейного программирования путем замены любой двоичной переменной y непрерывным ограничением $0 < y < 1$ и отбрасывания требования целочисленности для всех остальных переменных. В результате получается обычная задача линейного программирования.

Шаг 2. Решение задачи линейного программирования и определение ее оптимального решения.

Шаг 3. Имея полученное (непрерывное) оптимальное решение, добавляем специальные ограничения, которые итерационным путем изменяют пространство допустимых решений задачи линейного программирования таким образом, чтобы в конечном счете получилось оптимальное решение, удовлетворяющее требованиям целочисленности.

Разработаны два общих метода генерирования специальных ограничений, о которых идет речь при реализации шага 3.

1. Метод ветвей и границ.
2. Метод отсекающих плоскостей.

Метод ветвей и границ.

Метод ветвей и границ относится к комбинаторным методам решения целочисленных задач и применим как к полностью, так и к частично целочисленным задачам.

Суть метода ветвей и границ – в направленном частичном переборе допустимых решений. Будем рассматривать задачу линейного программирования. Вначале она решается без ограничений на целочисленность. При этом находится верхняя граница F(x), так как целочисленное решение не может улучшить значение функции цели.

Далее в методе ветвей и границ область допустимых значений переменных разбивается на ряд непересекающихся областей (ветвление), в каждой из которых оценивается экстремальное значение функции. Если целое решение не найдено, ветвление продолжается.

Ветвление производится последовательным введением дополнительных ограничений. Пусть xk – целочисленная переменная, значение которой в оптимальном решении получилось дробным. Интервал $[\beta k] \leq xk \leq [\beta k]+1$ не содержит целочисленных компонентов решения. Поэтому допустимое целое значение xk должно удовлетворять одному из неравенств $xk \geq [\beta k]+1$ или $xk \leq [\beta k]$. Это и есть дополнительные ограничения. Введение их в методе ветвей и границ на каждом шаге порождает две не связанные между собой подзадачи. Каждая подзадача решается как задача линейного программирования с исходной целевой функцией. После конечного числа шагов будет найдено целочисленное оптимальное решение.

Метод отсекающих плоскостей

Данный метод, как и метод ветвей и границ, начинает работу с оптимального решения "обычной" (непрерывной) задачи линейного программирования. Однако вместо ветвления и построения границ этот метод видоизменяет пространство допустимых решений, последовательно прибавляя специальным образом построенные ограничения (именуемые отсечениями).

Метод отсекающих плоскостей начинает работу с решения непрерывной задачи линейного программирования. В симплекс-таблице, соответствующей оптимальному решению задачи линейного программирования, следует выбрать одну из строк (называемую

производящей), для которой базисная переменная нецелочисленная. Искомое отсечение строится на основании дробных составляющих коэффициентов производящей строки. По этой причине его называют дробным отсечением.

Данный метод путем добавления отсечений (отсекающих плоскостей) преобразует пространство допустимых решений соответствующей задачи линейного программирования в выпуклый многогранник, вершина которого, соответствующая оптимуму, является целочисленной и представляет решение исходной задачи. На рисунке 24 показан пример двух таких отсечений. Мы начинаем с оптимального решения непрерывной задачи линейного программирования (x_1, x_2) = (41,31) и z = 661. Затем прибавляем отсечение I, которое вместе с ограничениями исходной задачи линейного программирования приводит к оптимальному решению (x_1 x_2) = (4, 4/3; 3) с z = 62. После этого прибавляется отсечение II, которое вместе с отсечением I и исходными ограничениями приводит к оптимальному решению (x_1 x_2) = (4, 3) с z = 58. Последнее решение является полностью целочисленным, как и требуется.

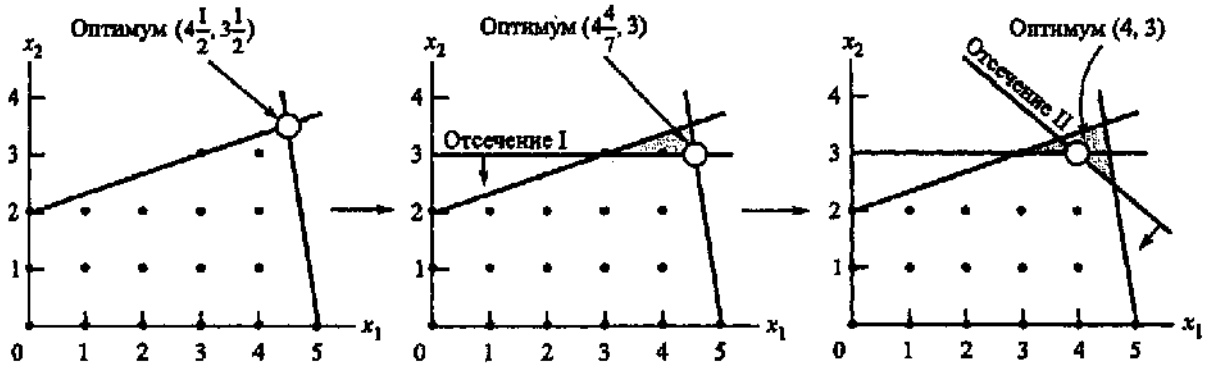


Рисунок 24 – Пример отсечений

Прибавленные отсечения не отбрасывают ни одной исходной допустимой целочисленной точки, но должны проходить по меньшей мере через одну целочисленную точку (допустимую или недопустимую). Этим основным требованиям должно удовлетворять любое отсечение.

В общем случае может потребоваться любое (конечное) число отсечений для достижения полностью целочисленной экстремальной точки. В действительности количество необходимых дня этого отсечений не зависит от размерности задачи в том смысле, что для решения задачи с небольшим количеством переменных и ограничений может потребоваться больше отсечений, чем для задачи большой размерности.

Пример построения математической модели

Методом ветвей и границ найти максимальное значение функции

$F(x) = 2x_1 + 3x_2$

при ограничениях

$3x_1 + 4x_2 \leq 24$

$2x_1 + 5x_2 \leq 22$

$x_{1,2} \geq 0$ - целые

1-й шаг метода ветвей и границ. Решается задача линейного программирования с отброшенными условиями целочисленности с помощью симплекс-метода (табл. 1 – 3).

По данным табл. 3 запишем оптимальное нецелое решение

$$x_1' = 2\frac{4}{7}$$

$$x_2' = 2\frac{4}{7}$$

$$F_{max} = 16\frac{6}{7}$$

Базисные переменные	Свободные члены	Небазисные переменные	
		x_1	x_2
x_3	24	3	4
x_4	22	2	5
F	0	-2	-3

Таблица 1 - симплекс-таблица для задачи ЛП

Базисные переменные	Свободные члены	Небазисные переменные	
		x_1	x_4
x_3	32/5	7/5	-4/5
x_2	22/5	2/5	1/5
F	66/5	-4/5	3/5

Таблица 2 - симплекс-таблица для задачи ЛП

Базисные переменные	Свободные члены	Небазисные переменные	
		x_3	x_4
x_1	32/7	5/7	-4/7
x_2	18/7	-2/5	3/7
F	118/7	4/7	1/7

Таблица 3 - симплекс-таблица для задачи ЛП

Графическая интерпретация задачи приведена на рис. 2. Здесь ОДЗП представлена четырехугольником ABCD, а координаты вершины C совпадают с x_1' и x_2'. Обе переменные в оптимальном решении являются нецелыми, поэтому любая из них может быть выбрана в качестве переменной, инициирующей процесс ветвления.

Пусть это будет x_2. Выбор x_2 порождает две подзадачи (2 и 3), одна из них получается путем добавления ограничения $x_2 \geq 3$ к исходной задаче, а другая – путем добавления ограничения $x_2 \leq 2$. При этом ОДЗП разбивается на две заштрихованные области (рис. 2), а полоса значений $2 < x_2 < 3$ исключается из рассмотрения. Однако множество допустимых целочисленных решений сохраняется, порожденные подзадачи содержат все целочисленные решения исходной задачи.

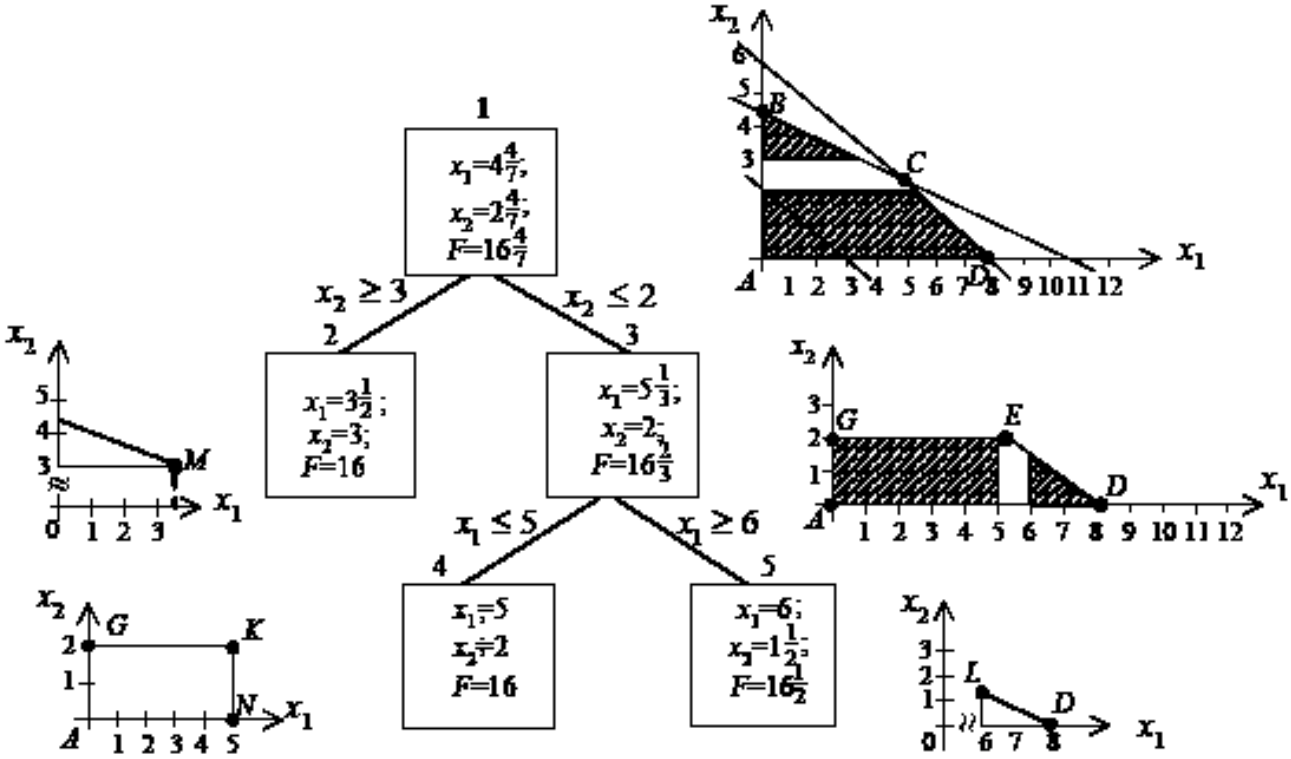


Рисунок 25 - графическая интерпритация решения примера методом ветвей и границ

2-й шаг метода ветвей и границ. Осуществляется выбор одной из обозначенных ранее подзадач. Не существует точных методов определения, какой из подзадач отдать предпочтение. Случайный выбор приводит к разным последовательностям подзадач и, следовательно, к различным количествам итераций, обеспечивающих получение оптимального решения.

Пусть вначале решается подзадача 3 с дополнительным ограничением $x_2 \leq 2$ или $x_2 + x_5 = 2$. Из табл. 3 для переменной x_2 справедливо следующее выражение $-2/7x_3 + 3/7x_4 + x_2 = 18/7$ или $x_2 = 18/7 + 2/7x_3 - 3/7x_4$, тогда $2/7x_3$ -

$3/7x_4 + x_5 = -4/7$. Включаем ограничение в табл. 3, при этом получим новую таблицу (табл. 4).

Осуществляя оптимизацию решения, переходим к табл. 5, которой соответствует решение

$$x_1' = 5\frac{1}{3}$$
$$x_2' = 2$$
$$F_{max} = 16\frac{2}{3}$$

Переменная x_1 нецелая, поэтому ветвление необходимо продолжить; при этом возникают подзадачи 4 и 5 с ограничениями $x_1 \leq 5$ и $x_1 \geq 6$ соответственно. Полоса значений $5 < x_1 < 6$ исключается из рассмотрения.

Базисные переменные	Свободные члены	Небазисные переменные	
		x_3	x_4
x_1	32/7	5/7	-4/7
x_2	18/7	-2/5	3/7
x_5	-4/7	2/7	-3/7
F	118/7	4/7	1/7

Таблица 4 - симплекс-таблица для задачи ЛП

Базисные переменные	Свободные члены	Небазисные переменные	
		x_3	x_5
x_1	16/3	1/3	-4/3
x_2	2	0	1
x_4	4/3	-2/3	-7/3
F	50/3	2/3	1/3

Таблица 5 - симплекс-таблица для задачи ЛП

3-й шаг метода ветвей и границ. Решаются подзадачи 4 и 5. Из рис. 1 видно, что оптимальное целочисленное решение подзадачи 4 достигается в вершине К с координатами $x_1' = 5$, $x_2' = 2$, однако это не означает, что найден оптимум исходной задачи. Причиной такого вывода являются еще не решенные подзадачи 3 и 5, которые также могут дать целочисленные решения. Найденное целочисленное решение F = 16 определяет нижнюю границу значений целевой функции, т.е. меньше этого значения оно быть не должно.

Подзадача 5 предполагает введение дополнительного ограничения $x_1 \geq 6$ в подзадачу 3 . Графическое решение на рис. 2 определяет вершину L с координатами $x_1' = 6$, $x_2' = 3/2$, в которой достигается оптимальное решение подзадачи 5: Fmax = 16.5 . Дальнейшее ветвление в этом направлении осуществлять нецелесообразно, так как большего, чем 16, целого значения функции цели получить невозможно. Ветвление подзадачи 5 в лучшем случае приведёт к другому целочисленному решению, в котором F = 16.

4-й шаг метода ветвей и границ. Исследуется подзадача 2 с ограничением $x_2 \geq 3$, находится её оптимальное решение, которое соответствует вершине M (рис. 2) с координатами $x_1' = 3.5$, $x_2' = 3$. Значение функции цели при этом Fmax = 16, которое не превышает найденного ранее решения. Таким образом, поиск вдоль ветви $x_2 \geq 3$ следует прекратить.

Пример программы, решающую поставленную задачу в общем виде:

Рассмотрим две задачи статической модели управления запасами: «Классическая задача экономичного размера заказа» и «Задача экономичного размера заказа с разрывами цен».

Основное диалоговое окно программы представлено на рисунке 26.

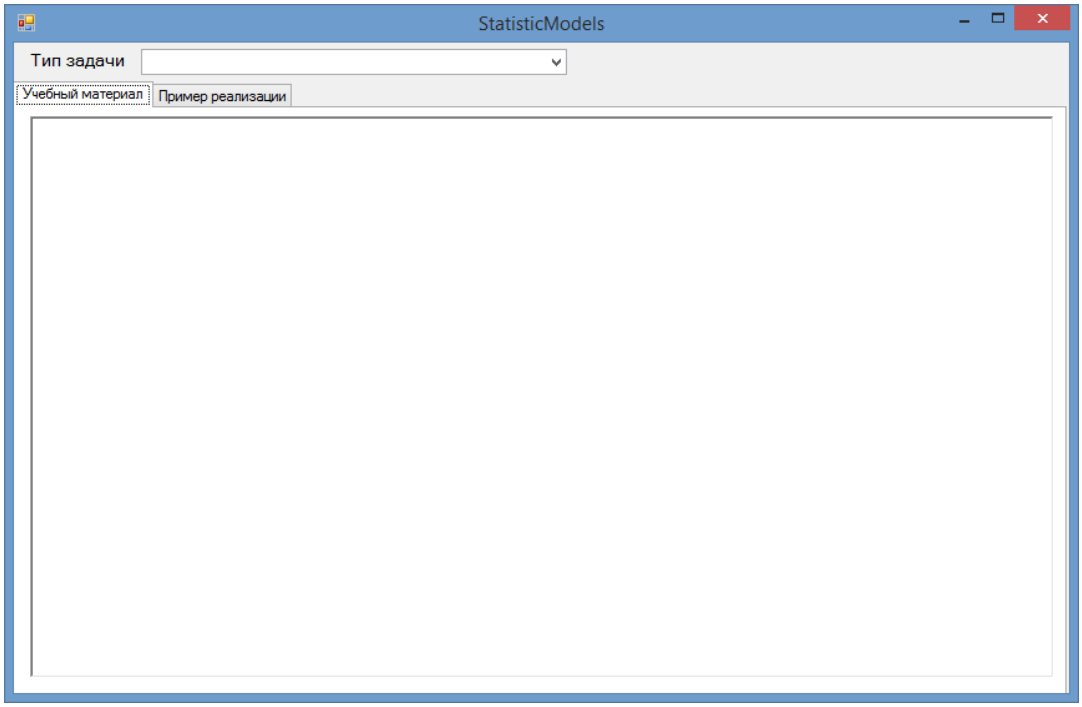


Рисунок 26 – Основное окно программы

Чтобы начать работу с программой пользователь должен указать тип задачи, который он хотел бы просмотреть, для этого нужно выбрать её в раскрывающимся списке. После чего на экране появиться теоретический материал по интересующей задаче как это показано на рисунке 27.

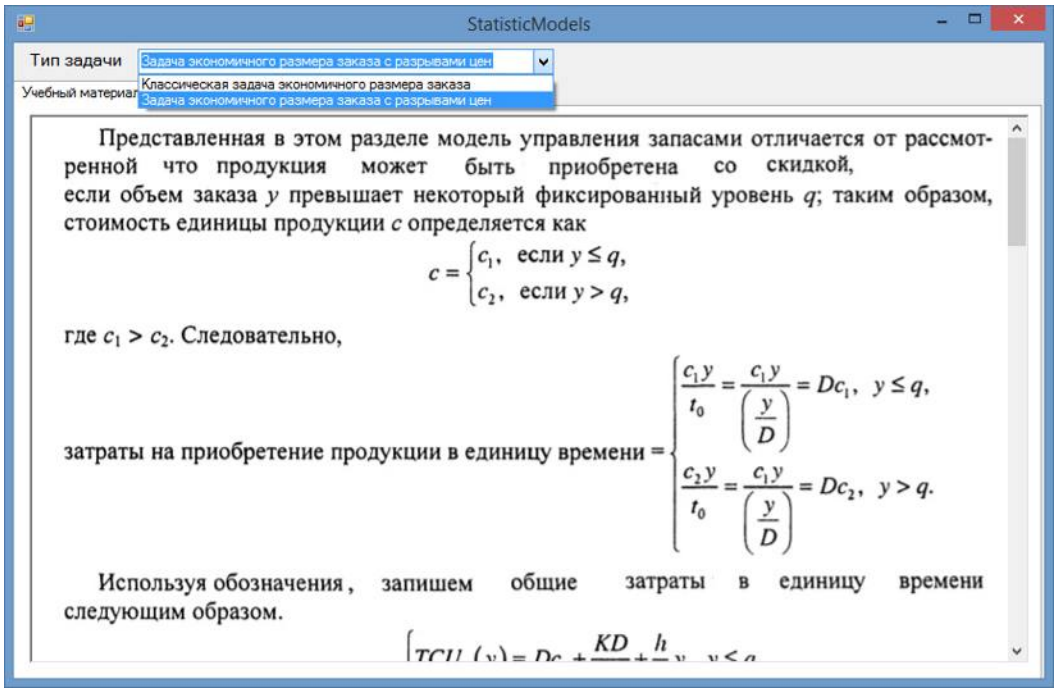


Рисунок 27 – Выбор типа решаемой задачи

На вкладке «Пример реализации» приведен пример решения задачи выбранного типа (рисунок 28).

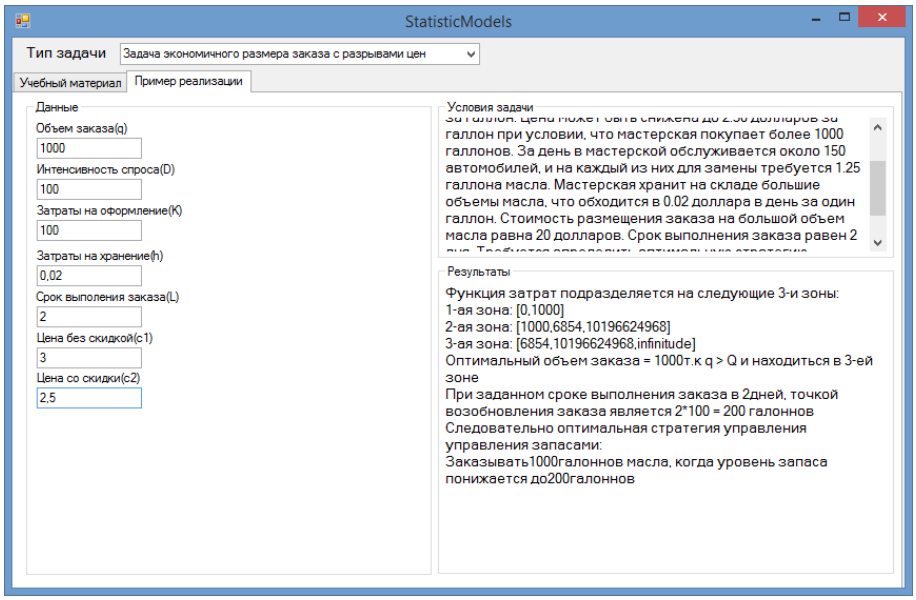


Рисунок 28 – Пример реализации

Листинг:

Модуль №1.

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Xml;

namespace inventory_Management
{
    public partial class StatisticModels :  Form
    {
        public StatisticModels()
        {
            InitializeComponent();
        }
        Work_xml WX = new Work_xml();
        any_works AW = new any_works();
        List<string> listText = new List<string>();
        List<string> listImg = new List<string>();
        List<string> listTask = new List<string>();

        private void comboBox1_SelectedIndexChanged(object sender, EventArgs e)
        {

            Clear_all_elem_text();
            Image img;
            if (comboBox1.SelectedItem != "")
            {
                if (comboBox1.SelectedIndex == 0)
                {
                    listImg = WX.get_img("statisticModel_1");
                    listTask = WX.get_task("statisticModel_1");
                    richTextBox3.AppendText(listTask[0]);
                    img = Image.FromFile(@"" + listImg[0]);
                    Clipboard.SetImage(img);
                    richTextBox1.Paste();
                    img = Image.FromFile(@"" + listImg[1]);
                    Clipboard.SetImage(img);
                    richTextBox1.Paste();
                    img = Image.FromFile(@"" + listImg[2]);
                    Clipboard.SetImage(img);
                    richTextBox1.Paste();
                    img = Image.FromFile(@"" + listImg[3]);
                    Clipboard.SetImage(img);
                    richTextBox1.Paste();
                    img = Image.FromFile(@"" + listImg[4]);
                    Clipboard.SetImage(img);
                    richTextBox1.Paste();
                    richTextBox1.ReadOnly = true;
                    richTextBox3.ReadOnly = true;


                    //--Work with Data
                    label1.Text = "Количество дней(y, предполагаемый срок выполнения)";
//this is "y"
                    label2.Text = "Интенсивность спроса(D)"; //----"D"
```

```
            label3.Text = "Затраты на оформление(K)"; //----"K"
            label4.Text = "Затраты на хранение(h)"; //-----"h"
            label5.Text = "Срок выполения заказа(L)"; //---"L"
            //-hide
            label6.Hide();
            textBox6.Hide();
            label7.Hide();
            textBox7.Hide();
        }
        else if (comboBox1.SelectedIndex == 1)
        {
           // listText = WX.get_text("statisticModel_2");
           // richTextBox1.AppendText(listText[0]);
            listTask = WX.get_task("statisticModel_2");
            listImg = WX.get_img("statisticModel_2");
            richTextBox3.AppendText(listTask[0]);
            img = Image.FromFile(@"" + listImg[0]);
            Clipboard.SetImage(img);
            richTextBox1.Paste();
            img = Image.FromFile(@"" + listImg[1]);
            Clipboard.SetImage(img);
            richTextBox1.Paste();
            img = Image.FromFile(@"" + listImg[2]);
            Clipboard.SetImage(img);
            richTextBox1.Paste();
            img = Image.FromFile(@"" + listImg[3]);
            Clipboard.SetImage(img);
            richTextBox1.Paste();
            richTextBox1.ReadOnly = true;
            richTextBox3.ReadOnly = true;
            //--Show element
            label6.Show();
            textBox6.Show();
            label7.Show();
            textBox7.Show();
            //-work witj data
            label1.Text = "Объем заказа(q)"; //-----------"q"
            label2.Text = "Интенсивность спроса(D)"; //----"D"
            label3.Text = "Затраты на оформление(K)"; //----"K"
            label4.Text = "Затраты на хранение(h)"; //-----"h"
            label5.Text = "Срок выполения заказа(L)"; //---"L"
            label6.Text = "Цена без скидкой(c1)"; //--------"c1"
            label7.Text = "Цена со скидки(c2)"; //--------"c2"
        }
    }
}

private void StatiticModels_Load(object sender, EventArgs e)
{

}

private void textBox1_TextChanged(object sender, EventArgs e)
{
    Text_char_chek(textBox1);
    Result_StatiticModels();
}

private void textBox2_TextChanged(object sender, EventArgs e)
{
    Text_char_chek(textBox2);
    Result_StatiticModels();
}
```

```
        private void textBox3_TextChanged(object sender, EventArgs e)
        {
            Text_char_chek(textBox3);
            Result_StatiticModels();
        }

        private void textBox4_TextChanged(object sender, EventArgs e)
        {
            Text_char_chek(textBox4);
            Result_StatiticModels();
        }
        private void textBox5_TextChanged(object sender, EventArgs e)
        {
            Text_char_chek(textBox5);
            Result_StatiticModels();
        }
        private void textBox6_TextChanged(object sender, EventArgs e)
        {
            Text_char_chek(textBox6);
            Result_StatiticModels();
        }
        private void textBox7_TextChanged(object sender, EventArgs e)
        {
            Text_char_chek(textBox7);
            Result_StatiticModels();
        }
        private void Result_StatiticModels()
        {
            if (comboBox1.SelectedIndex == 0)
            {
                richTextBox2.ReadOnly = false;
                richTextBox2.Clear();
                if (textBox1.Text != "" && textBox2.Text != "")
                {
                    richTextBox2.AppendText("Продолжительность цикла заказа: " +
Environment.NewLine);
                    richTextBox2.AppendText(Convert.ToString(Double.Parse(textBox1.Text)
* Double.Parse(textBox2.Text)) + Environment.NewLine);
                    richTextBox2.AppendText("Средний уровень запасов: " +
Environment.NewLine);
                    richTextBox2.AppendText(Convert.ToString(Double.Parse(textBox1.Text)
/ 2) + Environment.NewLine);

                    if (textBox3.Text != "" && textBox4.Text != "" && textBox4.Text !=
"0," && textBox4.Text != "0,0" && textBox4.Text != "0,00" && textBox4.Text != "0,00")
                    {
                        richTextBox2.AppendText("Суммарные затраты в единицу времени " +
Environment.NewLine);

richTextBox2.AppendText(Convert.ToString(Double.Parse(textBox3.Text) /
(Double.Parse(textBox1.Text) * Double.Parse(textBox2.Text)) + Double.Parse(textBox4.Text)
* (Double.Parse(textBox1.Text) / 2)) + Environment.NewLine);
                        richTextBox2.AppendText("Оптимальная стратегия управленя
запасами: " + Environment.NewLine);
                        richTextBox2.AppendText("Заказывать: " + Environment.NewLine);
                        double k = Math.Sqrt((2 * Double.Parse(textBox2.Text) *
Double.Parse(textBox3.Text)) / Double.Parse(textBox4.Text));
                        richTextBox2.AppendText(" " + k.ToString() + " единиц продукции"
+ Environment.NewLine);
                        richTextBox2.AppendText("Через каждые: " + Environment.NewLine);
                        richTextBox2.AppendText("" + Convert.ToString(k /
Double.Parse(textBox2.Text)) + " единиц времени");
                        richTextBox2.AppendText("(при условии что заявленный срок
выполения заказа меньше вычесленного)" + Environment.NewLine);
```

```
                        if (textBox5.Text != "" && Double.Parse(textBox5.Text) > (k /
Double.Parse(textBox2.Text)))
                        {
                            richTextBox2.AppendText("Иначе вычисляется другой эффективный
срок выполнения заказа, равный:" + Environment.NewLine);
                            int Count = 0;
                            for (int n = 0; n <
Convert.ToInt32(Double.Parse(textBox5.Text) / (k / Double.Parse(textBox2.Text))); n++)
                            {
                                if (Count < n)
                                {
                                    Count++;
                                }
                            }

richTextBox2.AppendText(Convert.ToString(Double.Parse(textBox5.Text) - (Count * (k /
Double.Parse(textBox2.Text)))) + Environment.NewLine);
                        }
                    }
                    richTextBox2.ReadOnly = true;
                }
            }
            else if (comboBox1.SelectedIndex == 1)
            {
                richTextBox2.Clear();
                double q, D, K, h, L, c1, c2, y_m, a, b, c, Q, optimal;

                double[] Q_mas;
                if (textBox1.Text != "" && textBox2.Text != "" && textBox3.Text != "" &&
textBox4.Text != "" && textBox4.Text != "0," && textBox4.Text != "0,0" && textBox4.Text
!= "0,00" && textBox4.Text != "0,00")
                {
                    D = Double.Parse(textBox2.Text);
                    K = Double.Parse(textBox3.Text);
                    h = Double.Parse(textBox4.Text);
                    q = Double.Parse(textBox1.Text);
                    y_m = Math.Sqrt((2 * K * D) / h);
                    if (y_m > q) { optimal = y_m; richTextBox2.AppendText("Оптимальный
объем заказа = " + y_m + "т.к y(m) > q и находиться в 1-ой зоне" + Environment.NewLine);
}
                    else
                    {
                        if (textBox6.Text != "" && textBox7.Text != "")
                        {
                            c1 = Double.Parse(textBox6.Text);
                            c2 = Double.Parse(textBox7.Text);
                            //(h*y_m)*Q^2 + (2D((c2-c1)y_m - K)-h*y_m)Q + 2k*D*y_m = 0 -
исходное уравнение
                            a = (h / 2);
                            b = c2 * D - c1 * D - ((K * D) / y_m) - ((h * y_m) / 2);
                            c = K * D;
                            Q_mas = AW.Solution_equation(a, b, c);
                            if (Q_mas[0] > y_m) { Q = Q_mas[0]; } else { Q = Q_mas[1]; }
                            richTextBox2.AppendText("Функция затрат подразделяется на
следующие 3-и зоны:" + Environment.NewLine);
                            richTextBox2.AppendText("1-ая зона: " + "[" + "0," +
y_m.ToString() + "]" + Environment.NewLine);
                            richTextBox2.AppendText("2-ая зона: " + "[" + y_m.ToString()
+ "," + Q.ToString() + "]" + Environment.NewLine);
                            richTextBox2.AppendText("3-ая зона: " + "[" + Q.ToString() +
",infinitude" + "]" + Environment.NewLine);
                            if (q > y_m && q < Q)
                            {
                                optimal = q;
```

```
                                richTextBox2.AppendText("Оптимальный объем заказа = " +
q.ToString() + "т.к q < Q и находиться во 2-ой зоне" + Environment.NewLine);
                            }
                            else { optimal = y_m; richTextBox2.AppendText("Оптимальный
объем заказа = " + y_m.ToString() + "т.к q > Q и находиться в 3-ей зоне" +
Environment.NewLine); }
                            if (textBox5.Text != "")
                            {
                                L = Double.Parse(textBox5.Text);
                                richTextBox2.AppendText("При заданном сроке выполнения
заказа в " + L.ToString() + "дней, точкой возобновления заказа ");
                                richTextBox2.AppendText("является " + L.ToString() + "*"
+ D.ToString() + " = " + (L*D).ToString() + " галоннов" + Environment.NewLine);
                                richTextBox2.AppendText("Следовательно оптимальная
стратегия управления управления запасами: " + Environment.NewLine);
                                richTextBox2.AppendText("Заказывать" + optimal.ToString()
+ "галоннов масла, когда уровень запаса понижается до" + (L * D).ToString() + "галоннов"
+ Environment.NewLine);
                            }
                        }

                    }
                }


            }
        }
        private void Text_char_chek(TextBox textB)
        {
             string strok = textB.Text;
             for (int i = 0; i < strok.Length; i++)
             {
                    if (strok[i] != '1' && strok[i] != '2' && strok[i] != '3' &&
strok[i] != '4' && strok[i] != '5'
                        && strok[i] != '6' && strok[i] != '7' && strok[i] != '8' &&
strok[i] != '9'
                        && strok[i] != '0' && strok[i] != ',')
                    {
                        textB.Text = textB.Text.Replace(strok[i].ToString(), "");
                        textB.SelectionStart = textB.Text.Length;
                    }
             }
        }
        private void Clear_all_elem_text()
        {
            richTextBox1.ReadOnly = false;
            richTextBox2.ReadOnly = false;
            richTextBox1.Clear();
            richTextBox2.Clear();
            richTextBox3.ReadOnly = false;
            richTextBox3.Clear();
            label1.Text = "";
            label2.Text = "";
            label3.Text = "";
            label4.Text = "";
            label5.Text = "";
            label7.Text = "";
            label6.Text = "";
        }

        private void label9_Click(object sender, EventArgs e)
        {

        }
```

```
        private void tabControl_statstic_SelectedIndexChanged(object sender, EventArgs e)
        {

        }
```

```
    }
}
```

Модуль №2.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace inventory_Management
{
    class any_works
    {
        public double[] Solution_equation(double a, double b, double c)
        {
            double[] mas = new double[2];
            double D, x1, x2;
            D = Math.Pow(b, 2) - (4 * a * c);
            x1 = (-b + Math.Sqrt(D)) / (2 * a);
            x2 = (-b - Math.Sqrt(D)) / (2 * a);
            mas[0] = x1;
            mas[1] = x2;
            return mas;
        }
    }
}
```

Модуль №3.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Xml;
using System.IO;

namespace inventory_Management
{
    class Work_xml
    {
        public List<string> get_text(string name)
        {
            List<string> ListText = new List<string>();
            XmlDocument XmlDoc = new XmlDocument();
            DirectoryInfo d = new DirectoryInfo(".");
            string path = d.Parent.Parent.FullName;
            XmlDoc.Load(path + @"\XML_BASE.xml");
            XmlNode nodeList = XmlDoc.SelectSingleNode("//"+name+"");
            XmlNode namList = nodeList.FirstChild;
            foreach (XmlNode N in namList)
            {

                ListText.Add(N.FirstChild.InnerText);
```

```
            }
            return ListText;

        }
        public List<string> get_img(string name)
        {
            List<string> ListText = new List<string>();
            XmlDocument XmlDoc = new XmlDocument();
            DirectoryInfo d = new DirectoryInfo(".");
            string path = d.Parent.Parent.FullName;
            XmlDoc.Load(path + @"\XML_BASE.xml");
            XmlNode nodeList = XmlDoc.SelectSingleNode("//" + name + "");
            XmlNode namList = nodeList["picture"];
            foreach (XmlNode N in namList)
            {

                ListText.Add(N.FirstChild.InnerText);

            }
            return ListText;
        }
        public List<string> get_task(string name)
        {
            List<string> ListText = new List<string>();
            XmlDocument XmlDoc = new XmlDocument();
            DirectoryInfo d = new DirectoryInfo(".");
            string path = d.Parent.Parent.FullName;
            XmlDoc.Load(path + @"\XML_BASE.xml");
            XmlNode nodeList = XmlDoc.SelectSingleNode("//" + name + "");
            XmlNode namList = nodeList["task"];
            foreach (XmlNode N in namList)
            {

                ListText.Add(N.FirstChild.InnerText);

            }
            return ListText;
        }
    }
}
```

Модуль №4.

```
using System;
using Microsoft.VisualStudio.TestTools.UnitTesting;
using inventory_Management;
using System.Collections.Generic;

namespace UnitTestProject1
{

    [TestClass]
    public class UnitTest1
    {
        //Получение списка картинок для первой задачи
        [TestMethod]
        public void Get_img_1_test()
        {
            Work_xml target = new Work_xml();
            List<string> actual = target.get_img("statisticModel_1");
            List<string> expected = new List<string> {
@"img\statistic_model_1_1.png",@"img\statistic_model_1_2.png",@"img\statistic_model_1_3.p
ng",@"img\statistic_model_1_4.png", @"img\statistic_model_1_5.png"};
            for (int i = 0; i < 5; i++)
            {
```

```
                Assert.AreEqual(expected[i], actual[i]);
            }

        }
        //Получение списка картинок для второй задачи
        [TestMethod]
        public void Get_img_2_test()
        {
            Work_xml target = new Work_xml();
            List<string> actual = target.get_img("statisticModel_2");
            List<string> expected = new List<string> { @"img\statistic_model_2_1.png",
@"img\statistic_model_2_2.png", @"img\statistic_model_2_3.png",
@"img\statistic_model_2_4.png"};
            for (int i = 0; i < 4; i++)
            {
                Assert.AreEqual(expected[i], actual[i]);
            }

        }
        //Получение условия задачи
        [TestMethod]
        public void Get_task_test()
        {
            Work_xml target = new Work_xml();
            List<string> actual = target.get_task("statisticModel_2");
            List<string> expected = new List<string> { "Автомобильная мастерская
специализируется на быстрой замене масла в автомобилях. Мастерская покупает автомобильное
масло в большом количестве по 3 доллара за галлон. Цена может быть снижена до 2.50
долларов за галлон при условии, что мастерская покупает более 1000 галлонов. За день в
мастерской обслуживается около 150 автомобилей, и на каждый из них для замены требуется
1.25 галлона масла. Мастерская хранит на складе большие объемы масла, что обходится в
0.02 доллара в день за один галлон. Стоимость размещения заказа на большой объем масла
равна 20 долларов. Срок выполнения заказа равен 2 дня. Требуется определить оптимальную
стратегию управления запасами." + Environment.NewLine +
"       "};
                Assert.AreEqual(expected[0], actual[0]);
        }
        //решение квадратного уравнения
        [TestMethod]
        public void Solution_equation_test()
        {
            any_works target = new any_works();
            double[] actual = target.Solution_equation(1, 4, 4);
            double[] expected = new double[] {-2,-2 };
            for (int i = 0; i < 2; i++)
            {
                Assert.AreEqual(expected[i], actual[i]);
            }

        }
    }
}
```

Задания для самостоятельной работы.

Вариант №1. Оптимизация капиталовложений.

Имеется 10 работ (A1 ¸ A2), каждая из которых характеризуется тремя технико-экономическими показателями:

AJ — трудозатраты;

Bj — размер, необходимых капиталовложений;

CJ — ожидаемый экономический эффект.

Исходные данные приведены в следующей таблице:

	A_1	A_2	A_3	A_4	A_5	A_6	A_7	A_8	A_9	A_{10}
a_j	3	3	3	4	4	2	4	3	6	5
b_j	4	3	2	4	6	4	3	5	3	4
c_j	3	7	5	6	8	4	7	4	7	6

Общие трудозатраты не должны превышать 20. Общий объем капиталовложений не должен превышать 20. Определите, какие из 10 работ следует выполнить, чтобы максимизировать ожидаемый экономический эффект, учитывая следующие условия взаимообусловленности и взаимоисключения:

а) $A_1 \rightarrow A_7 \vee A_4$; $A_7 \rightarrow A_{10} \vee A_2$ б) $A_3 \rightarrow A_5 \vee A_8 \vee A_9 \rightarrow A_6$

Вариант №2. Оптимизация производственной программы.

Автомобилестроительный завод выпускает три модели автомобилей, которые изготавливаются последовательно в трех цехах. Мощность цехов составляет 300, 250 и 200 человекодней в декаду. В первом цехе для сборки одного автомобиля первой модели требуется 6 человекодней, второй модели — 4 и третьей модели — 2 человекодня в декаду соответственно. Во втором цехе трудоемкость равна 3,4 и 5 человекодней соответственно, в третьем — по 3 человекодня на каждую модель. Прибыль, получаемая заводом от продажи одного автомобиля каждой модели, составляет соответственно 15, 13 и 10 тыс. долл.

Постройте модель для определения оптимального плана.

Вариант № 3. Двумерная задача раскроя.

Из минимального количества листов стекла размером 8 х 6 м2 требуется вырезать 10 оконных стекол размером 4 х 4 м2, 20 оконных стекол размером 4 х 5 м2 и 30 оконных стекол размером 3х3 м2. Множество вариантов раскроя (см. главу 3) показано в следующей таблице:

Размер, м² / Вариант	4×4	4×5	3×3
1	2	—	—
2	1	1	—
3	1	—	2
4	—	2	—
5	—	—	4
6	—	1	2

Постройте модель для определения плана раскроя, требующего минимального количества материала.

Вариант № 4. Задача о ранце.

Некая торговая компания имеет свои универсамы в Москве, Санкт-Петербурге, Нижнем Новгороде, Екатеринбурге, Самаре, Воронеже и Казани. В результате ошибок менеджмента экономическое положение компании стало ухудшаться, ей пришлось взять кредит в размере 13 млн руб. и в конечном счете, чтобы вовремя его погасить, срочно продавать некоторые из своих универсамов. Средства, которые компания могла бы получить от продажи универсамов в Москве, Санкт-Петербурге, Нижнем Новгороде, Екатеринбурге, Самаре, Воронеже или Казани, составляют соответственно 5,2; 4,9; 4,5; 3,6; 3,4; 3,2 и 3,1 млн руб. Однако продажа универсамов сопряжена с необходимостью увольнения персонала. Его численность составляет соответственно 200,190,180,170, 150,130 и 110 человек. По требованию объединенного профсоюза работников торговли компания должна минимизировать численность увольняемого персонала.

Постройте модель для нахождения оптимального решения.

Вариант №5. Руководство завода предполагает провести комплекс организационно-технических мероприятий в целях модернизации производства. Мероприятия потребуют следующих затрат производственных площадей, трудовых и финансовых ресурсов:

Мероприятие	Трудовые ресурсы, человекодни	Финансовые ресурсы, тыс. руб.	Производственные площади, м^2	Экономический эффект, тыс. руб.
Закупка станков с ЧПУ	350	400	130	13 000
Текущий ремонт	250	90	—	3000
Монтаж транспортного конвейера	100	60	300	8000
Установка рельсового крана	200	300	150	12 000
Ввод системы контроля качества	130	—	150	2500
Разработка АСУП	800	500	100	15 000

На реализацию всех мероприятий завод может выделить трудовых ресурсов 1300 человекодней, финансовых — 10 млн руб., производственных площадей — 700 м2.

Определите мероприятия, которые следует провести, располагая этими ресурсами, с тем чтобы общий экономический эффект был максимальным.

Вопросы:

1. Каков максимальный экономический эффект от проведения мероприятий?

2. Какое количество мероприятий следует провести?

Вариант №6. В текущем году заводу необходимо:

1) закупить два универсальных станка с ЧПУ общей стоимостью 200 тыс. руб. Для этого требуются трудовые ресурсы в объеме 250 человекодней и производственные площади 100 м2;

2) смонтировать транспортный конвейер стоимостью 100 тыс. руб. Необходимы трудовые ресурсы 190 человекодней и производственные площади 200 м2.

Для проведения этих мероприятий завод располагает финансовыми ресурсами 250 тыс. руб., трудовыми — 200 человекодней и производственными площадями 200 м2.

Недостаток средств и ресурсов можно компенсировать, проведя некоторые из следующих мероприятии:

1) внедрить новые резцы для обработки металла. Экономия трудозатрат — 130 человекодней, финансовые затраты — 50 тыс. руб.;

2) провести профилактический ремонт станочного парка. Трудозатраты — 10 человекодней, прибыль — 20 тыс. руб.;

3) внедрить систему контроля качества продукции. Экономия трудозатрат — 190 человекодней, затраты производственных площадей — 50 м2, прибыль — 5 тыс. руб.;

4) реализовать устаревшее оборудование. Трудозатраты — 60 человекодней, высвобождение производственных площадей — 200 м2, прибыль — 300 тыс. руб.;

5) провести инвентаризацию запасов материальных ресурсов. Трудозатраты — 20 человекодней, высвобождение производственных площадей — 150 м2.

Вопрос: Какое минимальное количество мероприятий следует провести, чтобы закупить станки с ЧПУ и смонтировать транспортный конвейер?

Вариант №7. В Сибири работают четыре химических завода. Они участвуют в конкурсе на размещение госзаказа по производству изделий пяти наименований в следующем объеме:

Изделие	1	2	3	4	5
Объем, шт.	350	250	400	150	150

Каждый завод представил несколько вариантов годовой производственной программы по выполнению госзаказа и соответствующие финансовые условия контракта:

	Варианты завода 1			Варианты завода 2		Варианты завода 3			Варианты завода 4	
	1	2	3	4	5	6	7	8	9	10
Изделие 1	100	200	200	50	80	—	—	100	100	50
Изделие 2	200	100	150	—	—	200	250	100	40	60
Изделие 3	300	250	250	120	100	100	50	50	60	100
Изделие 4	100	50	100	100	50	—	—	—	50	—
Изделие 5	50	100	80	—	—	100	100	80	150	100
Объем финансирования, млн руб.	12	16	14	7	9	16	15	17	5	8

Вопросы:

1. Каковы минимальные затраты на выполнение заказа?
2. Следует ли реализовать вариант 2 завода 1?

Вариант №8. Нефтеперерабатывающее предприятие использует в производстве нефть трех сортов. Резервные запасы нефти каждого сорта должны быть не меньше соответственно 20,40 и 60 тыс. т. Для хранения нефти могут быть использованы четыре резервуара вместимостью 25, 30, 35,40 тыс. т. Затраты на хранение 1 т нефти сорта 2 на 10% выше, чем затраты для нефти сорта 1, а для сорта 3 на 20% выше, чем для сорта 1. Смешение нефти разных сортов при хранении не допускается.

Вопросы:

1. Сколько резервуаров следует использовать?
2. Нефть какого сорта следует хранить в резервуаре вместимостью 30 тыс. т?

Вариант №9. Объединение кабельной промышленности состоит из трех заводов. Номенклатура выпускаемых изделий включает три позиции: кабель силовой, провод для осветительных установок и провод обмоточный. На трехлетний период планирования разработаны три варианта развития завода 1, два варианта развития завода 2 и один — завода 3. Производство кабельных изделий (в тыс. м) по годам приведено в следующей таблице:

		Кабель силовой			Провод для осветительных установок			Провод обмоточный			Затраты на 3 года, млн руб.
		Год 1	Год 2	Год 3	Год 1	Год 2	Год 3	Год 1	Год 2	Год 3	
Варианты завода 1	1	6,9	8,0	10,0	37	44	53	2,8	3,0	4,0	1557
	2	7,0	7,0	8,6	25	—	—	3,0	18,0	20,2	1399
	3	7,0	7,8	8,7	30	—	—	6,0	18,0	20,0	1034
Варианты завода 2	4	19,2	23,0	28,0	—	—	—	12,8	15,0	18,0	2822
	5	15,8	18,0	22,2	—	—	—	16,0	18,5	20,8	3044
Варианты завода 3	6	—	—	—	—	864	950	—	—	—	364
Потребность по годам, тыс. м		15	17	25	20	300	450	10	15	10	

Определите план выпуска продукции на трех заводах, обеспечивающий удовлетворение заданной потребности в кабельных изделиях с минимальными затратами.

Вопросы:

1. Каковы минимальные затраты?
2. Следует ли использовать вариант 3 для завода 1?

Вариант №10. В последующие два года добыча угля К2 должна возрасти на 180 и 234 тыс. т соответственно, а угля СС — на 150 и 195 тыс. т. Для обеспечения роста добычи могут быть введены в действие три шахты. Для каждой из них разработаны два варианта добычи угля. Для первого года с момента ввода шахты данные по объемам добычи (тыс. т) приведены в следующей таблице:

	Шахта 1		Шахта 2		Шахта 3	
	Вариант 1	Вариант 2	Вариант 1	Вариант 2	Вариант 1	Вариант 2
К2	80	120	30	50	60	40
СС	130	70	90	40	90	60
Затраты, тыс. руб.	100	120	50	40	70	50

На второй год с момента ввода:

На шахте 1 добыча угля К2 и СС выше по обоим вариантам на 10% при росте затрат на 10%;

На шахте 2 по первому варианту добыча К2 больше на 10%, а добыча СС меньше на 10% при неизменных затратах, по второму варианту добыча К2 меньше на 10%, а добыча СС больше на 10% при неизменных затратах;

На шахте 3 по обоим вариантам объем добычи и затраты те же, что и для первого года.

Любая шахта может быть введена как в первый, так и во второй год планового периода. Введенные мощности продолжают использоваться в последующие периоды времени.

Составьте план ввода мощностей по добыче угля, обеспечивающий выполнение плановых заданий с минимальными затратами.

Вопросы:

1. Каковы минимальные затраты?

2. Следует ли использовать вариант 1 развития шахты 2?

Вариант №11. Для реконструкции машиностроительного предприятия было представлено 10 проектов, каждый из которых характеризуется четырьмя агрегированными показателями: затратами труда, энергии, материалов, денежных средств, а также ежегодной прибылью в случае реализации проекта. Соответствующие данные и объем имеющихся ресурсов приведены в таблице:

Проект	1	2	3	4	5	6	7	8	9	10	Ресурсы
Труд, нормо-часы	50	60	30	40	80	70	50	20	40	50	300
Энергия, тыс. кВт·ч	4	4	2	5	5	2	3	6	6	3	24
Материалы, млн руб.	3	2	4	5	3	2	4	2	2	3	20
Денежные средства, млн руб.	7	5	9	6	4	3	7	2	4	5	30
Прибыль, млн руб.	9	8	8,5	8,8	9	8	9	8,7	8,9	8	

При выборе проектов необходимо учесть ряд ограничений технологического характера:

1) одновременно может быть реализовано не более семи проектов;

2) проекты 5 и 8 исключают друг друга;

3) проект 1 может быть реализован лишь при условии реализации проекта 2;

4) проект 4 может быть реализован лишь при условии реализации хотя бы одного из двух проектов: либо проекта 3, либо проекта 10.

Вопросы:

1. Какова максимальная прибыль?

2. Следует ли реализовывать проект 3?

Вариант №12. Имеются одинаковые заготовки, которые могут быть раскроены тремя способами и из которых могут быть получены не менее 10 деталей первого типоразмера, не менее 8 деталей второго типоразмера и не менее 10 деталей третьего типоразмера.

Способы раскроя представлены в следующей матрице:

$$A = \begin{bmatrix} 2 & 1 & 3 \\ 2 & 2 & 1 \\ 1 & 3 & 0 \end{bmatrix},$$

Где AIj — количество деталей типоразмера I, получаемое из одной заготовки путем ее раскроя способом J.

Количество заготовок, раскраиваемых каждым способом, должно быть целым и не превышать 4. Отходы от одной заготовки для каждого из способов раскроя составляют соответственно 4,5 и 5 см.

Выполните раскрой с минимальными суммарными отходами.

Вопросы:

1. Сколько заготовок должно быть раскроено вторым способом?
2. Чему равны минимальные суммарные отходы?

Вариант №13. Предприятию задана программа по изготовлению четырех видов изделий в количествах: вида А - 495, В - 265, С - 378, D - 162. На предприятии имеется три группы станков с различной производительностью. Задается суммарное допустимое время работы за этот период для каждой группы станков: первой - 80 ч., второй - 100 ч., третьей - 150 ч. Нормы времени (в часах) на изготовление одного изделия на каждом станке и данные об издержках (в рублях) на изготовление каждого изделия на станках различных групп приводятся в табл. 2.20. Требуется так распределить изготовление изделий по группам станков, чтобы была обеспечена заданная программа по изготовлению изделий и чтобы общие издержки были минимальны?

Группы станков	Нормы времени на станках, час				Издержки на изготовление единицы изделия, руб			
	1	2	3	4	A	B	C	D
1	0,5	0,3	0,4	0,1	0,12	0,25	0,3	0,4
11	0,4	0,2	0,2	0,5	0,15	0,15	0,4	0,2
111	0,4	0,1	0,3	0,6	0,18	0,35	0,5	0,1

Вариант №14. Предприятие должно выпустить по плану продукции А - 545 единиц, В - 367 единиц, С - 457 единиц на двух машинах. Каждая из двух машин может выполнить операции по производству всех трех видов продукции. Затраты времени на производстве единицы изделия каждой из двух машин приведены в табл. 2.19. Как распределить работу машин, при условии выпуска продукции пакетами по 100 шт. каждый, чтобы затраты времени на выполнение плана были минимальны?

Машины	Продукция		
	A	B	C
1	4,3	10,7	10,4
2	6,2	8,5	20,2

Вариант №15 Производство двух видов лесопродукции должно пройти три операции. Затраты времени на каждой операции на одно изделие, прибыль от реализации одного изделия в табл. 2.18. Сколько изделий

каждого вида должно произвести предприятие, чтобы получить максимум прибыли, причем число изделий А должно быть не менее 10, а В - не более 70 единиц.

Изделия	Затраты на одно изделие			Прибыль, руб
	1	2	3	
А В	11,3 6,1	7,2 8,3	16,1 9,5	25,5 38,3
Фонд времени на каждую операцию	600	700	1300	

Вариант №16. Леспромхоз имеет древесину трех видов в количествах: 1 - 1,12 тыс. м3, 2 - 0,52 тыс. м3, 3 - 0,73 тыс. м3, для изготовления изделий А, В, С и D. Нормы расхода древесины в м3 на изготовление единицы каждого изделия и прибыль от реализации единицы изделия даны в табл. 2.17. Определить, сколько изделий каждого вида должно произвести предприятие, чтобы общая прибыль от реализации всех изделий была максимальной?

Сырье	Нормы расхода сырья на единицу изделия			
	А	В	С	D
1 2 3	0,1 0,2 0,4	0,15 0,4 0,5	0,2 0,3 0,1	0,25 0,1 0,2
Прибыль, руб	13,4	24,2	33,7	11,1

Вариант №17. Фирма выпускает три продукта: А, В, С. На производство единицы продукта А требуется затратить 1,1 ч. труда ИТР, 12,3 ч. физического труда и 3,1 кг сырья. Для единицы продукта В соответствующие показатели равны 2,3 ч., 4,2 ч и 2,4 кг, для продукта С - 1,3 ч, 5,1 ч. и 1,2 кг. Ресурсы составляют 120 ч. труда ИТР, 640 ч. физического труда и 450 кг сырья. При оптовых закупках покупателю предоставляются скидки, так что прибыли от продажи продукции изменяются как показано в табл. 2.12. Например, если продается 120 ед. продукта А, то первые 40 ед. приносят по 63,2 руб. прибыли; следующие 60 - по 54,4 руб., а остальные 20 - по 48,3 руб. Сформулируйте задачу линейного программирования, решение которой определяет наиболее доходный производственный план.

Продукт А		Продукт В		Продукт С	
Прода-	Удельная	Прода-	Удельная	Прода-	Удельная

жа, ед.	прибыль, руб.	жа, ед.	прибыль, руб.	жа, ед.	прибыль, руб.
0-40	63,2	0-50	36,5	0-100	30,5
40-100	54,4	50-100	24,3	Более 100	24,8
100-150	48,3	Более 100	18,7	-	-
Более 150	42,1	-	-	-	-

Вариант №18. Лесхоз для кормления животных использует два вида корма. В дневном рационе животного должно содержаться не менее 6 единиц вещества А и 12 единиц В. Какое количество корма надо расходовать ежедневно на одного животного, чтобы затраты были минимальны.

Питательные вещества	Количество питательных веществ в 1 кг корма вида:	
	1	2
А В	2 2	1 4
Цена 1 кг корма, руб	2	3

Вариант №19. В леспромхозе производится раскряжевка хлыстов на сортименты. Требуется получить сортименты трех видов - длиной 6, 2,2 и 1,5 м. Длина среднего хлыста 31 м, средний диаметр 0,3 м. План поставки сортиментов, соответственно, 32,4 тыс. м3, 86,3 тыс м3 и 40,3 тыс. м3. Используя карту раскроя хлыстов без учета толщины пропила определить оптимальный план раскроя.

Сорти-	Варианты раскроя хлыстов										
мент, м	1	2	3	4	5	6	7	8	9	10	11
6 2,2 1,5	5 0 0	4 2 1	4 1 3	3 5 1	3 0 8	2 4 6	2 1 11	1 9 3	1 2 13	0 10 6	0 1 19
Отходы	1	1,1	0,3	0,5	1,0	1,2	0,3	0,7	1,1	0	0,3

Вариант № 20. Мебельное предприятие выпускает три вида наборов мебели, книжные полки и тумбу под телевизоры. Характеристики каждого вида продукции приведены в табл. 2.13. При условии получения максимальной прибыли объем товарной пилопродукции должен составить не менее 459,31 тыс. руб. Ситуация со сбытом продукции сложилась следующая. Книжными полками рынок насыщен поэтому торговые организации уменьшили объем договоров до 10 тыс. шт. Тумбы для телевизоров могут быть реализованы в объемах от 4 до 7 тыс. шт., наборы

мебели 2 - от 7 до 10 тыс. шт. Спрос на наборы мебели 1 и 3 неограничен и требуется не менее 10 тыс. шт. Предприятие имеет технологическое оборудование, число единиц которого и нормы затрат времени оборудования каждой группы на изготовление единицы каждого вида продукции приведены в табл. 2.13. Предприятие работает в две смены с эффективным временем работы каждой машины в 3945 ч. (коэффициент сменности 1,9). Оптимизировать производственную программу предприятия.

Показатель	Виды продукции				
	Набор мебели 1	Набор мебели 2	Набор мебели 3	Книж-ные полки	Тумба под теле-визор
Оптовая цена единицы изделия, тыс. руб	7,2	14,3	32,5	0,182	1,5
Прибыль от реализации, тыс. руб	2,4	4,5	60,3	0,06	0,45

Наименование оборудования	Число, шт.	Виды продукции				
		Набор мебели 1	Набор мебели 2	Набор мебели 3	Книжные полки	Тумба под телевизор
Линия раскроя древесно-стружечных плит	2	0,068	0,096	0,207	0,018	0,042
Гильотинные ножницы	1	0,045	0,080	0,158	0,011	0,035
Линия облицо-вывания	2	0,132	0,184	0,428	0,020	0,060
Линия обрезания кромок	2	0,057	0,082	0,230	0,010	0,028
Лаконаливная машина	2	0,063	0,090	0,217	0,010	0,032
Полироваль-ные станки	4	0,170	0,280	0,620	0,020	0,096

Список рекомендованной литературы

1. Вентцель Г.С. Исследование операций. − М.: Дрофа, 2006
2. Математические методы и модели исследования операций. − М.: Юнити, 2008.
3. Токарев В.В., Соколов А.В. Методы оптимальных решений. Общие положения. Математическое программирование. – М.: Физматлит, 2011 – 564 с.
4. Алексеев В.М. Сборник задач по оптимизации. – М.: Физматлит, 2011. – 256 с.
5. Карманов В.Г. Математическое программирование. – М.: Физматлит, 2010. – 264с.
6. Измаилов А.Ф., Солодов М.В. Численные методы оптимизации: учеб. пособие – 2-е изд., перераб. и доп. – М.: Физматлит, 2008. – 320с.
7. Балдин К.В., Брызгалов Н.А., Рукосуев А.В. Математическое программирование. – М.: Дашков и К°, 2010. – 220с.
8. Акулич И.Л. Математическое программирование в примерах и задачах: учебное пособие. – СПб.: Лань, 2009. – 352с.
9. Просветов Г.И. Методы оптимизации: задачи и решения. – М.: Альфа-Пресс, 2009. – 168с.
10. Ерзин А.И. Введение в исследование операций: учеб. пособие. Новосибирск: НГУ, 2006.
11. Гончаров Е.Н., Ерзин А.И., Залюбовский В.В. Исследование операций. Примеры и задачи: учеб. пособие. – Новосибирск: НГУ, 2005.
12. Гимади Э.Х., Глебов Н.И. Математические модели и методы принятия решений: учеб. пособие. – Новосибирск: НГУ, 2008. – 163 с.
13. Глухов В.В., Медников М.Д., Коробко С.Б. Математические методы и модели для менеджмента. - СПб.: Лань, 2000. – 480 с.
14. Акулич И.Л. Математическое программирование в примерах и задачах: учеб. пособие – 2-е изд., испр. и доп. – М.: Высш. шк.,1993. – 336 с.
15. Ашманов С.А. Линейное программирование. – М.: Наука, 1981.
16. Габасов Р., Кириллова Ф.М. Методы линейного программирования. Ч. 1. Общие задачи. – Минск: Изд-во БГУ им. В.И. Ленина, 1977. - 176 с.
17. Габасов Р., Кириллова Ф.М. Методы линейного программирования. Ч. 2. Транспортные задачи. – Минск: Изд-во БГУ им. В.И. Ленина, 1977. - 240 с.

МЕТОДЫ ОПТИМИЗАЦИИ

Методические указания к лабораторным работам

Составитель Янаева Марина Викторовна

Авторская правка

Компьютерная верстка М.В. Янаева