

Министерство образования Республики Беларусь

**Учреждение образования
«Гомельский государственный технический
университет имени П. О. Сухого»**

Кафедра «Информатика»

Н. С. Богданова

СИСТЕМНЫЙ АНАЛИЗ И ИССЛЕДОВАНИЕ ОПЕРАЦИЙ

ПРАКТИКУМ

**по выполнению лабораторных работ для студентов
специальности 1-40 04 01 «Информатика
и технологии программирования»
дневной формы обучения**

Гомель 2020

УДК 519.7+519.8(075.8)
ББК 32.972я73
Б73

*Рекомендовано научно-методическим советом
факультета автоматизированных и информационных систем
ГГТУ им. П. О. Сухого
(протокол № 10 от 03.06.2019 г.)*

Рецензент: зам. декана фак. автоматизированных и информационных систем
ГГТУ им. П. О. Сухого канд. физ.-мат. наук, доц. *О. В. Лукьяненко*

Богданова, Н. С.
Б73 Системный анализ и исследование операций : практикум по выполнению лаборатор. работ для студентов специальности 1-40 04 01 «Информатика и технологии программирования» днев. формы обучения / Н. С. Богданова. – Гомель : ГГТУ им. П. О. Сухого, 2020. – 70 с. – Систем. требования: PC не ниже Intel Celeron 300 МГц ; 32 Mb RAM ; свободное место на HDD 16 Mb ; Windows 98 и выше ; Adobe Acrobat Reader. – Режим доступа: <https://elib.gstu.by>. – Загл. с титул. экрана.

Практикум составлен в соответствии с учебной программой по дисциплине «Системный анализ и исследование операций» для студентов специальности 1-40 04 01 «Информатика и технологии программирования». Содержит 11 лабораторных работ для закрепления учебного материала по данной дисциплине. Рассмотрены современные подходы к разработке и нахождению оптимальных решений на основе математического моделирования.

Для студентов специальности 1-40 04 01 «Информатика и технологии программирования» дневной формы обучения.

УДК 519.7+519.8(075.8)
ББК 32.972я73

© Учреждение образования «Гомельский государственный технический университет имени П. О. Сухого», 2020

СОДЕРЖАНИЕ

ВВЕДЕНИЕ.....	4
ЛАБОРАТОРНАЯ РАБОТА №1	5
ПРОГРАММНАЯ РЕАЛИЗАЦИЯ РЕШЕНИЯ ЗАДАЧ ЦЕЛОЧИСЛЕННОГО ПРОГРАММИРОВАНИЯ	5
ЛАБОРАТОРНАЯ РАБОТА №2	9
ПРОГРАММНАЯ РЕАЛИЗАЦИЯ МЕТОДА ВЕТВЕЙ И ГРАНИЦ.....	9
ЛАБОРАТОРНАЯ РАБОТА №3	14
ПРОГРАММНАЯ РЕАЛИЗАЦИЯ РЕШЕНИЯ ЗАДАЧ ЦЕЛОЧИСЛЕННОГО ПРОГРАММИРОВАНИЯ МЕТОДОМ ГОМОРИ.....	14
ЛАБОРАТОРНАЯ РАБОТА №4	19
РЕШЕНИЕ ЗАДАЧИ РАСПРЕДЕЛЕНИЯ РЕСУРСОВ МЕТОДОМ ДИНАМИЧЕСКОГО ПРОГРАММИРОВАНИЯ.....	19
ЛАБОРАТОРНАЯ РАБОТА №5	25
СЕТЕВОЕ ПЛАНИРОВАНИЕ.....	25
ЛАБОРАТОРНАЯ РАБОТА №6	38
КОМПЬЮТЕРНАЯ РЕАЛИЗАЦИЯ МЕТОДА ПОТЕНЦИАЛОВ ДЛЯ РЕШЕНИЯ ЗАДАЧИ О ПОТОКЕ МИНИМАЛЬНОЙ СТОИМОСТИ	38
ЛАБОРАТОРНАЯ РАБОТА №7	42
ПОСТРОЕНИЕ ДЕРЕВА КРАТЧАЙШИХ ПУТЕЙ МЕТОДОМ ПОМЕТОК И АЛГОРИТМОМ ДЕЙКСТРЫ	42
ЛАБОРАТОРНАЯ РАБОТА №8	51
НАХОЖДЕНИЕ НА СЕТИ ПОТОКА МАКСИМАЛЬНОЙ ВЕЛИЧИНЫ.....	51
ЛАБОРАТОРНАЯ РАБОТА №9	55
РЕШЕНИЕ ЗАДАЧИ О НАЗНАЧЕНИЯХ ВЕНГЕРСКИМ МЕТОДОМ.....	55
ЛАБОРАТОРНАЯ РАБОТА №10	61
КОМПЬЮТЕРНАЯ РЕАЛИЗАЦИЯ АЛГОРИТМА ДЖОНСОНА.....	61
ЛАБОРАТОРНАЯ РАБОТА №11	65
ПРИНЦИП ГАРАНТИРОВАННОГО РЕЗУЛЬТАТА В МАТРИЧНЫХ ИГРАХ, РЕШЕНИЕ МАТРИЧНОЙ ИГРЫ В СМЕШАННЫХ СТРАТЕГИЯХ.	65
СПИСОК ИСТОЧНИКОВ.....	70

ВВЕДЕНИЕ

Учебная дисциплина «Системный анализ и исследование операций» - одна из базовых в программе подготовки студентов по специальности 1-40 04 01 «Информатика и технологии программирования», поскольку она содержит математические методы оптимизации различных систем и процессов, которыми на высоком уровне должен владеть современный специалист при решении конкретных практических задач в различных областях науки и техники.

Курс направлен на приобретение знаний теоретических основ методов системного анализа и исследования операций, получение навыков моделирования оптимизационных задач, овладение эффективными методами решения задач математического программирования, получение навыков по применению методов исследования операций и алгоритмов решения прикладных задач на высоком профессиональном уровне.

Практикум по выполнению лабораторных работ ориентирован на формирование у студентов представлений о математическом аппарате исследования операций, знакомит с принципами построения и анализа основных классов математических моделей исследования операций и их применением в практической деятельности.

Практикум предлагает для выполнения одиннадцать лабораторных работ. В каждой работе есть небольшая теоретическая часть, которая позволит студентам лучше справиться с поставленными задачами.

ЛАБОРАТОРНАЯ РАБОТА №1

ПРОГРАММНАЯ РЕАЛИЗАЦИЯ РЕШЕНИЯ ЗАДАЧ ЦЕЛОЧИСЛЕННОГО ПРОГРАММИРОВАНИЯ

Цель работы: получить практические навыки решения задач целочисленного программирования.

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Дискретное программирование - раздел матпрограммирования, изучающий экстремальные задачи, в которых значения переменных принимают конечное или счетное множество.

Целочисленное программирование (ЦП) является частным случаем дискретного. Задачи оптимизации, переменные которых принимают лишь целые значения, называются задачами целочисленного программирования.

В математической модели задачи ЦП как целевая функция, так и функции в системе ограничений могут быть линейными, нелинейными и смешанными. Ограничимся случаем, когда целевая функция и система ограничений задачи являются *линейными*.

Можно выделить следующие основные классы задач дискретного программирования (краткая классификация математических моделей дискретного программирования):

- 1) транспортная задача и ее модификации;
- 2) задачи с неделимостями (задача о рюкзаке);
- 3) экстремальные комбинаторные задачи (задача о коммивояжере, о покрытии, линейный и нелинейный варианты задач о назначениях);
- 4) модели с булевыми переменными (задача о назначениях);
- 5) задачи с разрывными целевыми функциями (транспортная задача с фиксированными доплатами);
- 6) задачи на неклассических областях.

В свою очередь, каждый тип моделей подразделяется на линейные и нелинейные, статические и динамические, детерминированные и стохастические и т. д.

Наиболее изучен класс задач ЦП детерминированного типа, в частности, детерминированная задача линейного ЦП.

Большинство целочисленных и комбинаторных типов задач, таких как задача с неделимостями, задача о коммивояжере, задача календарного планирования принадлежат к разряду так называемых «трудно решаемых задач». Это означает, что вычислительная сложность алгоритма их точного решения, т. е. зависимость числа элементарных операций (сложения или сравнения), необходимых для получения точного решения, от размерности

задачи n , является показательной (порядка 2^n), т. е. сравнимой с трудоемкостью полного перебора вариантов. В качестве n , например, для задачи с неделимостью служит число целочисленных переменных и число ограничений, для задачи коммивояжера - число городов или узлов графа маршрутов, для задачи календарного планирования - число деталей и число станков. Такие задачи называют еще NP -трудными или NP -полными. Получение их точного решения не может быть гарантировано, хотя для некоторых задач данного типа существуют эффективные методы, позволяющие находить точное решение даже при больших размерностях.

Задачи с вычислительной сложностью, определяемой полиномиальной зависимостью от n , называются «эффективно решаемыми задачами». К ним относятся задачи транспортного типа и линейные задачи о назначениях.

Общая постановка задачи ЦП отличается от общей постановки ЗЛП лишь наличием дополнительных ограничений – требований целочисленности: значения всех или части переменных модели в оптимальном решении должны быть целыми неотрицательными числами.

Если требование целочисленности распространяется на все переменные, то задачу ЦП называют полностью целочисленной задачей.

Если же требование целочисленности относится лишь к части переменных, то задачу называют частично целочисленной.

Если при этом целевая функция и функции, входящие в ограничения, линейные, то говорят, что данная задача является задачей линейного ЦП.

ЗЛП, которая отличается от рассматриваемой задачи ЦП лишь отсутствием требования целочисленности, называют задачей с ослабленными ограничениями, соответствующей задаче ЦП.

В общем виде математическая модель задачи целочисленного линейного программирования имеет вид:

$$z = \sum_{j=1}^n c_j x_j \rightarrow \max(\min)$$

при следующих ограничениях

$$\sum_{j=1}^n a_{ij} x_j \leq b_i, i = \overline{1, m}$$

$$x_j \geq 0, j = \overline{1, n}$$

$$x_j - \text{целые}, j = 1, 2, \dots, n_1 \leq n$$

Если $n_1 \leq n$, то задача называется частично целочисленной, если $n_1 = n$ – полностью целочисленной. Линейная функция называется целевой функцией. Множество планов X , удовлетворяющих системе ограничений – множеством допустимых решений Ω , $X \in \Omega$. Допустимый план $X \in \Omega$,

доставляющий целевой функции экстремальное значение, называется оптимальным планом.

Задание

- 1) Решить задачи целочисленного программирования (варианты см. в таблице 1.1); проверить правильность полученных решений средствами пакетов Mathcad или MS Excel.
- 2) Решить (см. общую постановку ЗЦП и таблицу вариантов ниже):
 - а) частично целочисленную задачу (x_1 – любое, x_2 – целочисленное);
 - б) полностью целочисленную задачу (x_1 и x_2 – целочисленные)

средствами пакетов Mathcad (графическим методом и с помощью блока решения) и MS Excel (с помощью надстройки «Поиск решения»). Сравнить полученные результаты.

$$z(x_1, x_2) = ax_1 + bx_2 \rightarrow \max$$

$$cx_1 + dx_2 \leq e$$

$$fx_1 + gx_2 \leq h$$

$$px_1 + rx_2 \geq q$$

$$x_1 \geq 0, \quad x_2 \geq 0$$

Таблица 1.1

Варианты заданий

Вариант	a	b	c	d	e	f	g	h	p	r	q
1	6	5	7	-8	3	-7	6	5	2	3	1
2	7	6	5	-8	3	-7	6	5	2	3	2
3	5	6	7	-7	3	-7	6	5	2	3	2
4	4	5	6	-7	2	-7	6	5	2	3	2
5	5	4	5	-9	2	-7	6	5	2	3	3
6	3	2	4	-5	2	-7	6	5	2	3	3
7	3	2	4	-5	2	-6	7	5	2	3	3
8	2	2	2	-4	1	-6	7	5	2	3	3
9	3	3	3	-5	3	-6	7	5	2	3	3
10	4	4	4	-6	3	-6	7	5	2	3	4
11	5	5	5	-2	4	-6	7	5	2	3	4
12	6	5	6	-3	4	-6	7	5	2	2	4
13	7	6	8	-5	4	-6	7	5	2	2	4
14	17	6	5	-5	4	-6	7	5	2	2	3
15	9	16	5	-5	3	-6	8	7	2	5	3
16	8	15	5	-5	3	-6	8	15	2	5	5
17	8	5	5	-8	3	-6	8	15	20	5	7

Окончание табл. 1.1

Вариант	a	b	c	d	e	f	g	h	p	r	q
18	8	5	5	-9	3	-6	8	15	20	5	9
19	8	-5	10	-9	3	-6	9	15	20	5	20
20	-6	8	10	-9	3	-6	9	15	20	5	20
21	-5	6	10	-9	3	-6	9	15	20	5	10
22	-3	9	10	-7	3	-6	9	15	20	5	10
23	-2	8	10	-7	3	-6	9	15	20	5	15
24	9	5	6	-7	3	-6	9	15	20	5	6
25	8	6	6	-7	3	-6	10	15	20	5	6
26	8	6	3	-3	4	-6	10	15	20	5	6
27	7	5	4	-1	4	-6	10	15	20	5	8
28	1	5	4	-1	10	-5	10	15	20	5	10
29	10	5	4	-1	10	-5	10	10	20	5	11
30	10	2	4	-10	12	-5	10	10	20	5	20

Требования к отчету

Отчет по лабораторной работе должен содержать:

- цель работы;
- номер варианта;
- исходные данные варианта;
- подробное описание выполнения всех заданий, включая таблицы с исходными данными задач, математические модели, копии экранов документов с подробными комментариями по выполняемым действиям;
- результаты вычислений;
- выводы.

Контрольные вопросы

1. Назвать основные классы задач дискретного программирования.
2. Общая постановка задачи целочисленного программирования.
3. Чем отличаются частично целочисленные от полностью целочисленных задач?
4. Математическая модель задачи целочисленного программирования.
5. Почему в задачах целочисленного программирования нельзя, в общем случае, получить целочисленное решение «округлением» нецелочисленного?

ЛАБОРАТОРНАЯ РАБОТА №2

ПРОГРАММНАЯ РЕАЛИЗАЦИЯ МЕТОДА ВЕТВЕЙ И ГРАНИЦ

Цель работы: получить практические навыки решения задач целочисленного программирования методом ветвей и границ.

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Метод ветвей и границ – один из комбинаторных методов. Его суть заключается в упорядоченном переборе вариантов и рассмотрении лишь тех из них, которые оказываются по определенным признакам перспективными, и отбрасывании бесперспективных вариантов.

Метод ветвей и границ состоит в следующем: множество допустимых решений (планов) некоторым способом разбивается на подмножества, каждое из которых этим же способом снова разбивается на подмножества. Процесс продолжается до тех пор, пока не получено оптимальное целочисленное решение исходной задачи.

Для задач линейного программирования решение ищется следующим образом.

Первоначально находим оптимальный план задачи без учета целочисленности переменных. Пусть им является план X_0 . Если среди компонент этого плана нет дробных чисел, то тем самым найдено искомое решение данной задачи и $F_{\max} = F(X_0)$. Если же среди компонент плана X_0 имеются дробные числа, то X_0 не удовлетворяет условию целочисленности и необходимо осуществить упорядоченный переход к новым планам, пока не будет найдено решение задачи. Покажем, как это можно сделать, предварительно отметив, что $F(X_0) \geq F(X)$ для всякого последующего плана X .

Предполагая, что найденный оптимальный план X_0 не удовлетворяет условию целочисленности переменных, тем самым считаем, что среди его компонент есть дробные числа. Пусть, например, переменная x_{i_0} приняла в плане X_0 дробное значение. Тогда в оптимальном целочисленном плане ее значение будет по крайней мере либо меньше или равно ближайшему меньшему целому числу K_{i_0} , либо больше или равно ближайшему большему целому числу $K_{i_0} + 1$. Определяя эти числа, находим решение двух задач линейного программирования:

$$(I) \begin{cases} F = \sum_{j=1}^n c_j x_j \rightarrow \max \\ \sum_{j=1}^n a_{tj} x_j = b_t, (t = \overline{1, m}) \\ x_{t0} \leq K_t \\ x_j \geq 0, (j = \overline{1, n}) \end{cases}$$

$$(II) \begin{cases} F = \sum_{j=1}^n c_j x_j \rightarrow \max \\ \sum_{j=1}^n a_{tj} x_j = b_t, (t = \overline{1, m}) \\ x_{t0} \geq K_t + 1 \\ x_j \geq 0, (j = \overline{1, n}) \end{cases}$$

Возможен один из следующих четырех случаев решения задач линейного программирования (I) и (II):

1. Одна из задач неразрешима, а другая имеет целочисленный оптимальный план. Тогда этот план и значение целевой функции на нем и дают решение исходной задачи.

2. Одна из задач неразрешима, а другая имеет оптимальный план, среди компонент которого есть дробные числа. Тогда рассматриваем вторую задачу и в ее оптимальном плане выбираем одну из компонент, значение которой равно дробному числу, и строим две задачи, аналогичные задачам (I) и (II).

3. Обе задачи разрешимы. Одна из задач имеет оптимальный целочисленный план, а в оптимальном плане другой задачи есть дробные числа. Тогда вычисляем значения целевой функции на этих планах и сравниваем их между собой. Если на целочисленном оптимальном плане значение целевой функции больше или равно ее значению на плане, среди компонент которого есть дробные числа, то данный целочисленный план является оптимальным для исходной задачи и он вместе со значением целевой функции на нем дает искомое решение. Если же значение целевой функции больше на плане, среди компонент которого есть дробные числа, то следует взять одно из таких чисел и для задачи, план которой рассматривается, необходимо построить две задачи, аналогичные (I) и (II).

4. Обе задачи разрешимы, и среди оптимальных планов обеих задач есть дробные числа. Тогда вычисляем значение целевой функции на данных оптимальных планах и рассматриваем ту из задач, для которой значение

целевой функции является наибольшим. В оптимальном плане этой задачи выбираем одну из компонент, значение которой является дробным числом, и строим две задачи, аналогичные (I) и (II).

Таким образом, описанный выше итерационный процесс может быть представлен в виде некоторого дерева, на котором исходная вершина отвечает оптимальному плану X_0 задачи, а каждая соединенная с ней ветвью вершина отвечает оптимальным планам задач (I) и (II). Каждая из этих вершин имеет свои ветвления. При этом на каждом шаге выбирается та вершина, для которой значение функции является наибольшим. Если на некотором шаге будет получен план, имеющий целочисленные компоненты, и значение функции на нем окажется больше или равно, чем значение функции в других возможных для ветвления вершинах, то данный план является оптимальным планом исходной задачи целочисленного программирования и значение целевой функции на нем является максимальным.

Итак, процесс нахождения решения задачи целочисленного программирования методом ветвей и границ включает следующие основные этапы:

1. Находят решение задачи линейного программирования без учета целочисленности.

2. Составляют дополнительные ограничения для одной из переменных, значение которой в оптимальном плане задачи является дробным числом.

3. Находят решение задач (I) и (II), которые получаются из предыдущей задачи в результате присоединения дополнительных ограничений.

4. В случае необходимости составляют дополнительные ограничения для переменной, значение которой является дробным, формулируют задачи, аналогичные задачам (I) и (II), и находят их решение. Итерационный процесс продолжают до тех пор, пока не будет найдена вершина, соответствующая целочисленному плану задачи и такая, что значение функции в этой вершине больше или равно значению функции в других возможных для ветвления вершинах.

Задание

С помощью средств Visual Studio применяя метод ветвей и границ, решить задачу коммивояжера и построить полное дерево ветвления с матрицей расстояний, заданной таблицей. Варианты смотри в таблице 2.1.

Таблица 2.1

Варианты заданий

Вар-т	Задание	Вар-т	Задание																																																																																						
1	<table border="1"> <thead> <tr> <th colspan="2" rowspan="2"></th> <th colspan="5">До города</th> </tr> <tr> <th>1</th> <th>2</th> <th>3</th> <th>4</th> <th>5</th> </tr> </thead> <tbody> <tr> <th rowspan="5">От города</th> <th>1</th> <td>–</td> <td>48</td> <td>27</td> <td>31</td> <td>43</td> </tr> <tr> <th>2</th> <td>33</td> <td>–</td> <td>28</td> <td>44</td> <td>43</td> </tr> <tr> <th>3</th> <td>41</td> <td>28</td> <td>–</td> <td>40</td> <td>36</td> </tr> <tr> <th>4</th> <td>37</td> <td>35</td> <td>29</td> <td>–</td> <td>46</td> </tr> <tr> <th>5</th> <td>48</td> <td>48</td> <td>25</td> <td>29</td> <td>–</td> </tr> </tbody> </table>			До города					1	2	3	4	5	От города	1	–	48	27	31	43	2	33	–	28	44	43	3	41	28	–	40	36	4	37	35	29	–	46	5	48	48	25	29	–	2	<table border="1"> <thead> <tr> <th colspan="2" rowspan="2"></th> <th colspan="4">До города</th> </tr> <tr> <th>1</th> <th>2</th> <th>3</th> <th>4</th> </tr> </thead> <tbody> <tr> <th rowspan="4">От города</th> <th>1</th> <td>–</td> <td>27</td> <td>44</td> <td>28</td> </tr> <tr> <th>2</th> <td>47</td> <td>–</td> <td>43</td> <td>36</td> </tr> <tr> <th>3</th> <td>29</td> <td>26</td> <td>–</td> <td>29</td> </tr> <tr> <th>4</th> <td>34</td> <td>35</td> <td>34</td> <td>–</td> </tr> </tbody> </table>			До города				1	2	3	4	От города	1	–	27	44	28	2	47	–	43	36	3	29	26	–	29	4	34	35	34	–												
				До города																																																																																					
		1	2	3	4	5																																																																																			
От города	1	–	48	27	31	43																																																																																			
	2	33	–	28	44	43																																																																																			
	3	41	28	–	40	36																																																																																			
	4	37	35	29	–	46																																																																																			
	5	48	48	25	29	–																																																																																			
		До города																																																																																							
		1	2	3	4																																																																																				
От города	1	–	27	44	28																																																																																				
	2	47	–	43	36																																																																																				
	3	29	26	–	29																																																																																				
	4	34	35	34	–																																																																																				
3	<table border="1"> <thead> <tr> <th colspan="2" rowspan="2"></th> <th colspan="5">До города</th> </tr> <tr> <th>1</th> <th>2</th> <th>3</th> <th>4</th> <th>5</th> </tr> </thead> <tbody> <tr> <th rowspan="5">От города</th> <th>1</th> <td>–</td> <td>32</td> <td>37</td> <td>37</td> <td>29</td> </tr> <tr> <th>2</th> <td>40</td> <td>–</td> <td>40</td> <td>28</td> <td>31</td> </tr> <tr> <th>3</th> <td>41</td> <td>34</td> <td>–</td> <td>34</td> <td>37</td> </tr> <tr> <th>4</th> <td>36</td> <td>37</td> <td>33</td> <td>–</td> <td>27</td> </tr> <tr> <th>5</th> <td>31</td> <td>39</td> <td>33</td> <td>38</td> <td>–</td> </tr> </tbody> </table>			До города					1	2	3	4	5	От города	1	–	32	37	37	29	2	40	–	40	28	31	3	41	34	–	34	37	4	36	37	33	–	27	5	31	39	33	38	–	4	<table border="1"> <thead> <tr> <th colspan="2" rowspan="2"></th> <th colspan="5">До города</th> </tr> <tr> <th>1</th> <th>2</th> <th>3</th> <th>4</th> <th>5</th> </tr> </thead> <tbody> <tr> <th rowspan="5">От города</th> <th>1</th> <td>–</td> <td>48</td> <td>27</td> <td>31</td> <td>43</td> </tr> <tr> <th>2</th> <td>33</td> <td>–</td> <td>28</td> <td>44</td> <td>43</td> </tr> <tr> <th>3</th> <td>41</td> <td>28</td> <td>–</td> <td>40</td> <td>36</td> </tr> <tr> <th>4</th> <td>37</td> <td>35</td> <td>29</td> <td>–</td> <td>46</td> </tr> <tr> <th>5</th> <td>48</td> <td>48</td> <td>25</td> <td>29</td> <td>–</td> </tr> </tbody> </table>			До города					1	2	3	4	5	От города	1	–	48	27	31	43	2	33	–	28	44	43	3	41	28	–	40	36	4	37	35	29	–	46	5	48	48	25	29	–
				До города																																																																																					
		1	2	3	4	5																																																																																			
От города	1	–	32	37	37	29																																																																																			
	2	40	–	40	28	31																																																																																			
	3	41	34	–	34	37																																																																																			
	4	36	37	33	–	27																																																																																			
	5	31	39	33	38	–																																																																																			
		До города																																																																																							
		1	2	3	4	5																																																																																			
От города	1	–	48	27	31	43																																																																																			
	2	33	–	28	44	43																																																																																			
	3	41	28	–	40	36																																																																																			
	4	37	35	29	–	46																																																																																			
	5	48	48	25	29	–																																																																																			
5	<table border="1"> <thead> <tr> <th colspan="2" rowspan="2"></th> <th colspan="4">До города</th> </tr> <tr> <th>1</th> <th>2</th> <th>3</th> <th>4</th> </tr> </thead> <tbody> <tr> <th rowspan="4">От города</th> <th>1</th> <td>–</td> <td>27</td> <td>44</td> <td>28</td> </tr> <tr> <th>2</th> <td>47</td> <td>–</td> <td>43</td> <td>36</td> </tr> <tr> <th>3</th> <td>29</td> <td>26</td> <td>–</td> <td>29</td> </tr> <tr> <th>4</th> <td>34</td> <td>35</td> <td>34</td> <td>–</td> </tr> </tbody> </table>			До города				1	2	3	4	От города	1	–	27	44	28	2	47	–	43	36	3	29	26	–	29	4	34	35	34	–	6	<table border="1"> <thead> <tr> <th colspan="2" rowspan="2"></th> <th colspan="5">До города</th> </tr> <tr> <th>1</th> <th>2</th> <th>3</th> <th>4</th> <th>5</th> </tr> </thead> <tbody> <tr> <th rowspan="5">От города</th> <th>1</th> <td>–</td> <td>32</td> <td>37</td> <td>37</td> <td>29</td> </tr> <tr> <th>2</th> <td>40</td> <td>–</td> <td>40</td> <td>28</td> <td>31</td> </tr> <tr> <th>3</th> <td>41</td> <td>34</td> <td>–</td> <td>34</td> <td>37</td> </tr> <tr> <th>4</th> <td>36</td> <td>37</td> <td>33</td> <td>–</td> <td>27</td> </tr> <tr> <th>5</th> <td>31</td> <td>39</td> <td>33</td> <td>38</td> <td>–</td> </tr> </tbody> </table>			До города					1	2	3	4	5	От города	1	–	32	37	37	29	2	40	–	40	28	31	3	41	34	–	34	37	4	36	37	33	–	27	5	31	39	33	38	–												
				До города																																																																																					
		1	2	3	4																																																																																				
От города	1	–	27	44	28																																																																																				
	2	47	–	43	36																																																																																				
	3	29	26	–	29																																																																																				
	4	34	35	34	–																																																																																				
		До города																																																																																							
		1	2	3	4	5																																																																																			
От города	1	–	32	37	37	29																																																																																			
	2	40	–	40	28	31																																																																																			
	3	41	34	–	34	37																																																																																			
	4	36	37	33	–	27																																																																																			
	5	31	39	33	38	–																																																																																			
7	<table border="1"> <thead> <tr> <th colspan="2" rowspan="2"></th> <th colspan="5">До города</th> </tr> <tr> <th>1</th> <th>2</th> <th>3</th> <th>4</th> <th>5</th> </tr> </thead> <tbody> <tr> <th rowspan="5">От города</th> <th>1</th> <td>–</td> <td>48</td> <td>27</td> <td>31</td> <td>43</td> </tr> <tr> <th>2</th> <td>33</td> <td>–</td> <td>28</td> <td>44</td> <td>43</td> </tr> <tr> <th>3</th> <td>41</td> <td>28</td> <td>–</td> <td>40</td> <td>36</td> </tr> <tr> <th>4</th> <td>37</td> <td>35</td> <td>29</td> <td>–</td> <td>46</td> </tr> <tr> <th>5</th> <td>48</td> <td>48</td> <td>25</td> <td>29</td> <td>–</td> </tr> </tbody> </table>			До города					1	2	3	4	5	От города	1	–	48	27	31	43	2	33	–	28	44	43	3	41	28	–	40	36	4	37	35	29	–	46	5	48	48	25	29	–	8	<table border="1"> <thead> <tr> <th colspan="2" rowspan="2"></th> <th colspan="4">До города</th> </tr> <tr> <th>1</th> <th>2</th> <th>3</th> <th>4</th> </tr> </thead> <tbody> <tr> <th rowspan="4">От города</th> <th>1</th> <td>–</td> <td>27</td> <td>44</td> <td>28</td> </tr> <tr> <th>2</th> <td>47</td> <td>–</td> <td>43</td> <td>36</td> </tr> <tr> <th>3</th> <td>29</td> <td>26</td> <td>–</td> <td>29</td> </tr> <tr> <th>4</th> <td>34</td> <td>35</td> <td>34</td> <td>–</td> </tr> </tbody> </table>			До города				1	2	3	4	От города	1	–	27	44	28	2	47	–	43	36	3	29	26	–	29	4	34	35	34	–												
				До города																																																																																					
		1	2	3	4	5																																																																																			
От города	1	–	48	27	31	43																																																																																			
	2	33	–	28	44	43																																																																																			
	3	41	28	–	40	36																																																																																			
	4	37	35	29	–	46																																																																																			
	5	48	48	25	29	–																																																																																			
		До города																																																																																							
		1	2	3	4																																																																																				
От города	1	–	27	44	28																																																																																				
	2	47	–	43	36																																																																																				
	3	29	26	–	29																																																																																				
	4	34	35	34	–																																																																																				
9	<table border="1"> <thead> <tr> <th colspan="2" rowspan="2"></th> <th colspan="5">До города</th> </tr> <tr> <th>1</th> <th>2</th> <th>3</th> <th>4</th> <th>5</th> </tr> </thead> <tbody> <tr> <th rowspan="5">От города</th> <th>1</th> <td>–</td> <td>32</td> <td>37</td> <td>37</td> <td>29</td> </tr> <tr> <th>2</th> <td>40</td> <td>–</td> <td>40</td> <td>28</td> <td>31</td> </tr> <tr> <th>3</th> <td>41</td> <td>34</td> <td>–</td> <td>34</td> <td>37</td> </tr> <tr> <th>4</th> <td>36</td> <td>37</td> <td>33</td> <td>–</td> <td>27</td> </tr> <tr> <th>5</th> <td>31</td> <td>39</td> <td>33</td> <td>38</td> <td>–</td> </tr> </tbody> </table>			До города					1	2	3	4	5	От города	1	–	32	37	37	29	2	40	–	40	28	31	3	41	34	–	34	37	4	36	37	33	–	27	5	31	39	33	38	–	10	<table border="1"> <thead> <tr> <th colspan="2" rowspan="2"></th> <th colspan="5">До города</th> </tr> <tr> <th>1</th> <th>2</th> <th>3</th> <th>4</th> <th>5</th> </tr> </thead> <tbody> <tr> <th rowspan="5">От города</th> <th>1</th> <td>–</td> <td>48</td> <td>27</td> <td>31</td> <td>43</td> </tr> <tr> <th>2</th> <td>33</td> <td>–</td> <td>28</td> <td>44</td> <td>43</td> </tr> <tr> <th>3</th> <td>41</td> <td>28</td> <td>–</td> <td>40</td> <td>36</td> </tr> <tr> <th>4</th> <td>37</td> <td>35</td> <td>29</td> <td>–</td> <td>46</td> </tr> <tr> <th>5</th> <td>48</td> <td>48</td> <td>25</td> <td>29</td> <td>–</td> </tr> </tbody> </table>			До города					1	2	3	4	5	От города	1	–	48	27	31	43	2	33	–	28	44	43	3	41	28	–	40	36	4	37	35	29	–	46	5	48	48	25	29	–
				До города																																																																																					
		1	2	3	4	5																																																																																			
От города	1	–	32	37	37	29																																																																																			
	2	40	–	40	28	31																																																																																			
	3	41	34	–	34	37																																																																																			
	4	36	37	33	–	27																																																																																			
	5	31	39	33	38	–																																																																																			
		До города																																																																																							
		1	2	3	4	5																																																																																			
От города	1	–	48	27	31	43																																																																																			
	2	33	–	28	44	43																																																																																			
	3	41	28	–	40	36																																																																																			
	4	37	35	29	–	46																																																																																			
	5	48	48	25	29	–																																																																																			

Окончание табл. 2.1

Вар-т	Задание	Вар-т	Задание																																																																										
11	<table border="1"> <thead> <tr> <th colspan="2" rowspan="2"></th> <th colspan="4">До города</th> </tr> <tr> <th>1</th> <th>2</th> <th>3</th> <th>4</th> </tr> </thead> <tbody> <tr> <th rowspan="4">От города</th> <th>1</th> <td>–</td> <td>27</td> <td>44</td> <td>28</td> </tr> <tr> <th>2</th> <td>47</td> <td>–</td> <td>43</td> <td>36</td> </tr> <tr> <th>3</th> <td>29</td> <td>26</td> <td>–</td> <td>29</td> </tr> <tr> <th>4</th> <td>34</td> <td>35</td> <td>34</td> <td>–</td> </tr> </tbody> </table>			До города				1	2	3	4	От города	1	–	27	44	28	2	47	–	43	36	3	29	26	–	29	4	34	35	34	–	12	<table border="1"> <thead> <tr> <th colspan="2" rowspan="2"></th> <th colspan="5">До города</th> </tr> <tr> <th>1</th> <th>2</th> <th>3</th> <th>4</th> <th>5</th> </tr> </thead> <tbody> <tr> <th rowspan="5">От города</th> <th>1</th> <td>–</td> <td>32</td> <td>37</td> <td>37</td> <td>29</td> </tr> <tr> <th>2</th> <td>40</td> <td>–</td> <td>40</td> <td>28</td> <td>31</td> </tr> <tr> <th>3</th> <td>41</td> <td>34</td> <td>–</td> <td>34</td> <td>37</td> </tr> <tr> <th>4</th> <td>36</td> <td>37</td> <td>33</td> <td>–</td> <td>27</td> </tr> <tr> <th>5</th> <td>31</td> <td>39</td> <td>33</td> <td>38</td> <td>–</td> </tr> </tbody> </table>			До города					1	2	3	4	5	От города	1	–	32	37	37	29	2	40	–	40	28	31	3	41	34	–	34	37	4	36	37	33	–	27	5	31	39	33	38	–
				До города																																																																									
		1	2	3	4																																																																								
От города	1	–	27	44	28																																																																								
	2	47	–	43	36																																																																								
	3	29	26	–	29																																																																								
	4	34	35	34	–																																																																								
		До города																																																																											
		1	2	3	4	5																																																																							
От города	1	–	32	37	37	29																																																																							
	2	40	–	40	28	31																																																																							
	3	41	34	–	34	37																																																																							
	4	36	37	33	–	27																																																																							
	5	31	39	33	38	–																																																																							
13	<table border="1"> <thead> <tr> <th colspan="2" rowspan="2"></th> <th colspan="5">До города</th> </tr> <tr> <th>1</th> <th>2</th> <th>3</th> <th>4</th> <th>5</th> </tr> </thead> <tbody> <tr> <th rowspan="5">От города</th> <th>1</th> <td>–</td> <td>32</td> <td>37</td> <td>37</td> <td>29</td> </tr> <tr> <th>2</th> <td>40</td> <td>–</td> <td>40</td> <td>28</td> <td>31</td> </tr> <tr> <th>3</th> <td>41</td> <td>34</td> <td>–</td> <td>34</td> <td>37</td> </tr> <tr> <th>4</th> <td>36</td> <td>37</td> <td>33</td> <td>–</td> <td>27</td> </tr> <tr> <th>5</th> <td>31</td> <td>39</td> <td>33</td> <td>38</td> <td>–</td> </tr> </tbody> </table>			До города					1	2	3	4	5	От города	1	–	32	37	37	29	2	40	–	40	28	31	3	41	34	–	34	37	4	36	37	33	–	27	5	31	39	33	38	–	14	<table border="1"> <thead> <tr> <th colspan="2" rowspan="2"></th> <th colspan="4">До города</th> </tr> <tr> <th>1</th> <th>2</th> <th>3</th> <th>4</th> </tr> </thead> <tbody> <tr> <th rowspan="4">От города</th> <th>1</th> <td>–</td> <td>27</td> <td>44</td> <td>28</td> </tr> <tr> <th>2</th> <td>47</td> <td>–</td> <td>43</td> <td>36</td> </tr> <tr> <th>3</th> <td>29</td> <td>26</td> <td>–</td> <td>29</td> </tr> <tr> <th>4</th> <td>34</td> <td>35</td> <td>34</td> <td>–</td> </tr> </tbody> </table>			До города				1	2	3	4	От города	1	–	27	44	28	2	47	–	43	36	3	29	26	–	29	4	34	35	34	–
				До города																																																																									
		1	2	3	4	5																																																																							
От города	1	–	32	37	37	29																																																																							
	2	40	–	40	28	31																																																																							
	3	41	34	–	34	37																																																																							
	4	36	37	33	–	27																																																																							
	5	31	39	33	38	–																																																																							
		До города																																																																											
		1	2	3	4																																																																								
От города	1	–	27	44	28																																																																								
	2	47	–	43	36																																																																								
	3	29	26	–	29																																																																								
	4	34	35	34	–																																																																								
15	<table border="1"> <thead> <tr> <th colspan="2" rowspan="2"></th> <th colspan="4">До города</th> </tr> <tr> <th>1</th> <th>2</th> <th>3</th> <th>4</th> </tr> </thead> <tbody> <tr> <th rowspan="4">От города</th> <th>1</th> <td>–</td> <td>27</td> <td>44</td> <td>28</td> </tr> <tr> <th>2</th> <td>47</td> <td>–</td> <td>43</td> <td>36</td> </tr> <tr> <th>3</th> <td>29</td> <td>26</td> <td>–</td> <td>29</td> </tr> <tr> <th>4</th> <td>34</td> <td>35</td> <td>34</td> <td>–</td> </tr> </tbody> </table>			До города				1	2	3	4	От города	1	–	27	44	28	2	47	–	43	36	3	29	26	–	29	4	34	35	34	–	<table border="1"> <thead> <tr> <th colspan="2" rowspan="2"></th> <th colspan="5">До города</th> </tr> <tr> <th>1</th> <th>2</th> <th>3</th> <th>4</th> <th>5</th> </tr> </thead> <tbody> <tr> <th rowspan="5">От города</th> <th>1</th> <td>–</td> <td>32</td> <td>37</td> <td>37</td> <td>29</td> </tr> <tr> <th>2</th> <td>40</td> <td>–</td> <td>40</td> <td>28</td> <td>31</td> </tr> <tr> <th>3</th> <td>41</td> <td>34</td> <td>–</td> <td>34</td> <td>37</td> </tr> <tr> <th>4</th> <td>36</td> <td>37</td> <td>33</td> <td>–</td> <td>27</td> </tr> <tr> <th>5</th> <td>31</td> <td>39</td> <td>33</td> <td>38</td> <td>–</td> </tr> </tbody> </table>			До города					1	2	3	4	5	От города	1	–	32	37	37	29	2	40	–	40	28	31	3	41	34	–	34	37	4	36	37	33	–	27	5	31	39	33	38	–	
				До города																																																																									
		1	2	3	4																																																																								
От города	1	–	27	44	28																																																																								
	2	47	–	43	36																																																																								
	3	29	26	–	29																																																																								
	4	34	35	34	–																																																																								
		До города																																																																											
		1	2	3	4	5																																																																							
От города	1	–	32	37	37	29																																																																							
	2	40	–	40	28	31																																																																							
	3	41	34	–	34	37																																																																							
	4	36	37	33	–	27																																																																							
	5	31	39	33	38	–																																																																							

Требования к отчету

Отчет по лабораторной работе должен содержать:

- цель работы;
- номер варианта;
- исходные данные варианта;
- подробное описание выполнения всех заданий, включая таблицы с исходными данными задач, математические модели, копии экранов документов с подробными комментариями по выполняемым действиям (листинг программы и полученные результаты), полное дерево ветвления;
- выводы.

Контрольные вопросы

1. Назвать основные классы задач дискретного программирования.
2. Общая постановка задачи целочисленного программирования.

3. Метод ветвей и границ.
4. Дайте определение терминам: «ветвление» и «оценивание».
5. Виды стратегий ветвления. Дать описание.
6. Постановка задачи о коммивояжере.
7. Алгоритм решения задачи о коммивояжере методом ветвей и границ.

ЛАБОРАТОРНАЯ РАБОТА №3

ПРОГРАММНАЯ РЕАЛИЗАЦИЯ РЕШЕНИЯ ЗАДАЧ ЦЕЛОЧИСЛЕННОГО ПРОГРАММИРОВАНИЯ МЕТОДОМ ГОМОРИ

Цель работы: получить практические навыки решения задач целочисленного программирования с помощью метода Гомори и графического метода.

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Постановка задачи: найти $x \in R^n$ такое, что $z = c'x \rightarrow \max$ при ограничениях $Ax = b, x \geq 0, x_i - \text{целые числа}, i = \overline{1, n}, b \in R^m, c \in R^n, A - (m \times n) - \text{матрица}$.

Идея метода. Сначала решается соответствующая непрерывная задача (условия целочисленности не учитываются) симплекс-методом. Если полученный оптимальный план целочислен, то он будет решением исходной задачи. Если нет, то по одной из дробных компонент этого плана строится дополнительное ограничение (сечение Гомори), которое "отсекает" его от множества планов, а все целочисленные остаются, непрерывная задача с дополнительным ограничением снова решается. При этом используется результат решения предыдущей задачи (последняя симплекс таблица) и ее удобно решать двойственным симплекс-методом. Если новый оптимальный план целочислен, то получено решение исходной задачи, в противном случае операция (построения сечения Гомори) повторяется. И т.д. Доказано, что метод Гомори через конечное число таких операций позволяет построить решение задачи.

Пример. Пусть мы имеем задачу целочисленного линейного программирования:

$$z = x_1 + x_2 \rightarrow \max$$

$$2x_1 + x_2 + x_3 = 6$$

$$2x_1 + 3x_2 + x_4 = 9, x_i \geq 0, i = \overline{1, n}$$

x_i – целые числа.

Применим к непрерывной задаче симплекс-метод. Получим оптимальный план:

$$x^0 = \left(\frac{9}{4}, \frac{3}{2}, 0, 0 \right), z_{\max} = 3\frac{3}{4}.$$

Обозначим:

$[\alpha]$ – целая часть числа

$\{\alpha\} = \alpha - [\alpha]$ – дробная часть числа α .

Последняя симплекс-таблица имеет вид:

Таблица 3.1

Симплекс-таблица

а _б	б	а ₁	а ₂	а ₃	а ₄
а ₁	9/4	1	0	3/4	-1/4
а ₂	3/2	0	1	-1/2	1/2
Δ	15/4	0	0	1/4	1/4

Так как план X^0 не целочисленен применим к задаче метод Гомори.

1. Из b_2 выбираем число с наибольшей дробной частью. Так как

$\max \left\{ \left\{ \frac{9}{4} \right\} = \frac{1}{4}; \left\{ \frac{3}{2} \right\} = \frac{1}{2} \right\} = \frac{1}{2}$, то по II строке строим отсечение Гомори по формуле:

$$\{b_i\} - \sum_{j \in I_n} \{a_{ij}\} x_{ij} \leq 0$$

имеем

$$\left\{ \frac{3}{2} \right\} - \left(\left\{ -\frac{1}{2} \right\} x_3 + \left\{ \frac{1}{2} \right\} x_4 \right) \leq 0$$

$$\frac{1}{2} - \left(\frac{1}{2} x_3 + \frac{1}{2} x_4 \right) \leq 0; -\frac{1}{2} x_3 - \frac{1}{2} x_4 \leq -\frac{1}{2}$$

Приводим это ограничение к равенству введя дополнительную переменную $x_5 \geq 0$.

$$-\frac{1}{2} x_3 - \frac{1}{2} x_4 + x_5 = -\frac{1}{2}$$

Так как $b_5 = -\frac{1}{2}$, значит строим двойственную симплекс-таблицу и решаем полученную задачу двойственным симплекс-методом ($\delta = \Delta$).

Таблица 3.2

Двойственная симплекс-таблица

аБ	b	а ₁	а ₂	а ₃	а ₄	а ₅
а ₁	9/4	1	0	3/4	-1/4	0
а ₂	3/2	0	1	-1/2	1/2	0
а ₅	-1/2	0	0	-1/2	-1/2	1
Δ	15/4	0	0	1/4	1/4	0
δ				1/2	1/2	
а ₁	3/2	1	0	0	-1	3/2
а ₂	2	0	1	0	1	-1
а ₃	1	0	0	1	1	-2
Δ	7/2	0	0	0	0	1/2

3. Строим сечение Гомори для I строки:

$$\left\{ \frac{3}{2} \right\} - \left(\{-1\} x_4 + \left\{ \frac{3}{2} \right\} x_5 \right) \leq 0$$

$$\frac{1}{2} - \left(0^* x_4 + \frac{1}{2} x_5 \right) \leq 0, \frac{1}{2} - \frac{1}{2} x_5 \leq 0; -\frac{1}{2} + x_5 + x_6 = -\frac{1}{2}$$

4. Строим новую двойственную симплекс-таблицу.

Таблица 3.3

Новая двойственная симплекс-таблица

аБ	b	а ₁	а ₂	а ₃	а ₄	а ₅	а ₆
а ₁	3/2	1	0	0	-1	3/2	0
а ₂	2	0	1	0	1	-1	0
а ₃	1	0	0	1	1	-2	0
а ₆	-1/2	0	0	0	0	-1/2	1
Δ	7/2	0	0	0	0	1/2	0
а ₁	0						
а ₂	3						
а ₃	3						
а ₅	1						
Δ	3	0	0	0	0	0	1

$$x^* = (0, 3, 3, 0), z_{\max} = 3.$$

Задание

Решить задачу целочисленного программирования (по своему варианту – см. варианты ниже) методом Гомори средствами Visual Studio.

Таблица 3.4

Варианты к заданию

Вар-т	Задание	Вар-т	Задание
1	$\max \{3x_1 + 2x_2\}$ $\begin{cases} x_1 \geq 0, \\ x_2 \geq 0, \\ 13x_1 - 3x_2 = 52, \\ x_1 + 2x_2 = 11, \\ -4x_1 + 4x_2 = 16, \\ x_1, x_2 \in \mathbb{Z} \end{cases}$	2	$\max \{x_1 + 2x_2\}$ $\begin{cases} 3x_1 + 7x_2 \leq 21, \\ -x_1 + 3x_2 \leq 10, \\ x_1, x_2 \in \mathbb{Z}, x_1, x_2 \geq 0 \end{cases}$
3	$\max \{x_1 + 2x_2\}$ $\begin{cases} 5x_1 + 7x_2 \leq 21, \\ -x_1 + 3x_2 \leq 8, \\ x_1, x_2 \in \mathbb{Z}, x_1, x_2 \geq 0 \end{cases}$	4	$\max \{x_1 + x_2\}$ $\begin{cases} x_1 \geq 0, \\ x_2 \geq 0, \\ -2x_1 + 3x_2 \leq 4, \\ 3x_1 - 2x_2 \leq 1, \\ x_1, x_2 \in \mathbb{Z} \end{cases}$
5	$\max \{x_1 + 2x_2\}$ $\begin{cases} x_1 \geq 0, \\ x_2 \geq 0, \\ -2x_1 + x_2 \leq 4, \\ 3x_1 - 2x_2 \leq 1, \\ x_1, x_2 \in \mathbb{Z} \end{cases}$	6	$\max \{3x_1 + 2x_2\}$ $\begin{cases} x_1 \geq 0, \\ x_2 \geq 0, \\ 13x_1 - 4x_2 = 52, \\ x_1 + 2x_2 = 11, \\ -x_1 + 4x_2 = 16, \\ x_1, x_2 \in \mathbb{Z} \end{cases}$
7	$\max \{x_1 + 2x_2\}$ $\begin{cases} 5x_1 + 3x_2 \leq 21, \\ -x_1 + 3x_2 \leq 10, \\ x_1, x_2 \in \mathbb{Z}, x_1, x_2 \geq 0 \end{cases}$	8	$\max \{x_1 + x_2\}$ $\begin{cases} x_1 \geq 0, \\ x_2 \geq 0, \\ -2x_1 + 3x_2 \leq 4, \\ 3x_1 - 2x_2 \leq 1, \\ x_1, x_2 \in \mathbb{Z} \end{cases}$

Вар-т	Задание	Вар-т	Задание
9	$\max \{2x_1 + x_2\}$ $\begin{cases} x_1 \geq 0, \\ x_2 \geq 0, \\ -2x_1 + 3x_2 \leq 5, \\ 3x_1 - 2x_2 \leq 1, \\ x_1, x_2 \in Z \end{cases}$	10	$\max \{x_1 + 4x_2\}$ $\begin{cases} 5x_1 + 7x_2 \leq 21, \\ -x_1 + 3x_2 \leq 8, \\ x_1, x_2 \in Z, x_1, x_2 \geq 0 \end{cases}$
11	$\max \{x_1 + 2x_2\}$ $\begin{cases} x_1 \geq 0, \\ x_2 \geq 0, \\ 13x_1 - 2x_2 = 52, \\ x_1 + 2x_2 = 14, \\ -x_1 + 4x_2 = 16, \\ x_1, x_2 \in Z \end{cases}$	12	$\max \{x_1 + x_2\}$ $\begin{cases} x_1 \geq 0, \\ x_2 \geq 0, \\ -x_1 + 2x_2 \leq 4, \\ x_1 - 2x_2 \leq 1, \\ x_1, x_2 \in Z \end{cases}$
13	$\max \{x_1 + x_2\}$ $\begin{cases} x_1 \geq 0, \\ x_2 \geq 0, \\ -x_1 + 2x_2 \leq 4, \\ x_1 - 3x_2 \leq 1, \\ x_1, x_2 \in Z \end{cases}$	14	$\max \{x_1 + 2x_2\}$ $\begin{cases} x_1 \geq 0, \\ x_2 \geq 0, \\ 13x_1 - x_2 = 40, \\ x_1 + 2x_2 = 14, \\ -x_1 + 4x_2 = 16, \\ x_1, x_2 \in Z \end{cases}$
15	$\max \{x_1 + 2x_2\}$ $\begin{cases} 5x_1 + 10x_2 \leq 21, \\ -x_1 + 3x_2 \leq 8, \\ x_1, x_2 \in Z, x_1, x_2 \geq 0 \end{cases}$	16	$\max \{3x_1 + 2x_2\}$ $\begin{cases} x_1 \geq 0, \\ x_2 \geq 0, \\ 12x_1 - 2x_2 = 51, \\ x_1 + 2x_2 = 11, \\ -x_1 + 4x_2 = 16, \\ x_1, x_2 \in Z \end{cases}$

Требования к отчету

Отчет по лабораторной работе должен содержать:

- цель работы;
- номер варианта;
- исходные данные варианта;

- подробное описание выполнения всех заданий, включая таблицы с исходными данными задач, математические модели, копии экранов документов с подробными комментариями по выполняемым действиям (листинг программы и полученные результаты;
- выводы.

Контрольные вопросы

1. Назвать основные классы задач дискретного программирования.
2. Общая постановка задачи целочисленного программирования.
3. В чем заключается общая идея методов отсечения при решении целочисленных задач?
4. Метод отсечения Гомори.
5. Почему в задачах целочисленного программирования нельзя, в общем случае, получить целочисленное решение «округлением» нецелочисленного?
6. Как решать графическим методом задачи целочисленного программирования?

ЛАБОРАТОРНАЯ РАБОТА №4

РЕШЕНИЕ ЗАДАЧИ РАСПРЕДЕЛЕНИЯ РЕСУРСОВ МЕТОДОМ ДИНАМИЧЕСКОГО ПРОГРАММИРОВАНИЯ

Цель работы: получить практические навыки решения задачи распределения ресурсов методом динамического программирования.

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Задача распределения ресурсов имеет вид:

$$\sum_{i=1}^n f_i(x_i) \rightarrow \max$$

$$\sum_{i=1}^n x_i = C, \quad ,$$

$$x_i \geq 0, i = \overline{1, n}$$

где n - число технологических процессов, между которыми нужно распределить сырье в объеме C , с целью получения максимальной прибыли;

x_i – количество сырья, выделяемое на i -й процесс;

$f_i(x_i)$ – прибыль получаемая в i -ом процессе при использовании в нем сырья в объеме x_i , $i = \overline{1, n}$.

Так как целевая функция и ограничения в задаче сепарабельны, то к ней применим метод динамического программирования.

Пусть C – целое число и x_i , $i = \overline{1, n}$ – могут принимать лишь целые неотрицательные значения. В этом случае для задачи удобно использовать следующую табличную реализацию метода динамического программирования.

Таблица 4.1

Табличная реализация метода динамического программирования

$B(y)$ \ y	1	2	...	k	...	C
$B_1(y)$	$B_1(1)$	$B_1(2)$...	$B_1(k)$...	$B_1(C)$
$B_2(y)$	$B_2(1)$ [$x_2(1)$]	$B_2(2)$ [$x_2(2)$]	...	$B_2(k)$ [$x_2(k)$]	...	$B_2(C)$ [$x_2(C)$]
...
$B_n(y)$	$B_n(1)$ [$x_n(1)$]	$B_n(2)$ [$x_n(2)$]	...	$B_n(k)$ [$x_n(k)$]	...	$B_n(C)$ [$x_n(C)$]

Рассчитываем функцию Беллмана $B_k(y)$ путем последовательного решения уравнения Беллмана

$$B_k(y) = \max_{0 \leq z \leq y} [f_k(z) + B_{k-1}(y-z)]$$

$$0 \leq y \leq C$$

с начальным условием

$$B_1(y) = f_1(y), 0 \leq y \leq C.$$

Данные заносим в таблицу, имеющую $c+1$ столбцов и $n+1$ строк.

При заполнении таблицы в ее клетке вместе со значениями $B_k(y)$, $1 \leq k \leq n, 1 \leq y \leq n$ записывается также (в квадратных скобках) и величину $x_k(y)$, определяемое как то число z , $0 \leq z \leq y$, на котором в уравнении Беллмана достигается максимум (если таких чисел несколько, записываем все).

1. Определяем оптимальное значение целевой функции задачи. Максимальная прибыль равна величине $B_n(C)$.

2. Определяем оптимальное распределение сырья между процессами x_i , $i = \overline{1, n}$.

Из таблицы последовательно находим

$$x_n^0 = x_n(C),$$

$$x_n^0 = x_i \left(C - \sum_{j=n}^1 x_j^0 \right), i = n-1, \dots, 2; x_i^0 = C - \sum_{j=n}^2 x_j^0$$

Решений $x_i^0 = \{x_i^0, i = \overline{1, n}\}$ может оказаться несколько.

Замечание: Описанную выше табличную реализацию удобно применять в случае изменения условий задачи: при сокращении или при упрощении либо количества сырья C , либо чисел n технологических процессов. При сокращении чисел C или n для подсчета решения новой задачи рассматривается соответствующая часть таблицы, получаемая путем вычеркивания ненулевых столбцов или строк. При увеличении C или n для получения решения новой задачи таблицу следует нарастить.

Задание

Записать задачу в виде задачи распределения ресурсов.

1. Решить задачу распределения ресурсов при $C=6, n=3$
2. Найти по таблице решение в случае сокращения ресурсов на одну единицу, $C=5$.
3. Эффективно ли введение еще одного дополнительного технологического процесса f_4 ?

Номер варианта выбирается по таблице ниже.

Таблица 4.2

Варианты к заданию

Номер по порядку i	f_1	f_2	f_3	f_4
1-10	A_i	B_i	B_i	B_i+1
11-17	$A_i - 10$	$B_i - 9$	$B_i - 8$	$B_i - 8$
18-24	$A_i - 17$	$B_i - 15$	$B_i - 16$	$B_i - 14$
25-28	$A_i - 24$	$B_i - 21$	$B_i - 20$	$B_i - 20$

i – порядковый номер студента по журналу.

Варианты функций прибыли см. в таблице ниже $f(x)=0, x=0$:

Таблица 4.3

Варианты к заданию для А

Вар-т	Задание	Вар-т	Задание
1	$\begin{cases} x^2 - 1, x = 1, 2, 3 \\ 2x + 1, x = 4, 5, 6 \end{cases}$	2	$\begin{cases} 4 + x^2 - 2x, x = 1, 2, 3 \\ 2x + 3, x = 4, 5, 6 \end{cases}$
3	$\begin{cases} 4 + x^2 - 2x, x = 1, 2, 3 \\ x + 4, x = 4, 5, 6 \end{cases}$	4	$\begin{cases} \frac{x^2}{2} + 2x, x = 1, 2, 3 \\ x + 6, x = 4, 5, 6 \end{cases}$
5	$\begin{cases} 3^{x-2}, x = 1, 2, 3 \\ 2^x - 2, x = 4, 5, 6 \end{cases}$	6	$\begin{cases} 2x - 2, x = 1, 2, 3 \\ 2^{x-3} + 3, x = 4, 5, 6 \end{cases}$
7	$\begin{cases} x^2, x = 1, 2, 3 \\ 9, x = 4, 5, 6 \end{cases}$	8	$\begin{cases} \frac{x^3}{3} + 1, x = 1, 2, 3 \\ x + 2, x = 4, 5, 6 \end{cases}$
9	$\begin{cases} 2x + 1, x = 1, 2, 3 \\ \frac{x^2}{4} + 3, x = 4, 5, 6 \end{cases}$	10	$\begin{cases} 3x, x = 1, 2, 3 \\ 10, x = 4, 5, 6 \end{cases}$
11	$\begin{cases} 4 + x^2 - 2x, x = 1, 2, 3 \\ 2x + 3, x = 4, 5, 6 \end{cases}$	12	$\begin{cases} \frac{x^2}{2} + 2x, x = 1, 2, 3 \\ 2x - 6, x = 4, 5, 6 \end{cases}$
13	$\begin{cases} \frac{x^3}{3} + 1, x = 1, 2, 3 \\ 3x + 8, x = 4, 5, 6 \end{cases}$	14	$\begin{cases} x^2, x = 1, 2, 3 \\ x - 1, x = 4, 5, 6 \end{cases}$
15	$\begin{cases} \frac{x^2}{4} + x, x = 1, 2, 3 \\ x + 2, x = 4, 5, 6 \end{cases}$	16	$\begin{cases} 3^{x-2}, x = 1, 2, 3 \\ 2^x + 1, x = 4, 5, 6 \end{cases}$

Таблица 4.4

Варианты к заданию для Б

Вар-т	Задание	Вар-т	Задание
1	$\frac{1}{2}(x^2 - 3x)$	2	$\frac{1}{18}(x^3 - x^2)$
3	$0,3(2^x - (x-1)^2)$	4	$2(7x - x^2)$
5	$2x$	6	$\frac{x^2}{4} + \frac{x}{2}$
7	$\frac{1}{2}x$	8	$0,2(x^2 + 3x)$
9	$0,05(3^{x-1} - \frac{(x-1)^2}{3})$	10	$0,3(x^2 + 2x)$
11	$x^2 + 3x$	12	$2^x - (x-1)^2$
13	$\frac{1}{18}(x^3 - x^2)$	14	$0,01(3^{x-1} - \frac{(x-1)^2}{3})$
15	$x^2 + 2x$	16	$3^{x-1} - \frac{(x-1)^2}{3}$

Таблица 4.5

Варианты к заданию для В

№ \ x	1	2	3	4	5	6
1	0	0	1	2	4	8
2	2	3	4	5	8	11
3	2	2	4	5	6	9
4	2	4	6	8	8	8
5	0	4	5	5	9	10
6	1	1	4	6	8	11
7	2	3	5	7	9	10
8	1	3	5	7	9	10
9	0	2	4	6	8	12
10	0	0	3	5	7	9
11	0	0	1	2	4	8
12	2	3	4	5	8	11
13	2	2	4	5	6	9
14	2	4	6	8	8	8
15	0	4	5	5	9	10
16	1	1	4	6	8	11

Требования к отчету

Отчет по лабораторной работе должен содержать:

- цель работы;
- номер варианта;
- исходные данные варианта;
- подробное описание выполнения задания 1 – 3, включая таблицы с исходными данными задач, математические модели, копии экранов документов с подробными комментариями по выполняемым действиям (листинг кода, полученные результаты);
- выводы.

Контрольные вопросы

1. Понятие динамическое программирование.
2. Основные этапы решения задачи.
3. Идея метода динамического программирования.
4. Принцип оптимальности Беллмана.

ЛАБОРАТОРНАЯ РАБОТА №5

СЕТЕВОЕ ПЛАНИРОВАНИЕ

Цель работы: получить практические навыки решения задач сетевого планирования.

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Сетевое планирование и управление (СПУ) – это комплекс графических и расчетных методов, организационных мероприятий, обеспечивающих моделирование, анализ и динамическую перестройку плана выполнения сложных проектов и разработок.

Сетевое планирование и управление включает три основных этапа:

1. Структурное планирование.
2. Календарное планирование.
3. Оперативное управление.

Структурное планирование начинается с разбиения проекта на четко определенные операции, для которых определяется продолжительность. Затем строится сетевой график, который представляет взаимосвязи работ проекта. Это позволяет детально анализировать все работы и вносить улучшения в структуру проекта еще до начала его реализации.

Календарное планирование предусматривает построение календарного графика, определяющего моменты начала и окончания каждой работы и другие временные характеристики сетевого графика. Это позволяет, в частности, выявлять критические операции, которым необходимо уделять особое внимание, чтобы закончить проект в директивный срок. Во время календарного планирования определяются временные характеристики всех работ с целью проведения в дальнейшем *оптимизации* сетевой модели, которая позволит улучшить эффективность использования какого-либо ресурса.

В ходе *оперативного управления* используются сетевой и календарный графики для составления периодических отчетов о ходе выполнения проекта. При этом сетевая модель может подвергаться оперативной корректировке, вследствие чего будет разрабатываться новый календарный план остальной части проекта.

Основными понятиями сетевых моделей являются: сетевая модель, работа, событие.

Сетевая модель – графическое изображение последовательности и связи событий, которые представляют собой результат одной или нескольких работ.

Работа – это некоторый процесс, приводящий к достижению определенного результата, требующий затрат каких-либо ресурсов и имеющий протяженность во времени.

По количеству затрачиваемого времени работа может быть:

- *действительной*, т.е. требующей затрат времени;
- *фиктивной*, т.е. формально не требующей затрат времени и представляющей связь между какими-либо работами.

Событие – это момент времени, когда завершаются одни работы и начинаются другие.

На сетевой модели (сетевом графике) работы изображаются *стрелками*, которые соединяют *вершины*, изображающие события. Начало и окончание любой работы описываются парой событий, которые называются *начальным* и *конечным* событиями. Поэтому для идентификации конкретной работы используют код работы (i, j) , состоящий из номеров начального (i -го) и конечного (j -го) событий (см. рис. 5.1).

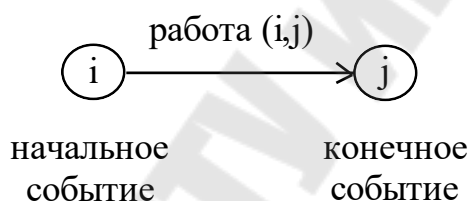


Рис. 5.1 Кодирование работы

Событие, не имеющее предшествующих ему событий, т.е. с которого начинается проект, называют *исходным*. Событие, которое не имеет последующих событий и отражает конечную цель проекта, называется *завершающим*.

Путь – это любая последовательность работ в сетевом графике (в частном случае это одна работа), в которой конечное событие одной работы совпадает с начальным событием следующей за ней работы. Различают следующие виды путей.

Полный путь – это путь от исходного до завершающего события. *Критический путь* – максимальный по продолжительности полный путь. Работы, лежащие на критическом пути, называют *критическими*. *Подкритический путь* – полный путь, ближайший по длительности к критическому пути.

Применение методов СПУ, в конечном счете, должно обеспечить получение календарного плана, определяющего сроки начала и окончания каждой операции. Построение сети является лишь первым шагом на пути к

достижению этой цели. Вторым шагом является расчет сетевой модели, который выполняют прямо на сетевом графике.

К временным параметрам событий относятся:

- $T_p(i)$ – ранний срок наступления события i . Это время, которое необходимо для выполнения всех работ, предшествующих данному событию i . Оно равно наибольшей из продолжительности путей, предшествующих данному событию.

- $T_n(i)$ – поздний срок наступления события i . Это такое время наступления события i , превышение которого вызовет аналогичную задержку наступления завершающего события сети. Поздний срок наступления любого события i равен разности между продолжительностью критического пути и наибольшей из продолжительностей путей, следующих за событием i .

- $R(i)$ – резерв времени наступления события i . Это такой промежуток времени, на который может быть отсрочено наступление события i без нарушения сроков завершения проекта в целом. Начальные и конечные события критических работ имеют нулевые резервы событий.

Рассчитанные численные значения временных параметров записываются прямо в вершины сетевого графика (см. рис. 5.2).

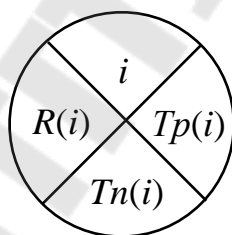


Рис. 5.2 Отображение временных параметров событий в вершинах сетевого графика

Расчет ранних сроков свершения событий $T_p(i)$ ведется от исходного к завершающему событию.

1. Для исходного события $T_p(C) = 0$.

2. Для всех остальных событий i $T_p(i) = \max[T_p(k) + t(k,i)]$, где максимум берется по всем работам (k,i) , входящим в событие i .

Поздние сроки свершения событий $T_n(i)$ рассчитываются от завершающего к исходному событию.

3. Для завершающего события Z $T_n(Z) = T_p(Z)$.

4. Для всех остальных событий $T_n(i) = \min[T_n(j) - t(i, j)]$, где минимум берется по всем работам (i, j) , выходящим из события i .

5. $R(i) = T_n(i) - T_p(i)$.

К наиболее важным временным параметрам работ относятся:

1. $T_{рн}(i, j)$ - ранний срок начала работы.
2. $T_{нн}(i, j)$ - поздний срок начала работы.
3. $T_{ро}(i, j)$ - ранний срок окончания работы.
4. $T_{но}(i, j)$ - поздний срок окончания работы.

Для критических работ $T_{рн}(i, j) = T_{нн}(i, j)$ и $T_{ро}(i, j) = T_{но}(i, j)$.

5. $R_n(i, j)$ - полный резерв работы показывает максимальное время, на которое может быть увеличена продолжительность работы (i, j) или отсрочено ее начало, чтобы продолжительность проходящего через нее максимального пути не превысила продолжительности критического пути. Важнейшее свойство полного резерва работы (i, j) заключается в том, что его частичное или полное использование уменьшает полный резерв у работ, лежащих с работой (i, j) на одном пути. Таким образом, полный резерв принадлежит не одной данной работе (i, j) , а всем работам, лежащим на путях, проходящим через эту работу.

6. $R_C(i, j)$ - свободный резерв работы показывает максимальное время, на которое можно увеличить продолжительность работы (i, j) или отсрочить ее начало, не меняя ранних сроков начала последующих работ. Использование свободного резерва одной из работ не меняет величины свободных резервов остальных работ сети.

Временные параметры работ сети определяются на основе ранних и поздних сроков событий.

1. $T_{рн}(i, j) = T_p(i)$.
2. $T_{ро}(i, j) = T_p(i) + t(i, j)$ или $T_{ро}(i, j) = T_{рн}(i, j) + t(i, j)$.
3. $T_{но}(i, j) = T_n(j)$.
4. $T_{нн}(i, j) = T_n(j) - t(i, j)$ или $T_{но}(i, j) = T_{нн}(i, j) - t(i, j)$.
5. $R_n(i, j) = T_n(j) - T_p(i) - t(i, j)$.
6. $R_C(i, j) = T_p(j) - T_p(i) - t(i, j)$.

Временные параметры работ вносятся в таблицу. При этом коды работ записывают в определенном порядке: сначала записываются все

работы, выходящие из исходного, т.е. первого, события, затем - выходящие из второго события, потом - из третьего и т.д.

Различие в использовании свободных и полных резервов заключается в том, что при сдвиге работы, с использованием свободного резерва, моменты начала следующих за ней работ остаются неизменными (т.е. последующие работы не сдвигаются). При сдвиге работы, с использованием полного резерва, все последующие работы сдвигаются.

Временные параметры работ вносятся в таблицу. При этом коды работ записывают в определенном порядке: сначала записываются все работы, выходящие из исходного, т.е. первого, события, затем - выходящие из второго события, потом - из третьего и т.д.

При оптимизации использования ресурса рабочей силы чаще всего сетевые работы стремятся организовать таким образом, чтобы:

- количество одновременно занятых исполнителей было минимальным;
- выровнять потребность в людских ресурсах на протяжении срока выполнения проекта.

Суть оптимизации загрузки сетевых моделей по критерию «минимум исполнителей» заключается в следующем: необходимо таким образом организовать выполнения сетевых работ, чтобы количество одновременно работающих исполнителей было минимальным. Для проведения подобных видов оптимизации необходимо построить и проанализировать *график привязки* и *график загрузки*.

График привязки отображает взаимосвязь выполняемых работ во времени и строится на основе данных либо о продолжительности работ (в данной лабораторной это T), либо о ранних сроках начала и окончания работ. При первом способе построения необходимо помнить, что работа (i, j) может начать выполняться только после того как будут выполнены все предшествующие ей работы (k, j) . По вертикальной оси графика привязки откладываются коды работ, по горизонтальной оси – длительность работ (раннее начало и раннее окончание работ).

На *графике загрузки* по горизонтальной оси откладывается время, например в днях, по вертикальной - количество человек, занятых работой в каждый конкретный день. Для построения графика загрузки необходимо:

- на графике привязки над каждой работой написать количество ее исполнителей;
- подсчитать количество работающих в каждый день исполнителей и отложить на графике загрузки.

Для удобства построения и анализа графики загрузки и привязки следует располагать один над другим.

Описанные виды оптимизации загрузки выполняются за счет сдвига во времени не критических работ, т.е. работ, имеющих полный и/или свободный резервы времени. Полный и свободный резервы любой работы можно определить без специальных расчетов, анализируя только график привязки. Сдвиг работы означает, что она будет выполняться уже в *другие дни* (т.е. изменится время ее начала и окончания), что в свою очередь приведет к изменению количества исполнителей, работающих одновременно (т.е. уровня ежедневной загрузки сети).

Задание

Решить задачу сетевого планирования (по своему варианту, см. ниже) и сделать выводы по полученным результатам:

1. Согласно номеру своего варианта (варианты см. ниже) получить исходные данные: время длительности работ сетевой модели, описание упорядочения этих работ, данные о количестве исполнителей, занятых на каждой работе сетевой модели, ограничение по численности одновременно занятых в работе исполнителей.

2. В соответствии с правилами построения сетевых графиков и на основе исходных данных варианта построить сетевую модель.

3. В соответствии с описанными методиками произвести компьютерные расчеты:

– расчет и отображение на сетевом графике временных параметров событий: ранний и поздний срок свершения события, резерв события;

– расчет и представление в таблице временных параметров работ: время раннего и позднего начала работ; время раннего и позднего окончания работ; полный и свободный резервы работ.

4. Согласно номеру своего варианта получить данные о количестве исполнителей, занятых на каждой работе сетевой модели, ограничении по численности одновременно занятых в работе исполнителей.

5. Построить графики привязки и загрузки, используя нормальные длительности работ сети.

6. Произвести уменьшения численности исполнителей, одновременно занятых на работах сети, до требуемого уровня.

Вариант 1

Таблица 5.1

Исходные данные

Название работы	A	B	C	D	E	F	G	H	I	J	K
Длительность работы	8	6	6	8	3	4	7	7	12	9	5
Количество исполнителей	2	2	1	3	1	7	2	2	3	5	7
N=11 человек											

Упорядочение работ

1. А, Е и F - исходные работы проекта, которые можно начинать одновременно;
2. Работы В и I начинаются сразу по окончании работы F;
3. Работа J следует за Е, а работа С - за А;
4. Работы Н и D следуют за В, но не могут начаться, пока не завершена С;
5. Работа К следует за I;
6. Работа G начинается после завершения Н и J.

Вариант 2

Таблица 5.2

Исходные данные

Название работы	A	B	C	D	E	F	G	H	I
Длительность работы	3	4	1	4	5	7	6	5	8
Количество исполнителей	5	7	1	3	2	3	6	1	5
N=11 человек									

Упорядочение работ

1. D - исходная работа проекта;
2. Работа Е следует за D;
3. Работы А, G и С следуют за Е;
4. Работа В следует за А;
5. Работа Н следует за G;
6. Работа F следует за С;
7. Работа I начинается после завершения В, Н, и F.

Вариант 3

Таблица 5.3

Исходные данные

Название работы	A	B	C	D	E	F	G	H	I	J
Длительность работы	5	5	4	7	12	3	6	2	8	3
Количество исполнителей	4	5	4	3	6	4	6	2	1	5
N=10 человек										

Упорядочение работ

1. С, Е и F - исходные работы проекта, которые можно начинать одновременно;
2. Работа А начинается сразу по окончании работы С;
3. Работа H следует за F;
4. Работа I следует за А, а работы D и J - за H;
5. Работа G следует за Е, но не может начаться, пока не завершены D и I;
6. Работа B следует за G и J.

Вариант 4

Таблица 5.4

Исходные данные

Название работы	A	B	C	D	E	F	G	H	I	J
Длительность работы	12	8	15	9	14	9	15	10	11	13
Количество исполнителей	1	4	5	2	3	3	5	5	2	6
N=10 человек										

Упорядочение работ

1. С, J и D - исходные работы проекта, которые можно начинать одновременно;
2. Работа А следует за D, а работа I - за А;
3. Работа H следует за I;
4. Работа F следует за H, но не может начаться, пока не завершена С;
5. Работа G следует за I;
6. Работа Е следует за J, а работа В - за Е.

Вариант 5

Таблица 5.5

Исходные данные

Название работы	A	B	C	D	E	F	G	H	I	J
Длительность работы	12	6	10	7	9	8	10	10	6	5
Количество исполнителей	2	10	2	5	7	6	1	7	1	4
N=11 человек										

Упорядочение работ

1. D - исходная работа проекта;
2. Работы C, E и F начинаются сразу по окончании работы D;
3. Работы A и J следуют за C, а работа G - за F;
4. Работа I следует за A, а работа B - за G;
5. Работа H начинается после завершения E, но не может начаться, пока не завершены I и B.

Вариант 6

Таблица 5.6

Исходные данные

Название работы	A	B	C	D	E	F	G	H	I	J
Длительность работы	9	3	12	6	8	4	7	10	7	12
Количество исполнителей	1	1	7	1	2	10	3	4	2	1
N=11 человек										

Упорядочение работ

1. F, C и B - исходные работы проекта, которые можно начинать одновременно;
2. Работа E следует за F;
3. Работа A следует за B, а работа G - за A;
4. Работы D и J следуют за E;
5. Работа I следует за C, но не может начаться прежде, чем закончатся J и G;

6. Работа Н следует за D.

Вариант 7

Таблица 5.7

Исходные данные.

Название работы	A	B	C	D	E	F	G	H	I	J
Длительность работы	7	6	8	9	10	11	5	9	12	6
Количество исполнителей	3	5	6	1	6	4	7	2	2	5
N=13 человек										

Упорядочение работ

1. G - исходная работа проекта;
2. Работы A, I и D следуют за G и могут выполняться одновременно;
3. Работы C и J следуют за A, работа F - за I, а работа B - за D;
4. Работа E следует за C;
5. Работа H следует за B, но не может начаться, пока не завершена F.

Вариант 8

Таблица 5.8

Исходные данные

Название работы	A	B	C	D	E	F	G	H	I	J
Длительность работы	9	10	6	5	16	12	14	15	11	3
Количество исполнителей	8	3	6	4	5	2	1	3	5	7
N=11 человек										

Упорядочение работ

1. C, D и E- исходные работы проекта, которые можно начинать одновременно;
2. Работа A следует за C, а работа F начинается сразу по окончании работы A;
3. Работа G следует за F;
4. Работа B следует за D, а работы I и J следуют за B;

5. Работа Н следует за I и E, но не может начаться, пока не завершена G.

Вариант 9

Таблица 5.9

Исходные данные

Название работы	A	B	C	D	E	F	G	H	I	J
Длительность работы	9	15	12	5	10	6	5	11	7	8
Количество исполнителей	3	2	6	2	1	9	3	4	5	1
N=11 человек										

Упорядочение работ

1. A, I и D - исходные работы проекта, которые можно начинать одновременно;
2. Работа F следует за A, работа B - за I, а работа C - за D;
3. Работы J и G следуют за F;
4. Работа E следует за J;
5. Работа H начинается после завершения E, G, B и C.

Вариант 10

Таблица 5.10

Исходные данные

Название работы	A	B	C	D	E	F	G	H	I	J
Длительность работы	7	6	8	9	10	11	5	9	12	6
Количество исполнителей	3	5	6	1	6	4	7	2	2	5
N=13 человек										

Упорядочение работ

1. G - исходная работа проекта;
2. Работы A, I и D следуют за G и могут выполняться одновременно;
3. Работы C и J следуют за A, работа F - за I, а работа B - за D;
4. Работа E следует за C;
5. Работа H следует за B, но не может начаться, пока не завершена F.

Вариант 11

Таблица 5.11

Исходные данные

Название работы	A	B	C	D	E	F	G	H	I	J
Длительность работы	9	10	6	5	16	12	14	15	11	3
Количество исполнителей	8	3	6	4	5	2	1	3	5	7
N=11 человек										

Упорядочение работ

1. C, D и E- исходные работы проекта, которые можно начинать одновременно;
2. Работа A следует за C, а работа F начинается сразу по окончании работы A;
3. Работа G следует за F;
4. Работа B следует за D, а работы I и J следуют за B;
5. Работа H следует за I и E, но не может начаться, пока не завершена G.

Вариант 12

Таблица 5.12

Исходные данные

Название работы	A	B	C	D	E	F	G	H	I	J
Длительность работы	9	15	12	5	10	6	5	11	7	8
Количество исполнителей	3	2	6	2	1	9	3	4	5	1
N=11 человек										

Упорядочение работ

1. A, I и D - исходные работы проекта, которые можно начинать одновременно;
2. Работа F следует за A, работа B - за I, а работа C - за D;
3. Работы J и G следуют за F;
4. Работа E следует за J;
5. Работа H начинается после завершения E, G, B и C.

Требования к отчету

В отчете должны быть отображены следующие пункты:

1. Полученные исходные данные: время длительности работ сетевой модели, описание упорядочения этих работ, данные о количестве исполнителей, занятых на каждой работе сетевой модели, ограничение по численности одновременно занятых в работе исполнителей.
2. Сетевая модель в виде сетевого графика.
3. Результаты расчетов:
 - расчет и отображение на сетевом графике временных параметров событий: ранний и поздний срок свершения события, резерв события;
 - расчет и представление в таблице временных параметров работ: время раннего и позднего начала работ; время раннего и позднего окончания работ; полный и свободный резервы работ.
4. Данные о количестве исполнителей, занятых на каждой работе сетевой модели, ограничении по численности одновременно занятых в работе исполнителей.
5. Графики привязки и загрузки, используя нормальные длительности работ сети.
6. Вывод по уменьшению численности исполнителей, одновременно занятых на работах сети, до требуемого уровня.

Контрольные вопросы

1. Что такое сетевой график (сетевая модель)?
2. Какие обозначения используются на сетевом графике?
3. Правила построения сетевого графика.
4. Дайте определения терминов: «полный путь», «критический путь» в сетевом графике.
5. Дайте определения терминов: «критический срок проекта», «критические работы».
6. Дайте определения терминов: «ранний срок свершения события», «поздний срок свершения события», «резерв времени события».
7. Дайте определения терминов: «ранний срок начала работы», «ранний срок окончания работы».
8. Дайте определения терминов: «ранний срок окончания работы», «поздний срок окончания работы», «полный резерв времени работы».

ЛАБОРАТОРНАЯ РАБОТА №6

КОМПЬЮТЕРНАЯ РЕАЛИЗАЦИЯ МЕТОДА ПОТЕНЦИАЛОВ ДЛЯ РЕШЕНИЯ ЗАДАЧИ О ПОТОКЕ МИНИМАЛЬНОЙ СТОИМОСТИ

Цель работы: получить практические навыки решения задачи о потоке минимальной стоимости методом потенциалов.

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Метод потенциалов является точным методом. Он состоит из конечного числа однопотенциальных операций. Каждая операция разбивается на два этапа. На первом этапе проверяется на оптимальность план, полученный на предыдущем этапе. Если план оказывается оптимальным, то переходим ко второму этапу. Строим новый план, который в невырожденном случае, связан с меньшими издержками.

Метод основан на использовании потенциалов. Потенциалами называется система чисел, присвоенная каждому поставщику u_i ($i = \overline{1, m}$) и каждому v_j ($j = \overline{1, n}$). Это, соответственно, ограничения по запасам u_i и ограничения по потребностям v_j .

Решение задачи заключается в определении такой системы потенциалов, для которой выполняются условия:

Для свободных клеток

$$u_i + v_j \leq c_{ij}$$

Для заполненных клеток

$$u_i + v_j = c_{ij}$$

Если предыдущие условия выполняются, то решением будет план X^* , соответствующий полученной системе потенциалов, является оптимальным. Решения X^* называют потенциальными.

Ниже приведен алгоритм решения транспортной задачи в самом общем виде:

- Построение транспортной таблицы.
- Проверка задачи на закрытость.
- Составление опорного плана.
- Проверка опорного плана на вырожденность.
- Вычисление потенциалов для плана перевозки.
- Проверка опорного плана на оптимальность.
- Перераспределение поставок.
- Если оптимальное решение найдено, переходим к п. 9, если нет – к

п. 5.

- Вычисление общих затрат на перевозку груза.
- Построение графа перевозок.

Задание

Методом потенциалов найти оптимальный план задачи о потоке минимальной стоимости, варианты см. ниже. Компьютерную реализацию алгоритма выполнить в Visual Studio.

Вариант 1

Таблица 6.1

Исходные данные

$a_i \backslash b_j$	60	40	40	30	10
60	5	2	0	7	3
40	6	1	4	2	8
70	7	4	3	6	1
30	3	5	6	4	2

Вариант 2

Таблица 6.2

Исходные данные

$a_i \backslash b_j$	60	40	40	30	10
60	5	2	0	1	1
40	10	3	4	2	8
70	7	14	7	5	1
30	3	5	6	4	12

Вариант 3

Таблица 6.2

Исходные данные

$a_i \backslash b_j$	20	30	20	10
20	6	3	5	1
30	4	9	4	7
40	1	5	8	4

Вариант 4

Таблица 6.4

Исходные данные

$a_i \backslash b_j$	20	25	30	25
40	4	2	5	7
30	6	0	3	1
10	5	4	2	3

Вариант 5

Таблица 6.5

Исходные данные

$a_i \backslash b_j$	80	80	60	180
80	3	2	1	4
120	1	3	2	5
100	2	3	1	4
100	5	4	3	2

Вариант 6

Таблица 6.6

Исходные данные

$a_i \backslash b_j$	100	150	80	90
60	6	1	5	7
130	3	2	4	5
100	5	4	2	6
80	4	8	3	2

Вариант 7

Таблица 6.7

Исходные данные

$a_i \backslash b_j$	60	60	140	140	80
200	5	4	3	4	5
140	3	2	1	3	2
60	2	3	2	3	1
100	4	5	2	3	4

Вариант 8

Таблица 6.8

Исходные данные

$a_i \backslash b_j$	100	100	200	100	60
200	3	2	1	1	2
100	2	1	3	2	3
200	3	3	2	1	1
100	4	3	2	1	5

Вариант 9

Таблица 6.9

Исходные данные

$a_i \backslash b_j$	50	40	10	15	25	30
70	1	3	1	5	7	4
50	8	4	8	4	3	6
20	3	5	5	6	2	4
30	5	1	6	3	6	2

Вариант 10

Таблица 6.10

Исходные данные

$a_i \backslash b_j$	80	60	120	140	80	20
100	4	3	5	1	4	3
200	1	4	5	2	3	1
100	2	1	3	3	1	3
50	5	4	2	2	4	2

Вариант 11

Таблица 6.11

Исходные данные

$a_i \backslash b_j$	20	40	80	60	100
50	2	3	1	4	5
80	5	1	1	2	3

Окончание табл. 6.11

$a_i \backslash b_j$	20	40	80	60	100
100	5	2	1	3	1
100	2	1	4	3	1

Вариант 12

Таблица 6.12

Исходные данные

$a_i \backslash b_j$	60	80	120	40	40
100	2	1	3	1	1
80	1	2	3	1	1
100	2	3	1	3	2
50	3	2	2	2	4

Требования к отчет

Контрольные вопросы

1. Определение сети.
2. Определение транспортной сети.
3. Определение потока транспортной сети.
4. Определение разреза и притока сети?
5. Чему равна пропускная способность разреза?
6. Приведите примеры прикладных задач, имеющих сетевую форму.
7. Метод потенциалов, пошаговое описание алгоритма.

ЛАБОРАТОРНАЯ РАБОТА №7

ПОСТРОЕНИЕ ДЕРЕВА КРАТЧАЙШИХ ПУТЕЙ МЕТОДОМ ПОМЕТОК И АЛГОРИТМОМ ДЕЙКСТРЫ

Цель работы: получить практические навыки решения задач сетевого планирования кратчайшего пути.

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Алгоритм расстановки пометок Дейкстры или просто алгоритм пометок имеет итеративный характер и заключается в выполнении следующих действий (шагов):

1 предварительное определение значений временных и постоянных пометок до начала выполнения основных итераций алгоритма определить следующие значения: для начальной вершины установить постоянную пометку $\bar{l}_s = 0$, а для всех остальных вершин — временные пометки: $l_i = \infty (\forall v_i \in V \setminus \{v_s\})$; для удобства установить $v_s = v_1$ и $k = 1$, т.е. $V = \{v_1\}$ и $V_0 = \{v_2, v_3, \dots, v_n\}$; множество дуг, входящих в минимальный путь, до начала выполнения алгоритма устанавливается равным: $E_{min} = \emptyset$. После чего можно перейти к выполнению действий шага 2;

2 обновление временных пометок среди вершин $v_j \in V_k$ с V_0 , инцидентных вершине v_k с постоянной пометкой \bar{l}_k , и имеющих временные пометки l_j , изменить эти временные пометки на новые значения l'_j с использованием следующей формулы:

$$l'_j = \min\{l_j, \bar{l}_k + c_{kj}\}, v_j \in V_k$$

После чего перейти к выполнению действий шага 3;

3 превращение временной пометки в постоянную среди всех вершин графа V_0 , имеющих временные пометки, выбрать вершину с минимальным значением временной пометки l'_j ; сделать эту пометку постоянной, т.е. $\bar{l}_j = l'_j$ и тем самым изменить множество вершин с постоянными и временными пометками следующим образом: вершина v_j исключается из множества V_0 и добавляется в множество V , которые становятся равными: $V'_0 = V_0 / \{v_j\}$ и $V' = V \cup \{v_j\}$; установить $k = j$ и перейти к выполнению действий шага 4;

4 проверка условия окончания основных итераций при нахождении только минимального пути из начальной вершины v_s в конечную вершину v_t , проверить выполнение условия: $k = t$. Если это условие выполняется, то конечная вершина v_t исходного графа имеет постоянную пометку \bar{l}_t , которая равна длине минимального пути в нее из начальной вершины v_s . На этом выполнение основных итераций алгоритма заканчивается и выполняется переход к выполнению действий последнего шага 5, на котором определяется множество дуг E_{min} , входящих в минимальный путь. Если же условие $k = t$ не выполняется, то установить: $V_0 = V'_0$, $\bar{V} = \bar{V}'$ и перейти к выполнению действий шага 2;

5 определение структуры минимального пути для каждой пары вершин, входящих в множество V с постоянными пометками, проверить выполнение следующего условия: $\bar{l}_j - \bar{l}_i = c_{ij}$. Если это условие выполняется, то дугу $(v_i, v_j) \in E$, где $v_i, v_j \in \bar{V}$, следует включить в множество E_{min} . После окончания проверки сформулированного условия выполнения для последней пары вершин из множества V выполнение алгоритма заканчивается.

Если на шаге 3 алгоритма пометок имеется несколько вершин с одинаковым значением минимальных временных пометок l'_j , то следует выбрать любое из них. При этом произвольный выбор дуги с минимальным значением не оказывает принципиального влияния на конечный результат решения задачи. Проверку условия на шаге 5 целесообразно выполнять последовательно, начиная от конечной вершины графа, при этом выстраивать соответствующий путь.

Следует заметить, что в случае определения множества минимальных путей из начальной вершины во все остальные, в качестве базового условия на шаге 4 следует проверять условие: $V'_0 = \emptyset$. В случае его истинности принимается решение о завершении основных итераций алгоритма, а в случае ложности — осуществляется переход к выполнению действий шага 2 описанного алгоритма.

Нетрудно заметить, что в силу конечности вершин исходного графа, рассмотренный алгоритм пометок является конечным. Более того, поскольку заранее известно, что на каждой основной итерации в точности одна вершина исходного графа получает постоянную пометку, общее количество итераций данного алгоритма, связанных с выполняемыми на шагах 2 и 3 действиями, конечно и не превышает: $n - 1$.

Рассмотренный алгоритм пометок может быть использован также для нахождения минимальных путей как в ориентированном, так и в неориентированном графе.

Задание

С помощью алгоритма расстановки пометок Дейкстры отыскать кратчайший путь в ориентированном графе, варианты см. ниже. Компьютерную реализацию алгоритма выполнить в Visual Studio.

Вариант 1

Для ориентированного графа найти кратчайший путь из вершины 1 в вершину 4

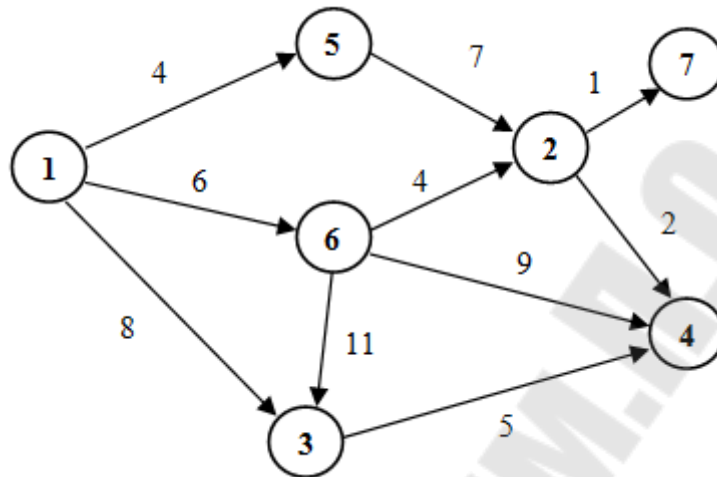


Рис. 7.1 Ориентированный граф

Вариант 2

Для ориентированного графа найти кратчайший путь из вершины 5 в вершину 4

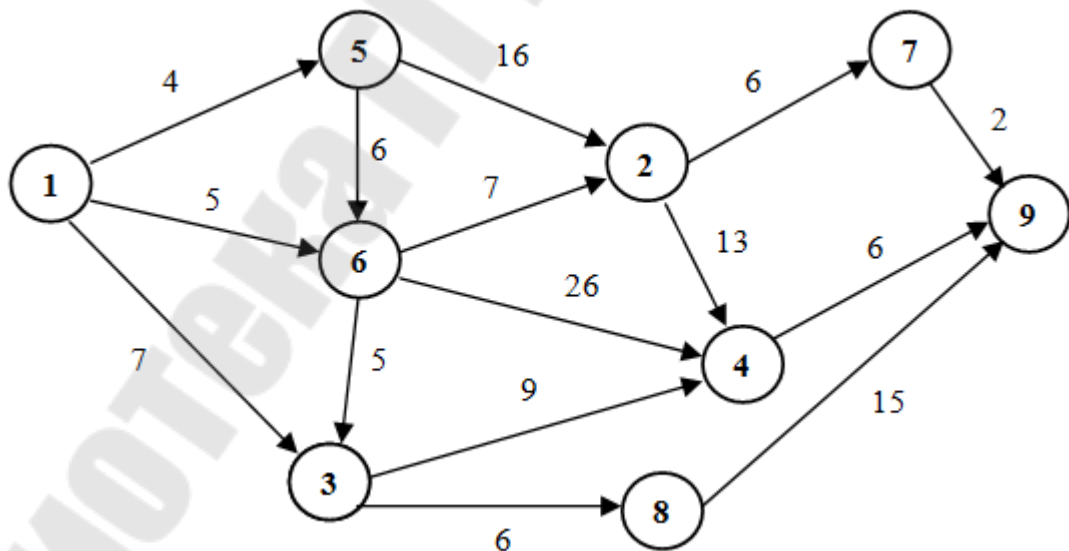


Рис. 7.2 Ориентированный граф

Вариант 3

Для ориентированного графа найти кратчайший путь из вершины 6 в вершину 9

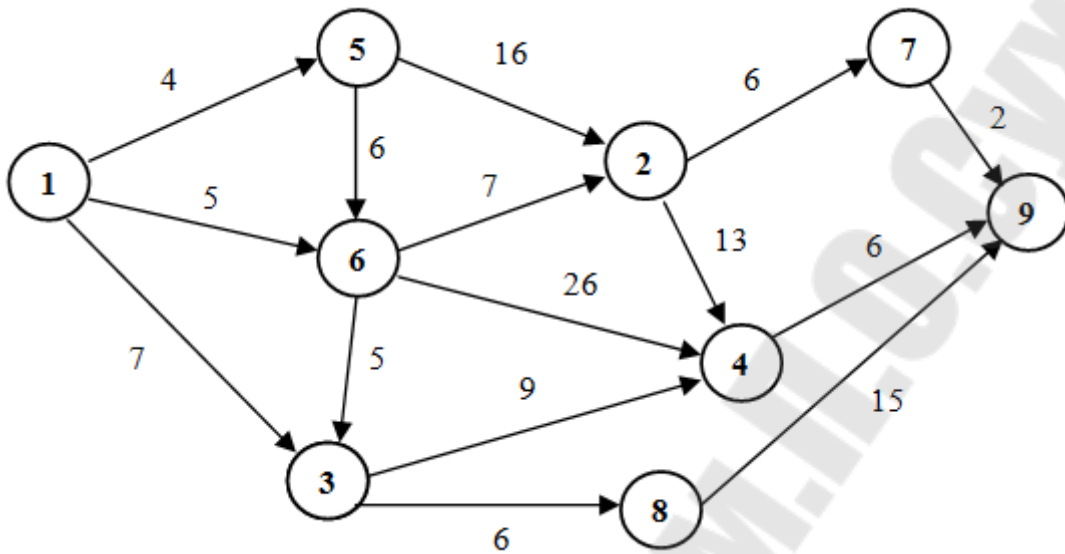


Рис. 7.3 Ориентированный граф

Вариант 4

Для ориентированного графа найти кратчайший путь из вершины 1 в вершину 7

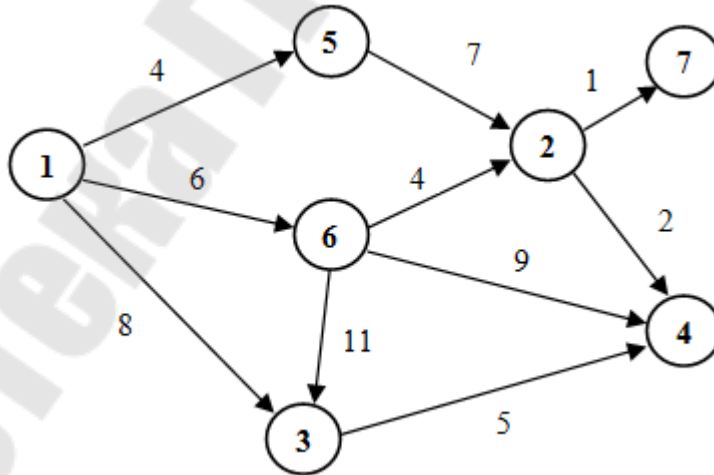


Рис. 7.4 Ориентированный граф

Вариант 5

Для ориентированного графа найти кратчайший путь из вершины 1 в вершину 2

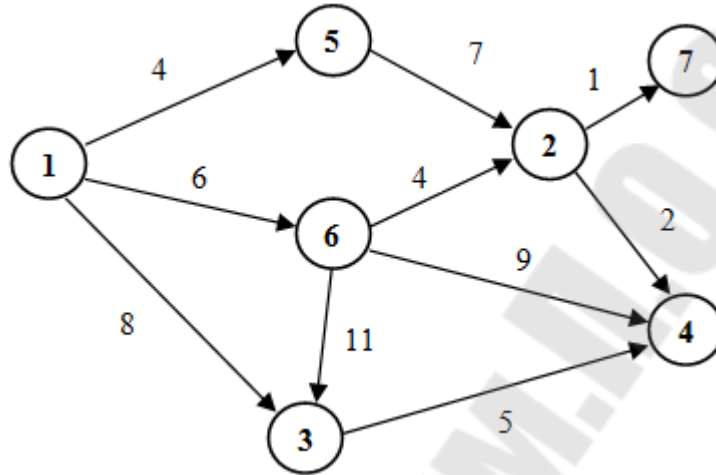


Рис. 7.5 Ориентированный граф

Вариант 6

Для ориентированного графа найти кратчайший путь из вершины 5 в вершину 4

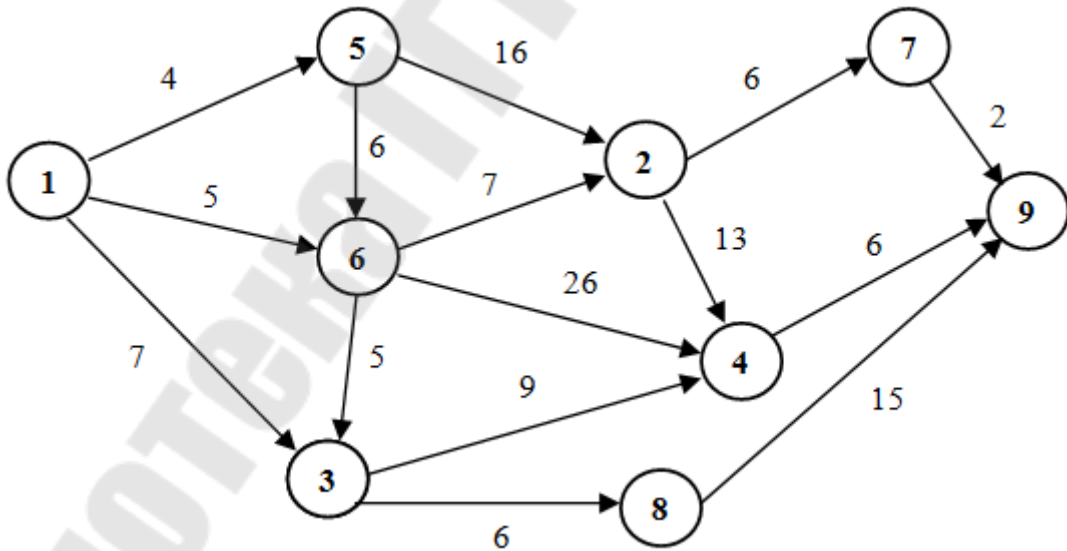


Рис. 7.6 Ориентированный граф

Вариант 7

Для ориентированного графа найти кратчайший путь из вершины 5 в вершину 9

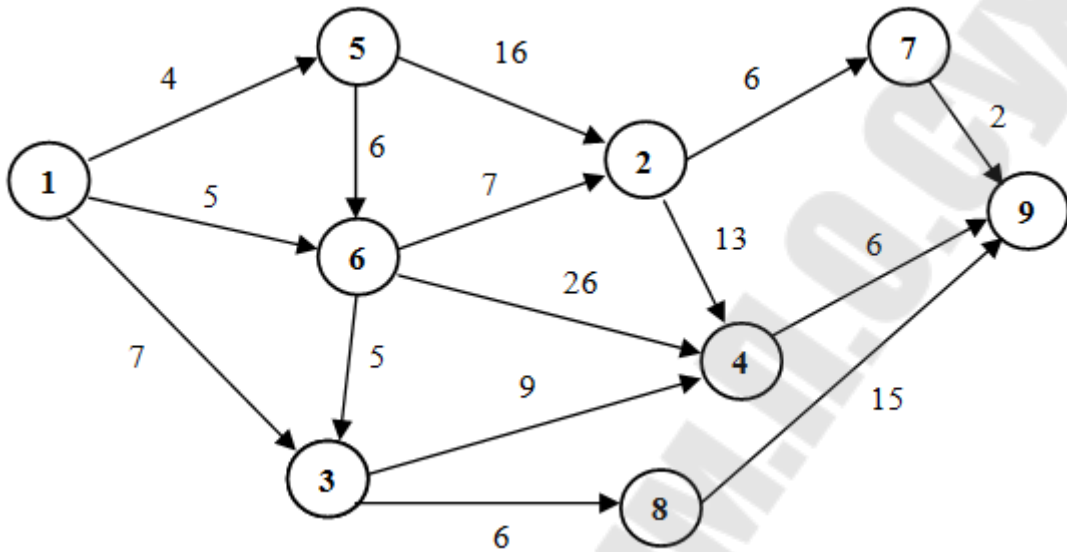


Рис. 7.7 Ориентированный граф

Вариант 8

Для ориентированного графа найти кратчайший путь из вершины 3 в вершину 9

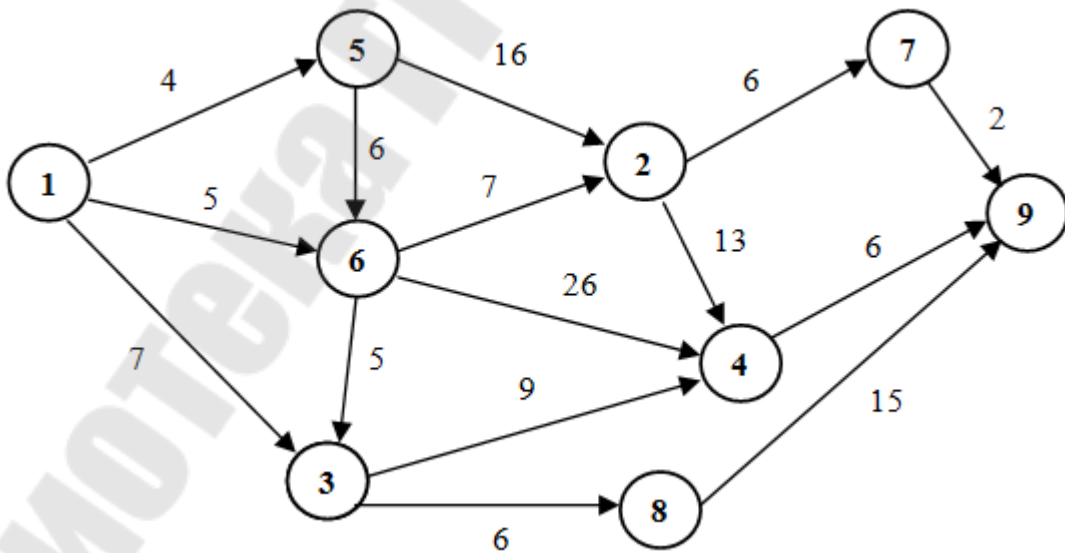


Рис. 7.8 Ориентированный граф

Вариант 9

Для ориентированного графа найти кратчайший путь из вершины 3 в вершину 9

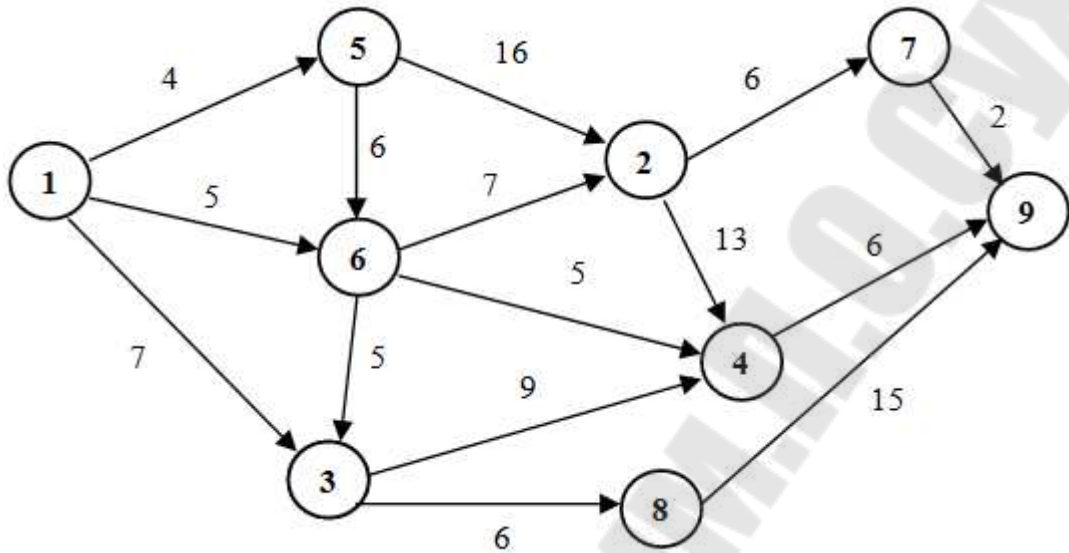


Рис. 7.9 Ориентированный граф

Вариант 10

Для ориентированного графа найти кратчайший путь из вершины 1 в вершину 7

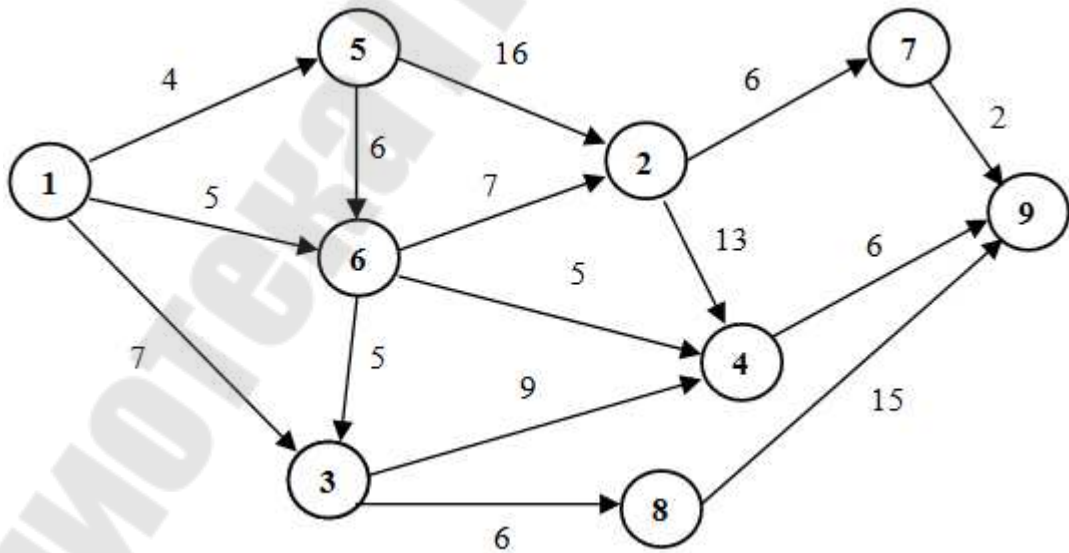


Рис. 7.10 Ориентированный граф

Вариант 11

Для ориентированного графа найти кратчайший путь из вершины 1 в вершину 4

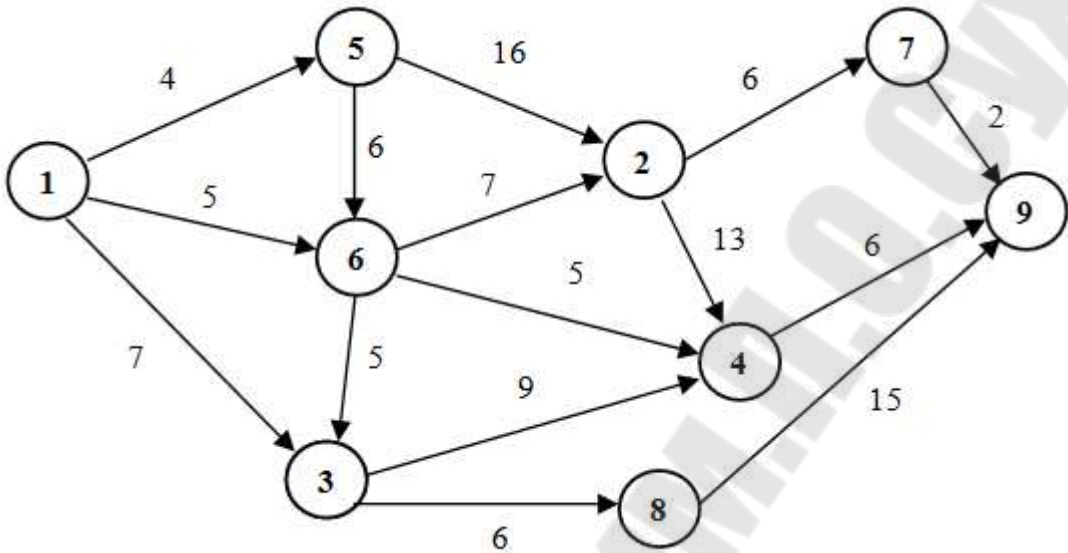


Рис. 7.11 Ориентированный граф

Вариант 12

Для ориентированного графа найти кратчайший путь из вершины 6 в вершину 9

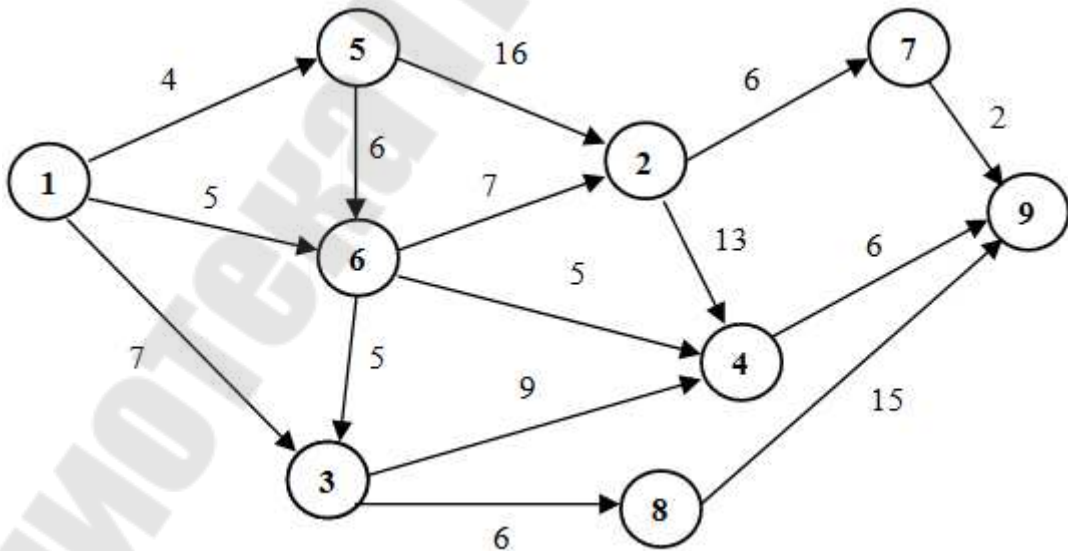


Рис. 7.12 Ориентированный граф

Требования к отчету

Отчет по лабораторной работе должен содержать:

- цель работы;
- номер варианта;
- исходные данные варианта;
- листинг кода реализации алгоритмов;
- результаты вычислений;
- выводы.

Контрольные вопросы

1. Кратчайший путь?
2. Формальная и математическая постановка задачи о нахождения кратчайшего пути в сетевом графе.
3. Алгоритм Флойда для нахождения кратчайшего пути сетевого графика.
4. Понятие «дерево графа», «ориентированный граф»?

ЛАБОРАТОРНАЯ РАБОТА №8

НАХОЖДЕНИЕ НА СЕТИ ПОТОКА МАКСИМАЛЬНОЙ ВЕЛИЧИНЫ

Цель работы: получить практические навыки решения задач отыскания на сети потока максимальной величины.

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Алгоритм основывается на теореме Форда-Фалкерсона: в любой транспортной сети максимальный поток равен минимальной пропускной способности. Если поток максимален, то найдётся такое сечение, пропускная способность которого равна мощности потока.

Доказывается эта теорема применением алгоритма Форда-Фалкерсона. Согласно этому алгоритму, начиная с некоторого начального неполного потока, по итеративному алгоритму можно получить полный поток, если прибавлять к различным значениям потоков $l(g)$ пути S_i минимальное из чисел $[c(g) - l(g)]$, которые вычислены по этому пути S_i . После такой операции путь S_i уже будет содержать хотя бы одну насыщенную дугу. Если таким же образом поступить с другими путями S_j , то в конце концов получим полный поток.

Введём ещё несколько определений. Дуга g называется *насыщенной*, если $\phi(g)=C(g)$. Поток называется *полным*, если любой путь из x_0 в x_n содержит хотя бы одну насыщенную дугу. Если некоторый *начальный* поток является неполным, то можно найти такой путь S , который соединяет x_0 с x_n и не содержит насыщенной дуги.

Поэтому алгоритм определения максимального потока состоит из следующих шагов:

1. Строим начальный поток X^0 .
2. Проверяем, попал ли узел x_n в множество узлов S , которые достижимы по ненасыщенным рёбрам из x_0 . Если узел x_n не попал, то считают, что построенный поток максимален, и алгоритм расчёта останавливается.
3. Если узел x_n попал в множество S , то выделяют путь S_j , состоящий из ненасыщенных рёбер и ведущий грузы из x_0 в x_n .
4. Увеличивают поток через каждое ребро этого пути на величину $\min(c_{ij} - \phi_{ij})$.
5. Строят новый поток X и переходят к шагу 2.
6. Обычно сеть задаётся матрицей пропускных способностей $\Sigma = \|c_{ij}\|$ всех рёбер сети U . Задавшись $k = 0$, затем вычисляют на k -м шаге матрицу значений потоков на дугах $X^k = \|\phi_{ij}^k\|$. Строят матрицу разностей $\Sigma - X^k = \|c_{ij} - \phi_{ij}^k\|$. В этой матрице насыщенным рёбрам при потоке X^k будут соответствовать нулевые элементы, а ненасыщенным – ненулевые элементы.

Поэтому вычисление матрицы $\|\Sigma - X^k\|$ достаточно как для построения множества узлов, по которым вещество из x_0 достигает по ненасыщенным рёбрам до x_n , так и для построения последовательности ненасыщенных рёбер S_1 , позволяющих достигать x_n из x_0 .

Технология составления этих списков следующая:

- сначала составляют список узлов, в каждый из которых ведёт ненасыщенное ребро из вершины i ;
- далее для каждого i -го узла составляют свой список узлов, в каждый из которых из i -го узла ведёт ненасыщенное ребро (за исключением тех узлов, которые уже вошли в ранее составленные списки), и так далее. Этот процесс выписывания списков заканчивается в двух случаях. Либо появился узел x_n , что означает продолжение работы алгоритма, либо в список выписанных узлов не попал узел x_n , что означает конец расчётов.

Поэтому с вычислительной точки зрения алгоритм Форда-Фалкерсона реализует такую последовательность действий:

1. Вычисление матрицы $\|\Sigma - X^k\|$.
2. Построение списков вершин.

3. Нахождение последовательности ненасыщенных рёбер, соединяющих x_0 с x_n .
4. Нахождение величины дополнительного потока $\Delta\phi = \min \{c_{ij} - \phi_{ij}\}$.
5. Построение нового потока X^{k+1} .
6. Если в списке отсутствует x_n , то поток X_k – максимален.

Задание

Ниже в таблице 8.1 приведены множества I, U и веса дуг соответственно перечислению их в U . Вес дуги – пропускная способность дуги. Считаем вершину 1 источником, вершину 9 – стоком.

а) Найти поток максимальной величины из источника в сток, используя алгоритм Форда-Фалкерсона.

б) Найти соответствующий найденному потоку разрез минимальной пропускной способности.

Компьютерную реализацию алгоритма выполнить в Visual Studio.

Таблица 8.1

Исходные данные

Вариант 1	I
U	{(1,2), (1,4), (2,3), (2,5), (3,5), (3,6), (4,5), (4,7), (5, 1), (5,6), (5,7), (5,8), (6,9), (7,8), (8,9), (9,5)}
веса	10, 4, 3, 9, 11, 5, 12, 6, 11, 13, 10, 4, 7, 8, 6, 8
Вариант 2	I
U	{(1,2), (1,4), (2,3), (2,5), (3,6), (4,2), (4,5), (4,7), (4,8), (5, 6), (5,8), (5,9), (6,9), (7,8), (8,9)}
веса	10, 5, 3, 9, 11, 5, 10, 6, 15, 13, 10, 5, 7, 8, 4
Вариант 3	I
U	{(1,2), (1,4), (1,5), (2,3), (2,6), (3,6), (4,5), (4,7), (5,2), (5, 6), (5,7), (6,8), (7,8), (8,9)}
веса	10, 14, 13, 9, 11, 15, 12, 16, 15, 13, 10, 14, 7, 8,
Вариант 4	I

Окончание табл. 8.1

U	{(1,2), (1,4), (2,3), (2,4), (2,6), (3,6), (4,5), (4,7), (4,8), (5,2), (5,8), (6,5), (6,8), (6,9), (7,8), (8,9)}
веса	25, 14, 23, 19, 16, 15, 4, 22, 28, 6, 26, 24, 27, 18, 14, 18
Вариант 5	I
U	{(1,2), (1,4), (2,3), (2,5), (3,5), (3,6), (4,5), (4,7), (5, 1), (5,6), (5,7), (5,8), (6,9), (7,8), (8,9), (9,5)}
веса	20, 15, 22, 16, 25, 17, 18, 14, 18, 14, 13, 19, 21, 23, 20, 24
Вариант 6	I
U	{(1,2), (1,4), (2,3), (2,5), (3,6), (4,2), (4,5), (4,7), (4,8), (5, 6), (5,8), (5,9), (6,9), (7,8), (8,9)}
веса	5, 15, 4, 8, 10, 4, 3, 9, 11, 12, 6, 10, 4, 7, 8,
Вариант 7	I
U	{(1,2), (1,4), (1,5), (2,3), (2,6), (3,6), (4,5), (4,7), (5,2), (5, 6), (5,7), (6,8), (7,8), (8,9)}
веса	5, 4, 7, 10, 5, 6, 15, 13, 10, 5, 9, 8, 3, 8
Вариант 8	I
U	{(1,2), (1,4), (2,3), (2,4), (2,6), (3,6), (4,5), (4,7), (4,8), (5,2), (5,8), (6,5), (6,8), (6,9), (7,8), (8,9)}
веса	14, 14, 12, 8, 10, 15, 13, 16, 15, 13, 11, 10, 7, 8, 14, 9
Вариант 9	I
U	{(1,2), (1,4), (2,3), (2,5), (3,5), (3,6), (4,5), (4,7), (5, 1), (5,6), (5,7), (5,8), (6,9), (7,8), (8,9), (9,5)}
веса	25, 23, 20, 24, 14, 18, 15, 14, 23, 19, 27, 18, 16, 15, 14, 26,
Вариант 10	I
U	{(1,2), (1,4), (2,3), (2,5), (3,6), (4,2), (4,5), (4,7), (4,8), (5, 6), (5,8), (5,9), (6,9), (7,8), (8,9)}
веса	23, 20, 24, 17, 15, 22, 16, 14, 18, 20, 14, 13, 19, 21, 25
Вариант 11	I
U	{(1,2), (1,4), (1,5), (2,3), (2,6), (3,6), (4,5), (4,7), (5,2), (5, 6), (5,7), (6,8), (7,8), (8,9)}
веса	4, 11, 5, 12, 6, 15, 4, 8, 3, 9, 13, 10, 7, 8

Требования к отчету

Отчет по лабораторной работе должен содержать:

- цель работы;
- номер варианта;
- исходные данные варианта;
- листинг кода реализации алгоритмов;
- результаты вычислений;
- выводы.

Контрольные вопросы

1. Содержательная и математическая постановка задачи о максимальном потоке в сети
2. Задача о наибольшем потоке в сети
3. Условие максимальности потока
4. Алгоритм определения максимального потока

ЛАБОРАТОРНАЯ РАБОТА №9

РЕШЕНИЕ ЗАДАЧИ О НАЗНАЧЕНИЯХ ВЕНГЕРСКИМ МЕТОДОМ

Цель работы: получить практические навыки решения задачи о назначении венгерским методом.

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Постановка задачи: пусть имеется n видов работ и n исполнителей, каждый из которых может выполнить любую из этих работ; c_{ij} – эффективность выполнения i -ой работы j -ым исполнителем, $i, j = 1, n$. Из параметров c_{ij} можно составить $(n \times n)$ матрицу стоимостей (эффективностей) C . Найти такую расстановку исполнителей, чтобы суммарный эффект их труда был наибольшим.

Математическая модель задачи: введем переменные

$$x_{ij} = \begin{cases} 1, & \text{работа } i \text{ выполняется } j\text{-м исполнителем,} \\ 0, & \text{в противном случае.} \end{cases}$$

Получаем задачу:

$$L(x) = \sum_{i=1}^n \sum_{j=1}^n c_{ij} x_{ij} \rightarrow \max \quad \sum_{j=1}^n x_{ij} = 1 \quad (1), \quad \sum_{i=1}^n x_{ij} = 1 \quad (2), \quad i, j = \overline{1, n}$$

Ограничения (1) означают, что на каждую работу должен быть назначен только один исполнитель. Ограничения (2) означают, что каждый исполнитель выполняет только одну работу.

Определение. Система нулевых элементов матрицы, обладающая тем свойством, что никакая пара из них не лежит в одной строке или в одном столбце, называется *системой независимых нулей (СНИ)*.

Идея метода.

Венгерский метод состоит в преобразовании исходной задачи с матрицей C в эквивалентную ей задачу минимизации с матрицей C'' . Если в матрице C'' имеется система из n независимых нулей, то решением задачи будет матрица X , у которой на местах, соответствующих независимым нулям, стоят 1, а остальные элементы являются нулевыми. Таким образом алгоритм направлен на построение СНИ путем эквивалентных преобразований матрицы C'' .

Алгоритм метода.

0. Преобразуем матрицу C в матрицу C'' следующим образом:

$$c'_{ij} = \max\{c_{ij}\} - c_{ij}, \quad i, j = \overline{1, n};$$

$$c'' = c'_{ij} - \min_{1 \leq j \leq n} \{c'_{ij}\}, \quad i, j = \overline{1, n}.$$

Приходим к задаче:

$$L(x) = \sum_{i=1}^n \sum_{j=1}^n c''_{ij} x_{ij} \rightarrow \max \quad \sum_{j=1}^n x_{ij} = 1 \quad (1), \quad \sum_{i=1}^n x_{ij} = 1 \quad (2), \quad i, j = \overline{1, n}$$

1. Строим систему независимых нулей, помечая их *. Пусть число таких элементов равно k . Если $k=n$, то пишем ответ — матрицу X и $L(X)$. Если $k \neq n$, то помечаем столбцы, содержащие 0^* сверху знаком «+». Все элементы в этих столбцах назовем *выделенными элементами (ВЭ)* (аналогично, так назовем элементы и в строках со знаком «+»).

2. Среди невыделенных элементов ищутся нулевые. Если они есть, то помечаем один из них $0'$ и переходим на 3. Если нет, то — 5.

3. Если в строке, содержащей только что найденный $0'$ нет 0^* , то — переход на 4. Иначе с соответствующего столбца снимается знак выделения и выделяется строка, содержащая $0'$ и осуществляется переход на 2.

4. Начиная с только что найденного $0'$ строится цепочка нулевых элементов матрицы по следующему правилу: исходный $0'$; 0^* , лежащий в одном столбце с этим $0'$; $0'$, лежащий в строке с только что пройденным 0^* и т.д. Заканчивается цепочка $0'$. После построения цепочки штрихи у нулевых элементов этой цепочки заменяются на *, старые * в цепочке уничтожаются. Убираются знаки выделения строк и столбцов. * не входящие в цепочку переписываются. Далее переход на 1.

5. Ищем минимальный ненулевой неВЭ и обозначаем h . Затем, прибавляем h к элементам, которые стоят на пересечении выделенных строк и выделенных столбцов; отнимаем h от элементов, которые стоят на пересечении невыделенных строк и не выделенных столбцов; остальные элементы переписываем без изменений. Возвращаемся к 2.

Пример. Решить задачу о назначениях с матрицей C венгерским методом.

$$C = \begin{pmatrix} 9 & 6 & 5 & 8 \\ 4 & 8 & 6 & 2 \\ 6 & 7 & 9 & 4 \\ 2 & 7 & 3 & 1 \end{pmatrix}$$

Решение.

$$C = \begin{pmatrix} 9 & 6 & 5 & 8 \\ 4 & 8 & 6 & 2 \\ 6 & 7 & 9 & 4 \\ 2 & 7 & 3 & 1 \end{pmatrix} \rightarrow C' = \begin{pmatrix} 0 & 2 & 4 & 0 \\ 5 & 0 & 3 & 6 \\ 3 & 1 & 0 & 4 \\ 7 & 1 & 6 & 7 \end{pmatrix} \rightarrow C'' = \begin{pmatrix} 0^* & 2 & 4 & 0' \\ 5 & 0^* & 3 & 6 \\ 3 & 1 & 0^* & 4 \\ 6 & 0 & 5 & 6 \end{pmatrix}^+$$

1. В матрице C_n строим и подсчитываем число независимых нулей; $k = 3 < 4$, значит столбцы, содержащие 0^* помечаем +.

2. Среди невыделенных элементов ищется нулевой, это элемент c_{14} помечаем его '.

3. В первой строке есть 0^* , поэтому с первого столбца надо снять знак выделения и отметить знаком + первую строку.

4. Среди неВЭ нет нулевых.

5. Выбираем минимальный элемент среди неВЭ $c_{31} = h = 3$ и

— прибавляем h к элементам, которые стоят на пересечении ВЭ строк и ВЭ столбцов;

— вычитаем h от элементов, которые стоят на пересечении неВЭ строк и неВЭ столбцов;

— оставляем без изменения остальные элементы. Получаем матрицу:

$$C''' = \begin{pmatrix} 0^* & 5 & 7 & 0' \\ 2 & 0^* & 3 & 3 \\ 0' & 1 & 0^* & 1 \\ 3 & 0 & 5 & 3 \end{pmatrix}^+$$

2. Среди неВЭ есть нулевой: $c_{31} = 0$, помечаем его '.

3. В третьей строке существует 0^* , значит снимаем с третьего столбца знак \oplus и помечаем третью строку +.

2. Среди неВЭ нет нулей.

5. Выбираем в C''' $c_{21} = h = 2$ и снова строим новые нули, получаем матрицу:

$$C^{IV} = \begin{pmatrix} 0^* & 7 & 7 & 0' \\ 0' & 0^* & 1 & 1 \\ 0' & 3 & 0^* & 1 \\ 1 & 0' & 3 & 1 \end{pmatrix} \begin{matrix} + \\ + \\ + \\ \end{matrix}$$

1. Подсчитываем число $k = 4 = n$, значит записываем ответ:

$$X = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}, \quad L(X) = 8 + 4 + 9 + 7 = 28(\text{ед.})$$

Задание

Решить венгерским методом задачу о назначениях по своему варианту (см. таблицу вариантов ниже). При этом:

- составить таблицу с исходными данными задачи;
- построить математическую модель задачи;
- составить требуемые таблицы стоимости выполнения работ, подробно пояснить ход решения и полученные результаты.

Условие задачи:

Для дорог республиканского значения с облегченным покрытием межремонтный срок службы составляет 10 лет. К истекшему сроку ДРСУ запланировало произвести капитальный ремонт автомагистрали. Для этого был объявлен тендер на проведение ремонтных работ, в ходе которого было отобрано 5 строительных организаций-подрядчиков (A_i). Каждая организация дала оценку времени в сутках t_{ij} ($i = \overline{1,5}; j = \overline{1,4}$), требующегося ей для выполнения всех работ (B_j): B_1 – уборка полосы отвода (вырубка леса и кустарника), B_2 – ремонт искусственных сооружений, B_3 – укрепление земляного полотна, B_4 – косметический ремонт дорожной одежды.

Качество выполнения организациями работ одинаковое. Организации, занятые выполнением заказа, потребовали оплату за одни сутки в размере c_i . Организация № 3 не выполняет работы, связанные с укреплением земляного полотна. Какая из организаций не получит заказ? Как ДРСУ следует распределить работы между организациями, чтобы

минимизировать общие издержки капитального ремонта автомагистрали?
Исходные данные и варианты заданий приведены в таблицах 9.1 – 9.3.

Таблица 9.1

Организация	Комплекс ремонтных работ				Стоимость работ c_{ij} , ден. ед. /сут.
	B_1	B_2	B_3	B_4	
	Время выполнения в сутки				
A_1	t_{11}	t_{12}	t_{13}	t_{14}	c_{1j}
A_2	t_{21}	t_{22}	t_{23}	t_{24}	c_{2j}
A_3	t_{31}	t_{32}	t_{33}	t_{34}	c_{3j}
A_4	t_{41}	t_{42}	t_{43}	t_{44}	c_{4j}
A_5	t_{51}	t_{52}	t_{53}	t_{54}	c_{5j}

Таблица 9.2

Варианты заданий (время выполнения работ)

Вариант	Время																			
	t_{11}	t_{12}	t_{13}	t_{14}	t_{21}	t_{22}	t_{23}	t_{24}	t_{31}	t_{32}	t_{33}	t_{34}	t_{41}	t_{42}	t_{43}	t_{44}	t_{51}	t_{52}	t_{53}	t_{54}
1	5	16	7	8	10	4	9	8	19	4	5	7	6	5	10	7	5	17	7	6
2	9	7	9	8	10	9	11	4	7	9	7	4	10	5	7	8	8	7	9	17
3	10	5	9	6	10	5	10	6	4	5	3	10	4	5	7	6	10	8	8	5
4	8	12	10	7	8	7	16	7	5	14	5	12	13	8	4	9	6	5	4	12
5	4	9	5	8	7	5	5	8	4	6	5	10	16	7	8	5	5	2	3	9
6	12	25	23	21	18	12	19	20	19	23	16	13	15	29	13	15	29	15	25	14
7	7	27	21	22	17	14	17	22	17	24	15	12	13	36	12	13	36	12	25	23
8	7	31	19	23	16	13	15	29	15	25	14	11	11	43	11	11	43	7	27	21
9	5	35	17	24	15	12	13	36	12	25	23	21	18	12	19	20	19	23	16	13
10	12	39	15	25	14	11	11	43	7	27	21	22	17	14	17	22	17	24	15	12
11	5	3	10	4	5	7	6	10	18	12	19	20	19	23	16	13	15	29	15	22
12	4	5	6	3	8	4	9	6	17	14	17	22	17	24	15	12	13	36	12	29
13	6	5	10	6	7	8	5	5	16	13	15	29	15	25	14	11	11	43	11	36
14	23	16	13	15	29	13	5	6	15	12	13	36	10	4	5	7	6	10	18	43
15	24	15	12	13	36	12	9	7	10	7	5	7	2	3	8	4	9	16	17	10
16	6	7	5	14	5	22	10	5	7	8	8	7	10	6	7	8	15	15	16	6

Таблица 9.3

Варианты заданий (стоимость выполнения работ)

время	Стоимость выполнения i -й организацией j - работы c_{ij} , ден. ед. / сут.					Невыполнение организацией работы	
	c_{1j}	c_{2j}	c_{3j}	c_{4j}	c_{5j}	Организация	Наименование работы
1	2	1	3	1	2	A_1	B_1
2	2	2	3	1	1	A_2	B_2
3	4	3	2	3	4	A_3	B_3
4	5	4	5	4	4	A_4	B_4
5	3	3	2	3	2	A_5	B_1
6	3	2	2	1	3	A_1	B_2
7	4	4	3	3	3	A_2	B_3
8	3	3	4	3	4	A_3	B_4
9	1	2	3	1	3	A_4	B_1
10	2	2	3	3	2	A_5	B_2
11	1	2	1	2	2	A_1	B_3
12	4	4	5	4	4	A_2	B_4
13	3	3	3	3	2	A_3	B_1
14	3	2	2	1	3	A_4	B_2
15	4	4	3	4	3	A_5	B_3
16	3	2	3	2	3	A_1	B_4

Требования к отчету

Отчет по лабораторной работе должен содержать:

- цель работы;
- номер варианта;
- исходные данные варианта;
- листинг кода реализации алгоритмов;
- результаты вычислений;
- выводы.

Контрольные вопросы

1. Постановка задачи о назначениях (задачи выбора).
2. Чем отличаются задачи о назначениях открытого и закрытого типов?

3. Как привести задачу о назначениях открытого типа к закрытому типу?
4. Алгоритм решения задачи о назначениях венгерским методом.
5. Что такое матрица стоимостей выполнения работ при решении задачи о назначениях?

ЛАБОРАТОРНАЯ РАБОТА №10

КОМПЬЮТЕРНАЯ РЕАЛИЗАЦИЯ АЛГОРИТМА ДЖОНСОНА

Цель работы: изучить теоретические и практические приемы построения расписания методом Джонсона.

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Сформулируем классическую задачу теории расписаний – задачу Джонсона:

Пусть $N=1, 2, \dots, n$ - множество работ, которые выполняются на m машинах, каждая из которых имеет свой номер; пусть t_{ij} – время выполнения i -ой работы на j -ой машине; требуется так организовать выполнение работ, чтобы суммарное время простоя машин оказалось минимальным.

В этой постановке задачи предполагается, что

- 1) в любой момент времени на одной машине выполняется не более одной работы;
- 2) в любой момент времени одна работа выполняется не более, чем на одной машине.

Рассмотрим простейший случай задачи Джонсона – случай двух машин в последовательном варианте; это означает, что все работы выполняются только на одних и тех же двух машинах, причем все выполняются сначала на первой, а потом на второй машине.

Джонсону принадлежит алгоритм решения такой задачи, который мы сейчас сформулируем по шагам.

Шаг 0. Записывается матрица (t_{ij}) , $i=1, \dots, n, j=1, 2$ времен выполнения данных работ.

Шаг 1. В матрице (t_{ij}) фиксируется минимальный элемент (любой); если он оказался в первом столбце (машина №1), то соответствующую работу назначают первой к исполнению; если он оказался во втором

столбце (машина №2), то соответствующую работу назначают последней к исполнению.

Шаг 2. Исключают из дальнейшего рассмотрения время выполнения упорядоченной работы (т.е. вычеркивают из матрицы (t_{ij}) строку). Если после этого от матрицы (t_{ij}) ничего не остается, то задача решена; в противном случае переходим к Шагу 1.

Напомним, что описанный алгоритм минимизирует суммарное время простоя обеих машин.

Имеется обобщение алгоритма Джонсона для последовательного случая на случай произвольный, но для двух машин. А именно:

пусть по-прежнему $N=1, 2, \dots, n$ – множество работ, которые выполняются на двух машинах (№1 и №2), пусть по-прежнему t_{ij} – время выполнения i -ой работы на j -ой машине, но:

N_1 - подмножество работ, которые выполняются только на первой машине;

N_2 - подмножество работ, которые выполняются только на второй машине;

$N_{1,2}$ - подмножество работ, которые выполняются сначала на первой, а потом на второй машине;

$N_{2,1}$ - подмножество работ, которые выполняются сначала на второй, а потом - на первой машине.

Таким образом, $N = N_1 \cup N_2 \cup N_{1,2} \cup N_{2,1}$.

Можно доказать, что суммарный простой двух имеющихся машин будет минимальным, если организовать выполнение работ так:

1) в соответствии с алгоритмом Джонсона для последовательного варианта задачи для двух машин упорядочим работы из множеств $N_{1,2}$ и $N_{2,1}$;

2) на первую машину запускаются сначала работы из $N_{1,2}$, затем – из N_1 , а затем - из $N_{2,1}$;

3) на вторую машину запускаются сначала работы из $N_{2,1}$, затем – из N_2 , а затем - из $N_{1,2}$.

Рассмотренный общий случай задачи Джонсона для двух машин часто называют смешанным вариантом задачи Джонсона.

Утверждение №2. Пусть для матрицы (t_{ij}) выполнено хотя бы одно из двух следующих условий:

$$\min_i t_{i1} \geq \max_i t_{i2}$$

$$\min_i t_{i3} \geq \max_i t_{i2}$$

Построим новую матрицу (τ_{ij}) , в которой $i=1, 2, \dots, n, j=1, 2$ и $\tau_{i1} = t_{i1} + t_{i2}, \tau_{i2} = t_{i2} + t_{i3}$, и будем считать ее матрицей времен задачи Джонсона для двух машин в последовательном варианте и множества тех

же работ $N=1, 2, \dots, n$. Тогда суммарное время простоя исходных трех машин при выполнении исходных работ будет минимальным, если эти работы направлять на исполнение в том порядке, который является оптимальным в задаче с матрицей (τ_{ij}) .

Пример решения задачи расписаний указанным методом.

Таблица 10.1

Исходная таблица Работы

Работы	1	2	3	4	5
τ_{v1}	6	0	5	8	2
τ_{v2}	3	2	4	6	1
min1	2	-	-	-	-
min2	0	-	-	-	-

Поскольку условия теоремы не выполнены, требуется переставить 1-й и 2-й столбцы и так далее, пока не будет перестановок.

Общее время выполнения задания $T=23$ условные единицы. Получим результирующую таблицу.

Таблица 10.2

Результирующая таблица

Работы	1	2	3	4	5
τ_{v1}	0	8	5	6	2
τ_{v2}	2	6	4	3	1
min1	0	4	3	1	-
min2	2	5	4	2	-
T_{v1}	0	8	13	19	21
T_{v2}	8	14	19	22	23

Задание

Требуется построить расписание прохождения задач с заданной длительностью выполнения на каждом процессоре, минимизирующее общее время выполнение потока задач на двухпроцессорной системе (варианты см. таблицу 10.3). Используя карту Гантта определить время выполнения задания и сравнить с расчетным.

Таблица 10.2

Исходные данные задачи

Вариант	Номера работ Длительность	1	2	3	4	5	6	7	8
1	τ_1	1	8	2	3	4	7	5	1
	τ_2	5	3	4	4	2	9	15	4
2	τ_1	2	6	3	4	9	4	1	5
	τ_2	6	9	7	3	4	1	6	1
3	τ_1	6	9	7	3	4	1	6	1
	τ_2	2	4	5	1	6	2	3	9
4	τ_1	4	7	3	4	2	5	2	4
	τ_2	2	4	1	3	5	6	5	4
5	τ_1	3	7	9	5	1	2	2	7
	τ_2	4	5	5	7	8	7	5	4
6	τ_1	1	3	3	4	5	2	7	5
	τ_2	6	4	8	2	7	1	5	7
7	τ_1	2	4	5	1	6	2	3	9
	τ_2	4	7	3	4	2	5	2	4
8	τ_1	1	2	6	4	4	3	1	3
	τ_2	2	3	5	7	5	1	2	3
9	τ_1	2	4	3	2	6	3	1	2
	τ_2	3	6	5	5	4	1	5	7
10	τ_1	10	2	6	14	8	7	3	2
	τ_2	3	2	4	1	5	2	3	7
11	τ_1	1	8	3	3	4	7	5	1
	τ_2	6	3	4	4	2	11	15	4
12	τ_1	2	6	3	4	9	14	1	5
	τ_2	6	10	7	3	4	1	7	1

Требования к отчету

Отчет по лабораторной работе должен содержать:

- цель работы;
- номер варианта;
- исходные данные варианта;
- описание реализации алгоритма решения;
- результаты вычислений;
- выводы.

Контрольные вопросы

1. Задача Джонсона для двух и трех машин?
2. Алгоритм Джонсона.
3. Теорема Джонсона.
4. Построение расписания методом Джонсона?

ЛАБОРАТОРНАЯ РАБОТА №11

ПРИНЦИП ГАРАНТИРОВАННОГО РЕЗУЛЬТАТА В МАТРИЧНЫХ ИГРАХ, РЕШЕНИЕ МАТРИЧНОЙ ИГРЫ В СМЕШАННЫХ СТРАТЕГИЯХ.

Цель работы: получить практические навыки решения задач теории матричных игр.

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Общая постановка задачи игрового программирования.

Имеется M возможных решений (альтернатив): A_1, A_2, \dots, A_M . Известно, что каждое из решений может быть реализовано в одном из N вариантов внешних условий: B_1, B_2, \dots, B_N . Для каждого из решений известны его последствия (выигрыши стороны, принимающей решения) в каждом из вариантов внешних условий: E_{ij} , $i = 1, \dots, M, j = 1, \dots, N$. Эти выигрыши можно свести в таблицу, называемую матрицей выигрышей (E_{ij}) (или платежной матрицей). Такая матрица представляет собой математическую модель задачи.

Метод построения матрицы выигрышей полностью зависит от конкретных условий задачи.

Требуется выбрать наиболее эффективный вариант решения, т.е. одно из решений A_1, A_2, \dots, A_M или их комбинацию.

Пример. В игре принимают участие два игрока. Каждый из них может записать независимо от другого число 4, 5 или 6.

Если разность между числами, записанными игроками A и B :

– положительна, то игрок A выигрывает количество очков, равное этой разности.

– отрицательна, то выигрывает игрок B .

– равна нулю, то игра заканчивается вничью.

Необходимо составить *платежную матрицу игры*.

Решение

У игрока A имеется три стратегии:

A_1 - записать число 4;

A_2 - записать число 5;

A_3 - записать число 6.

Поэтому платежная матрица будет иметь три строки.

Таблица 11.1

Платёжная матрица для примера

Стратегии	$B_1(4)$	$B_2(5)$	$B_3(6)$
$A_1(4)$	0	-1	-2
$A_2(5)$	1	0	-1
$A_3(6)$	2	1	0

Игрок B также имеет три стратегии: B_1, B_2, B_3 (записать число 4, 5 или 6). Платежная матрица имеет три столбца.

В случае, если игрок A запишет число 4 (стратегия A_1 и игрок B также запишет 4 (стратегия B_1), то выигрыш игрока A составит $4 - 4 = 0$, т. е. элемент платежной матрицы $a_{11} = 0$. Аналогично рассчитываются все остальные элементы платежной матрицы.

Например, если игрок A выберет стратегию A_3 (запишет число 6), а игрок B выберет стратегию B_1 (запишет число 4), то выигрыш игрока A составит $a_{31} = 6 - 4 = 2$. Столько же проиграет игрок B .

Отрицательный выигрыш означает на самом деле проигрыш. Так, $a_{23} = -1$ означает, что если игрок A выберет стратегию A_2 (запишет 5), а игрок B выберет стратегию B_3 (запишет 6), то A выиграет -1 очко (т. е. проиграет 1 очко), а B проиграет -1 очко (т. е. выиграет 1 очко).

Задание

1. На основе описания правил матричной игры (Задача 1):
 - а) составить платежную матрицу;
 - б) найти решение игры.

Задача 1

В игре принимают участие два игрока: A и B . Каждый из игроков может записать независимо от другого игрока число a , $a+2$ или $a-3$. Если разность между числами, записанными игроками A и B , - четное число, то игрок A выигрывает количество очков, равное модулю этой разности. Если разность - нечетное число, то выигрывает игрок B . Если разность равна нулю, то игра заканчивается вничью.

Параметр a равен номеру варианта.

2. Решить в смешанных стратегиях матричную игру, сведя ее к задаче линейного программирования (ЗЛП). Полученную ЗЛП решить средствами Visual Studio (см. варианты матриц игры ниже).

$$\begin{array}{l}
 \text{№1} \begin{pmatrix} -3 & 2 & 2 & 1 & 3 \\ 4 & 0 & -1 & 5 & 5 \\ 2 & 4 & 3 & 2 & 1 \\ -4 & 4 & 2 & 1 & 2 \\ 1 & 3 & 3 & -2 & 1 \end{pmatrix} \\
 \text{№2} \begin{pmatrix} 7 & 3 & 9 & 8 & 9 \\ 2 & 0 & 8 & 7 & 8 \\ 6 & 1 & 6 & 2 & 9 \\ 5 & 4 & 2 & 3 & 8 \\ 6 & 4 & 6 & 1 & 8 \end{pmatrix} \\
 \text{№3} \begin{pmatrix} 6 & 9 & 2 & 10 & 5 \\ 5 & 6 & -3 & 8 & 13 \\ 4 & 6 & -1 & 7 & 5 \\ 3 & 3 & 3 & 3 & 3 \\ 7 & 8 & 7 & 9 & 2 \end{pmatrix} \\
 \text{№4} \begin{pmatrix} 3 & 3 & 5 & 3 & 3 \\ 2 & 3 & 4 & -1 & 2 \\ 2 & 0 & 2 & 2 & 3 \\ 4 & 4 & 0 & 5 & 5 \\ 4 & 4 & 4 & 0 & 2 \end{pmatrix} \\
 \text{№5} \begin{pmatrix} 3 & 3 & -2 & 6 & 4 \\ -1 & 2 & -3 & 6 & 2 \\ 2 & 1 & -6 & 2 & 1 \\ -3 & 5 & 1 & 4 & 6 \\ 2 & 4 & -4 & -1 & 4 \end{pmatrix} \\
 \text{№6} \begin{pmatrix} -1 & -4 & 0 & -2 & -2 \\ 7 & 4 & 0 & 1 & 4 \\ 5 & 3 & -2 & 0 & 3 \\ 6 & -6 & 7 & -3 & 9 \\ 4 & -7 & 7 & 1 & 6 \end{pmatrix}
 \end{array}$$

$$\begin{array}{l}
 \text{№7} \begin{pmatrix} 2 & -1 & 3 & 4 & 5 \\ 5 & 4 & -5 & 6 & -3 \\ -3 & 4 & 2 & -1 & 3 \\ 7 & 6 & -5 & 7 & -5 \\ -4 & 3 & 0 & -3 & 1 \end{pmatrix} \\
 \text{№9} \begin{pmatrix} -3 & -4 & -2 & -6 & -7 \\ 1 & 0 & -1 & 0 & -3 \\ -4 & -5 & -2 & -3 & -2 \\ 3 & -4 & 1 & 1 & -3 \\ -8 & -3 & -1 & -6 & -7 \end{pmatrix} \\
 \text{№11} \begin{pmatrix} 4 & 3 & 3 & -1 & 0 \\ -5 & 4 & 1 & 4 & 4 \\ -4 & 2 & 2 & 5 & 5 \\ 5 & -3 & -4 & 3 & 4 \\ 3 & 3 & 2 & -1 & 1 \end{pmatrix} \\
 \text{№8} \begin{pmatrix} -5 & 2 & -5 & 7 & 3 \\ 4 & 3 & 4 & 6 & 3 \\ -2 & 8 & 3 & 4 & 8 \\ 6 & 5 & 0 & 7 & 8 \\ 1 & 1 & 2 & 5 & 2 \end{pmatrix} \\
 \text{№10} \begin{pmatrix} 3 & -3 & 1 & -4 & -2 \\ 1 & 3 & -4 & 5 & 4 \\ 4 & -3 & 2 & -2 & -1 \\ -1 & 4 & 3 & 4 & 5 \\ 1 & 5 & -3 & 6 & 6 \end{pmatrix} \\
 \text{№12} \begin{pmatrix} 1 & -2 & 2 & 3 & -3 \\ 2 & -1 & 3 & 5 & 0 \\ -2 & 3 & 4 & 6 & 4 \\ 3 & 2 & -4 & -2 & 2 \\ -5 & 3 & 3 & 2 & -4 \end{pmatrix}
 \end{array}$$

3. Найти решение игры с природой (Задача 2). Получить решение, используя критерии: а) Байеса; б) Байеса – Лапласа; в) максиминный критерий Вальда; г) Сэвиджа; д) Гурвица.

Задача 2

Торговая организация реализует автомобили определенной марки. Объем реализации за месяц изменяется в зависимости от потребительского спроса в пределах от a до $a+10$ единиц. Если автомобилей меньше, чем требует спрос на них, то организация может заказать у завода-изготовителя недостающее количество. При этом возникнут дополнительные расходы в размере $20a$ условных единиц за каждый дополнительный автомобиль. А если количество автомобилей превышает спрос, то потери за невостребованный автомобиль составят $30a$. Прибыль от реализации одного автомобиля составляет: $200a$ – при $a \leq 15$; $50a$ – при $a > 15$.

Требуется определить, какое количество автомобилей выгоднее брать на реализацию на месяц.

Параметр a равен номеру варианта.

Требования к отчету

Отчет по лабораторной работе должен содержать:

- цель работы;
- номер варианта;
- исходные данные варианта;
- подробное описание выполнения всех заданий, согласно пп. 1-3, включая расчеты элементов матриц игр, математические модели ЗЛП, копии экранов документов с подробными комментариями по выполняемым действиям и выводами на основе полученных результатов об оптимальных стратегиях для каждой из задач.
- выводы.

Контрольные вопросы

1. Назвать основные признаки классификации игр.
2. Какие бывают игры в зависимости от числа игроков и количества стратегий?
3. Какие бывают игры в зависимости от взаимоотношений игроков?
4. Общая постановка задачи для матричной игры.
5. Что такое платежная матрица игры?
6. Приведите пример расчета платежной матрицы для простой матричной игры.
7. Дайте определения терминов: «седловая точка», «чистая цена игры».
8. Дайте определения терминов: «чистая стратегия», «смешанная стратегия».
9. Как свести игру в смешанных стратегиях к ЗЛП?
10. Дайте определение термина: «игра с природой»

СПИСОК ИСТОЧНИКОВ

1. Костюкова, О.И. Исследование операций. Учебное пособие для студентов специальности 31.03.04 «Информатика» всех форм обучения / О.И. Костюкова. – Мн.: БГУИР, 2003.
2. Ларичев О.И. Теория и методы принятия решений. М.: Логос, 2000.
3. Черников, Ю.Г. Системный анализ и исследование операций / Ю.Г. Черников. – М.: Изд-во МГГУ, 2006.

Богданова Наталья Сергеевна

**СИСТЕМНЫЙ АНАЛИЗ
И ИССЛЕДОВАНИЕ ОПЕРАЦИЙ**

**Практикум
по выполнению лабораторных работ для студентов
специальности 1-40 04 01 «Информатика
и технологии программирования»
дневной формы обучения**

Подписано к размещению в электронную библиотеку
ГГТУ им. П. О. Сухого в качестве электронного
учебно-методического документа 18.02.20.

Рег. № 46Е.
<http://www.gstu.by>