



федеральное государственное бюджетное образовательное  
учреждение высшего образования  
**«Национальный исследовательский университет «МЭИ»**

---

Институт информационных и вычислительных технологий  
Кафедра управления и интеллектуальных технологий

**Отчёт по лабораторной работе №1**  
**По дисциплине «Управление в больших системах»**  
**«Синтез больших систем управления. Распределение задач по узлам управления»**

Выполнил студент: Михайловский М. Ю.

Группа: А-03-21

Вариант: 5

Проверили: Новиков В. Н, Обычайко Д. С.

Москва 2024

# Содержание

<b>1</b>	<b>Постановка задачи</b>	<b>3</b>
<b>2</b>	<b>Алгоритмы решения задачи</b>	<b>4</b>
2.1	Первая оптимизация . . . . .	4
2.2	Вторая оптимизация . . . . .	4
2.3	Оптимизация методом ветвления . . . . .	5
<b>3</b>	<b>Реализация программы</b>	<b>7</b>
3.1	Описание модулей программы . . . . .	7
3.2	Пример работы программы . . . . .	7
<b>А</b>	<b>Блок-схемы алгоритмов</b>	<b>11</b>
<b>Б</b>	<b>Листинги файлов программы</b>	<b>13</b>

# 1 Постановка задачи

Имеется  $I$  задач, которые должны быть решены последовательно друг за другом. Для их решения имеется  $J$  узлов. Затраты и время решения  $i$ -ой задачи на  $j$ -ом узле заданы соответственно матрицами стоимости затрат  $C = [c_{ij}]$  и временных затрат  $T = [t_{ij}]$ .

$$C = \begin{bmatrix} 4,5 & 7 & 2 & 2 \\ 5 & 8 & 1 & 3 \\ 5,5 & 9 & 6 & 2 \\ 6 & 10 & 7 & 1 \\ 6,5 & 7 & 3 & 1 \end{bmatrix}, \quad T = \begin{bmatrix} 4,5 & 3 & 2 & 9 \\ 5 & 6 & 5 & 10 \\ 5,5 & 7 & 6 & 11 \\ 6 & 8 & 7 & 12 \\ 6,5 & 9 & 8 & 5 \end{bmatrix} \quad (1)$$

Требуется минимизировать суммарную стоимость затрат при заданном ограничении на суммарные временные затраты  $T_3 = 25$ . Для записи оптимизационной задачи используем  $x_{ij}$ :

$$x_{ij} = \begin{cases} 1, & i\text{-ая задача решается на } j\text{-ом узле} \\ 0, & \text{иначе} \end{cases}.$$

Тогда задача оптимизации примет вид:

$$\begin{cases} \sum_{i=1}^I \sum_{j=1}^J c_{ij} x_{ij} \rightarrow \min \\ \sum_{i=1}^I \sum_{j=1}^J t_{ij} x_{ij} \leq T_3 \\ \sum_{j=1}^J x_{ij} = 1, \forall i : i = \overline{1, I} \end{cases} \quad (2)$$

Для проведения оптимизации будет написана программа с пользовательским интерфейсом для ввода исходных данных.

## 2 Алгоритмы решения задачи

Решать задачу мы будем в несколько этапов. Сначала над матрицами затрат будут проведены две процедуры оптимизации, которые исключают заведомо неоптимальные узлы для использования.

Качественно алгоритмы также представлены в приложении.

### 2.1 Первая оптимизация

Оптимизация проводится построчно, пусть алгоритм находится на  $i$ -ой строке. Фиксируется номер узла  $s$ , на котором стоимость затрат наименьшая:

$$s = \arg \min_j c_{ij}$$

Затем, исключаются те элементы строки, которые удовлетворяют условию (3). При чём ровно одно из этих неравенств может выполняться нестрого.

$$\begin{aligned} c_{ij} &> c_{is} \\ t_{ij} &> t_{is} \end{aligned} \quad (3)$$

Так мы исключаем узлы, которые не лучше зафиксированного  $s$ -ого узла ни по стоимости, ни по времени затрат. В результате такой процедуры данные матрицы (1) преобразуются к следующему виду:

$$C^{(0)} = \begin{bmatrix} - & - & 2 & - \\ - & - & 1 & - \\ 5,5 & 9 & 6 & 2 \\ 6 & 10 & 7 & 1 \\ - & - & - & 1 \end{bmatrix}, \quad T^{(0)} = \begin{bmatrix} - & - & 2 & - \\ - & - & 5 & - \\ 5,5 & 7 & 6 & 11 \\ 6 & 8 & 7 & 12 \\ - & - & - & 5 \end{bmatrix} \quad (4)$$

### 2.2 Вторая оптимизация

Вторая оптимизация так же как и первая проводится построчно. Будем рассматривать её для фиксированной  $i$ -ой строки. Эта оптимизация исключает те узлы, использование которых напрямую приводит к нарушению временного ограничения  $T_3$ .

Для этого используется вектор наименьших временных затрат на узлах системы:

$$T_{\min} = \left( \min_j t_{1j} \quad \min_j t_{2j} \quad \dots \quad \min_j t_{Ij} \right)^T$$

То есть теоретически наименьшее время решения всех  $I$  задач будет равно:

$$T_{\text{теор.мин}} = \|T_{\min}\|_1 = \sum_{i=1}^I T_{\min i}$$

Тогда для данной  $i$ -ой строки  $j$ -ый элемент исключается если выполняется условие (5). Это означает, что при решении  $i$ -ой задачи на  $j$ -ом узле, в любом случае будет нарушено временное ограничение  $T_3$ .

$$T_{\text{теор.мин}} - T_{\min i} + t_{ij} > T_3 \quad (5)$$

В результате такой оптимизации матрицы (4) приобретают следующий вид:

$$C^{(1)} = \begin{bmatrix} - & - & 2 & - \\ - & - & 1 & - \\ 5,5 & 9 & 6 & - \\ 6 & - & 7 & - \\ - & - & - & 1 \end{bmatrix}, \quad T^{(1)} = \begin{bmatrix} - & - & 2 & - \\ - & - & 5 & - \\ 5,5 & 7 & 6 & - \\ 6 & - & 7 & - \\ - & - & - & 5 \end{bmatrix} \quad (6)$$

В результате в каждой  $i$ -ой строке остаётся некоторое допустимое множество узлов для использования  $J_{\text{доп}i} \subseteq J$ .

## 2.3 Оптимизация методом ветвления

В самом начале фиксируются два различных решения задачи, доставляющие минимумы суммарным стоимостям затрат  $C_{\min}$  и временным затратам  $T_{\min}$ .

$$C_{\min} = \left( \min_j c_{1j} \quad \min_j c_{2j} \quad \dots \quad \min_j c_{Ij} \right)^T$$

$$T_{\min} = \left( \min_j t_{1j} \quad \min_j t_{2j} \quad \dots \quad \min_j t_{Ij} \right)^T$$

Для них рассчитываем теоретические минимумы. Этими двумя метриками и будем определять показатели каждой вершины дерева в методе ветвления.

$$C_{\text{теор мин}} = \|C_{\min}\|_1 = 15,5, \quad T_{\text{теор мин}} = \|T_{\min}\|_1 = 23,5.$$

Соотнесём им корень дерева  $v^0$ . Пусть корень дерева будет на нулевом уровне  $l = 0$ . Тогда ему будут смежны вершины  $v_j^1$ ,  $j \in J_{\text{доп}i}$ , которые будут на первом уровне  $l = 1$ .

И введём вектор номеров узлов соответствующих вершинам, которые со-

ставляют путь до вершины  $v_{jl}^l$ :

$$J(v_{jl}^l) = (j_1 \ j_2 \ \dots \ j_l)^T, \ l \leq J$$

Тогда на каждой строке матрицы  $i$  или уровне дерева  $l$  ( $i = l$ ) будем проводить описанную далее процедуру.

1. Рассчитываем для вершин  $v_j^l$ ,  $j \in J_{\text{доп } l}$  метрики  $\|C_{\text{метр } j}\|_1$ ,  $\|T_{\text{метр } j}\|_1$ . Здесь  $C_{\text{метр } j}$  вектор, где первые  $l$  элементов берутся  $J(v_j^l)$ , все последующие элементы берутся  $C_{\text{мин}}$ . Метрика временных затрат  $T_{\text{метр } j}$  определяется аналогично.
2. Сравниваем эти вершины по их метрикам. Искключаем из рассмотрения те вершины, для которых метрика временных затрат  $T_{\text{метр } j} > T_3$ . Из оставшихся вершин выбираем вершину с наименьшей метрикой затрат стоимости  $C_{\text{метр } j}$ .
3. Вершины следующего уровня  $v_j^{l+1}$ ,  $j \in J_{\text{доп } l}$  делаем смежными выбранной вершине  $v_j^l$ .

В результате полученное дерево будет иметь вид представленный на рис. 2.1. В результате данного алгоритма на каждом уровне выбирается единственная вершина.

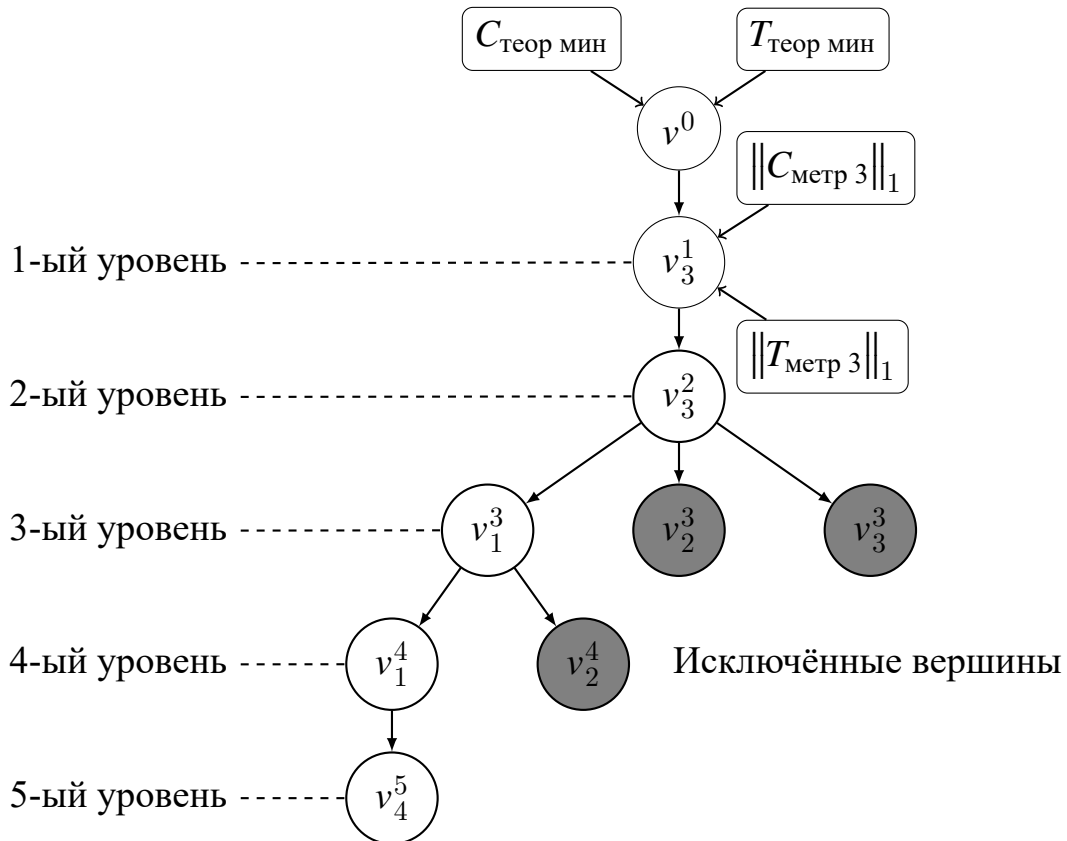


Рис. 2.1. Пример дерева ветвления

## 3 Реализация программы

Программа написана на языке python 3.11.3. Используются следующие библиотеки: *numpy=2.1.1*, *PyQt6=6.7.1*, *PyQt6-Qt6=6.7.3*, *PyQt6\_sip=13.8.0*.

### 3.1 Описание модулей программы

Программа состоит из 5 основных файлов. Опишем их основные назначения:

- *main.py* – основной модуль. Именно он запускает программу. В нём объявляются основные объекты классов из других модулей и привязываются функции для взаимодействия с интерфейсом программы
- *tableHandlers.py* – модуль, который содержит класс для работы с таблицами в интерфейсе. Он реализует такие основные функции, как чтение таблицы, установление стандартных значений в неё, проверка корректности введённых данных и перекрашивание ячеек в соответствии с заданной маской.
- *userInfo.py* – отдельный модуль, который сделан специально для считывания данных из таблицы с уведомлением пользователя об ошибках в введенных данных.
- *optimizer.py* – в этом модуле реализуется два класса решающих задачу оптимизации. **MatrixOptimizer** – класс, который занимается процедурами первой, второй оптимизации и метода ветвления. **Tree** – класс, который реализует граф в виде дерева.
- *main\_window.ui* – файл с основной разметкой пользовательского интерфейса, сгенерированный в Qt Designer.

Листинги этих файлов приведены в приложении.

### 3.2 Пример работы программы

При запуске программы открывается следующее окно: рис. 3.1.

Здесь в интерактивных таблицах можно ввести исходные данные. При введении некорректных исходных данных показывает окно с предупреждением о неверности введённых данных, рис. 3.2. При проверке введенных данных выполняется несколько проверок, среди них:

Распределение задач по узлам управления

Введите данные

Матрица C

	1	2	3	4
1	4.5	7.0	2.0	2.0
2	5.0	8.0	1.0	3.0
3	5.5	9.0	6.0	2.0
4	6.0	10.0	7.0	1.0
5	6.5	7.0	3.0	1.0

Матрица T

	1	2	3	4
1	4.5	3.0	2.0	9.0
2	5.0	6.0	5.0	10.0
3	5.5	7.0	6.0	11.0
4	6.0	8.0	7.0	12.0
5	6.5	9.0	8.0	5.0

Значение  $T_z$  25

Ввод данных

1 оптимизация 2 оптимизация Построить дерево решений

Рис. 3.1. Окно при запуске программы

- Неотрицательность данных;
- Введенные значения являются числами;
- Введенные данные не пустые;
- Введенное значение  $T_z$  больше чем теоретический минимум  $T_{\text{теор. мин.}}$ .

После нажатия кнопки ввод данных исходные значения фиксируются и становятся недоступными для редактирования, рис. 3.3. Становятся доступными кнопки первой и второй оптимизации и построения дерева решений.

Введите данные

Матрица C

	1	2	3	4
1	4.5	7.0	2.0	2.0
2	5.0	8.0	1.0	3.0
3	5.5	9.0	6.0	2.0
4	6.0	10.0	7.0	-1.0
5	6.5	7.0	3.0	1.0

Матрица T

	1	2	3	4
1	4.5	3.0	2.0	9.0
2	5.0	6.0	5.0	10.0
3	5.5	7.0	6.0	11.0
4	6.0	8.0	7.0	12.0
5	6.5	9.0	8.0	5.0

python

Введены неверные данные!  
Значение не может быть отрицательным.

OK

1 оптимизация 2 оптимизация Построить дерево решений

Рис. 3.2. Предупреждение о неверно введенных данных



Выберите действие

Матрица С					Матрица Т				
	1	2	3	4		1	2	3	4
1	4.5	7.0	2.0	2.0	1	4.5	3.0	2.0	9.0
2	5.0	8.0	1.0	3.0	2	5.0	6.0	5.0	10.0
3	5.5	9.0	6.0	2.0	3	5.5	7.0	6.0	11.0
4	6.0	10.0	7.0	1.0	4	6.0	8.0	7.0	12.0
5	6.5	7.0	3.0	1.0	5	6.5	9.0	8.0	5.0

Значение Tz:

Рис. 3.3. После нажатия кнопки ввод данных

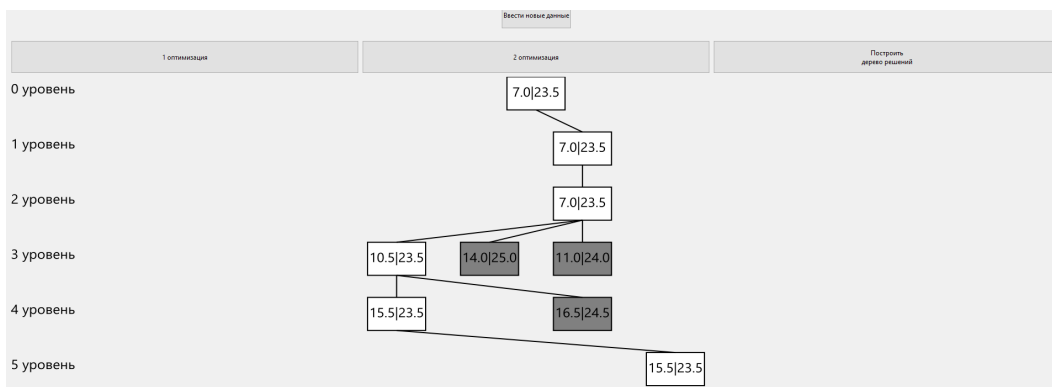


Рис. 3.5. Построенное дерево решений

При проведении любой из оптимизаций значения, которые были исключены в результате выполнения алгоритма окрашиваются в красный цвет, рис. 3.4.

а)

б)

Рис. 3.4. Результаты проведения а) первой, б) второй после первой оптимизаций

Все три операции в нижней части окна можно производить в произвольном порядке. Оптимизации лишь упрощают дальнейшее построение дерева и не зависят друг от друга, поэтому могут даже и не использоваться.

При нажатии кнопки построения дерева решений происходит последний этап оптимизации и в нижней части окна выводится получившееся дерево, рис. 3.5.

В приложении выводится сформированное *svg*-изображение. Оно также сохраняется в папке с исходными файлами программы в файле *out.svg*. На изображении вершины дерева упорядочены в соответствии с номером узла, который

каждая из них представляют. Собрав последовательность узлов, которые не закрашены серым мы получаем ответ, на задачу оптимизации, данный в результате выполнения алгоритма.

В данном случае это последовательность узлов  $3 - 3 - 1 - 1 - 4$ .

Если необходимо провести исследование при других исходных данных, то пользователь может нажать кнопку *ввести новые данные* и вернуться к этапу, когда он только открыл программу. Построенное изображение дерева останется на экране, и будет обновлено, если пользователь построит новое дерево, в том числе при других исходных данных.

## Приложение А. Блок-схемы алгоритмов

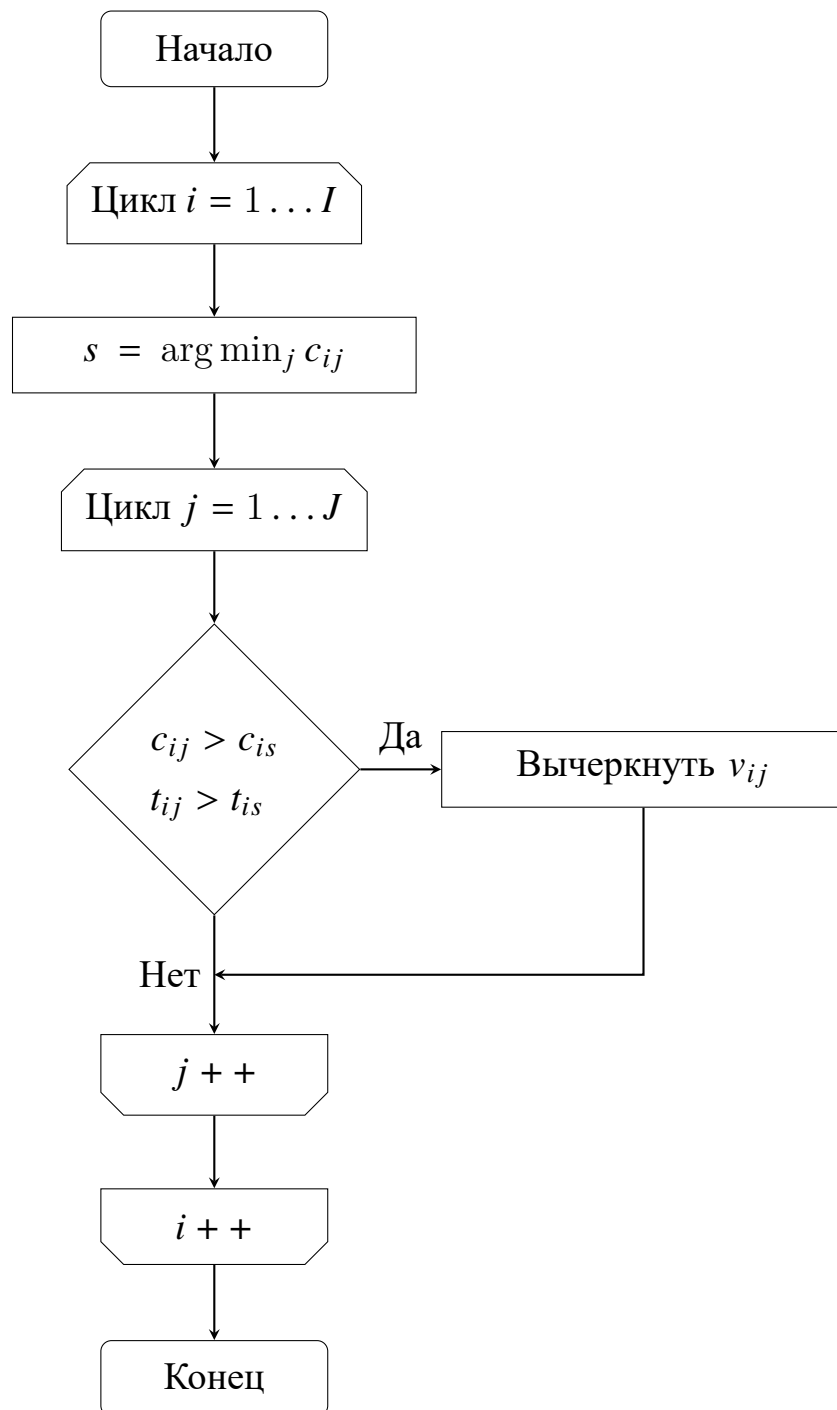


Рис. А.1. Первая оптимизация

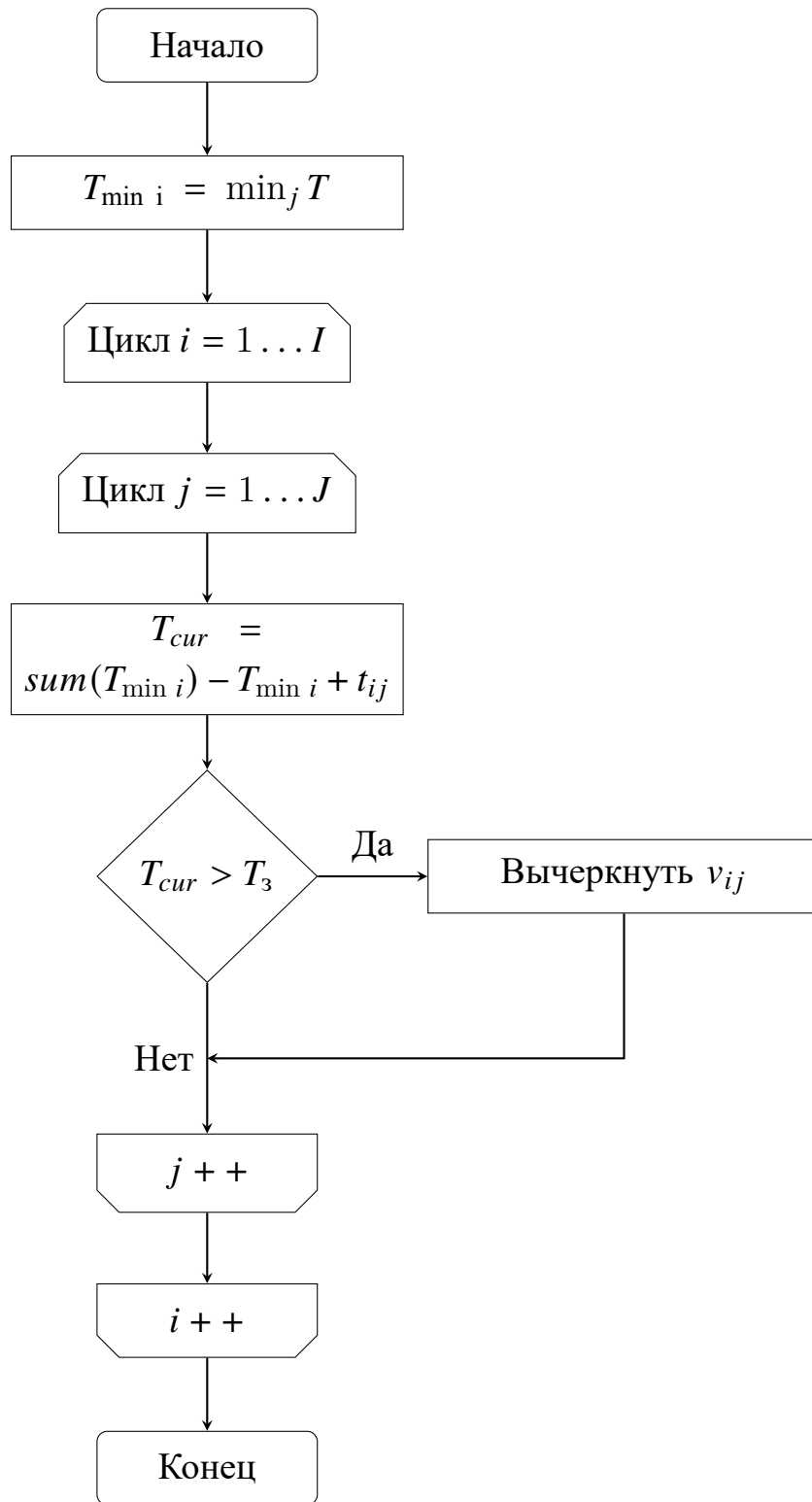


Рис. А.2. Вторая оптимизация

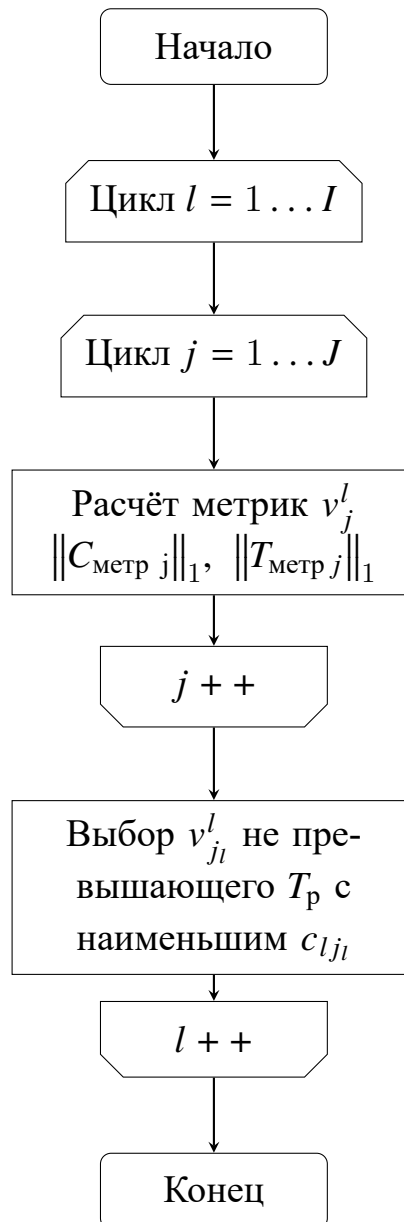


Рис. А.3. Метод ветвления

## Приложение Б. Листинги файлов программы

### Листинг 1. main.py

```

1  from PyQt6 import uic
2  from PyQt6.QtWidgets import QApplication, QPushButton, QLabel, QScrollArea
3  from PyQt6.QtCore import QRegularExpression
4  from PyQt6.QtSvg import QSvgRenderer
5  from PyQt6.QtGui import QRegularExpressionValidator, QPainter
6  from userInfo import DataGetter
7  import numpy as np
8
9  cMatrixDefault = np.array([
10     [4.5, 7, 2, 2],

```

```

11     [5, 8, 1, 3],
12     [5.5, 9, 6, 2],
13     [6, 10, 7, 1],
14     [6.5, 7, 3, 1]
15 ], dtype=np.float16)
16 tMatrixDefault = np.array([
17     [4.5, 3, 2, 9],
18     [5, 6, 5, 10],
19     [5.5, 7, 6, 11],
20     [6, 8, 7, 12],
21     [6.5, 9, 8, 5]
22 ], dtype=np.float16)
23 max_T_default = 25
24
25 def InputBtnClick(dataGetter: DataGetter, matrixOptimizer, inputBtn: QPushButton,
26     ↪ instructionLabel: QLabel, actionBtns: list):
27     modes = {
28         'input': 'Ввод данных',
29         'reset': 'Ввести новые данные'
30     }
31     instructionTexts = {
32         'input': 'Введите данные',
33         'reset': 'Выберите действие'
34     }
35     if dataGetter.inputBtnMode == 'input':
36         gotError: bool = dataGetter.catch_input_errors()
37         if not gotError:
38             [ tableHandler.table.setEnabled(False) for tableHandler in
39                 ↪ dataGetter.tables.values()]
40             [ lineEdit.setEnabled(False) for lineEdit in
41                 ↪ dataGetter.lineEdits.values()]
42             [ btn.setEnabled(True) for btn in actionBtns]
43             inputBtn.setText(modes['reset'])
44             dataGetter.inputBtnMode = 'reset'
45             instructionLabel.setText(instructionTexts['reset'])
46             matrixOptimizer.max_T = float(dataGetter.lineEditsTexts['Значение
47                 ↪ T3'])
48
49     return
50
51 if dataGetter.inputBtnMode == 'reset':
52     [ tableHandler.table.setEnabled(True) for tableHandler in
53         ↪ dataGetter.tables.values()]
54     [ tableHandler.decolorize_cells() for tableHandler in
55         ↪ dataGetter.tables.values()]
56     [ lineEdit.setEnabled(True) for lineEdit in dataGetter.lineEdits.values()]
57     [ btn.setEnabled(False) for btn in actionBtns]
58     inputBtn.setText(modes['input'])
59     dataGetter.inputBtnMode = 'input'
60     instructionLabel.setText(instructionTexts['input'])
61     return

```

```

59 def main():
60     Form, Window = uic.loadUiType("main_window.ui")
61     app = QApplication([])
62     window = Window()
63     form = Form()
64     form.setupUi(window)
65
66     from tableHandlers import TableHandler
67     cTable = TableHandler(form.cMatrix, cMatrixDefault)
68     tTable = TableHandler(form.tMatrix, tMatrixDefault)
69     cTable.table.itemChanged.connect(TableHandler.floatValidateAndMessage)
70     tTable.table.itemChanged.connect(TableHandler.floatValidateAndMessage)
71
72     from optimizer import MatrixOptimizer
73     matrixOptimizer = MatrixOptimizer(cTable, tTable, max_T_default)
74     dataGetter = DataGetter(tables={"Матрица C": cTable, "Матрица T": tTable},
75     ↪ lineEdits={"Значение Tз": form.T_max_lEdit},
76                                     lineEditsLinkedTables={"Значение Tз": 'Матрица T'})
77
78     form.dataInput.clicked.connect(lambda: (
79         InputBtnClick(dataGetter, matrixOptimizer, form.dataInput,
80         ↪ form.instructionLabel, [form.optimize1, form.optimize2,
81         ↪ form.decisionTreeBtn]),
82         matrixOptimizer.refreshValues()
83     ))
84
85     form.optimize1.clicked.connect(lambda: (
86         matrixOptimizer.optimization1(),
87         matrixOptimizer.cTable.colorize_cells(matrixOptimizer.eliminated),
88         matrixOptimizer.tTable.colorize_cells(matrixOptimizer.eliminated)
89     ))
90
91     form.optimize2.clicked.connect(lambda: (
92         matrixOptimizer.optimization2(),
93         matrixOptimizer.cTable.colorize_cells(matrixOptimizer.eliminated),
94         matrixOptimizer.tTable.colorize_cells(matrixOptimizer.eliminated)
95     ))
96
97     form.decisionTreeBtn.clicked.connect(lambda: (
98         matrixOptimizer.tree_optimization(),
99         form.svgView.update()
100     ))
101
102     validator = QRegularExpressionValidator(QRegularExpression(r'[0-9]+\.[0-9]*'))
103     form.T_max_lEdit.setValidator(validator)
104     form.T_max_lEdit.setText(str(max_T_default))
105
106     def showSVG(event):
107         if matrixOptimizer.treeDone:
108             svgRenderer = QSvgRenderer('out.svg')
109             painter = QPainter(form.svgView)
110             svgRenderer.render(painter)
111         form.svgView.paintEvent = showSVG
112     window.show()
113     app.exec()
114
115 if __name__ == '__main__':

```

---

```
110 main()
```

---

## Листинг 2. userInfo.py

---

```
1 from PyQt6.QtWidgets import QMessageBox
2
3 def showMessageBox(title: str, description: str, icon: QMessageBox.Icon):
4     msgBox = QMessageBox()
5     msgBox.setText(title)
6     msgBox.setInformativeText(description)
7     msgBox.setIcon(icon)
8     msgBox.exec()
9
10 class TemplateMessageBox(QMessageBox):
11     def __init__(self, title: str, description: str, icon: QMessageBox.Icon):
12         super().__init__()
13         self.setText(title)
14         self.setInformativeText(description)
15         self.setIcon(icon)
16
17 class DataGetter:
18     def __init__(self, tables: dict, lineEdits: dict, lineEditsLinkedTables: dict
19         ↪ = {}):
20         '''
21         tables -> dict(str: TableHandler)
22         lineEdits -> dict(str: QLineEdit)
23         '''
24         self.tables = tables
25         self.lineEdits = lineEdits
26         self.data_good = False
27         self.lineEditsTexts = dict()
28         self.lineEditsLinkedTables = lineEditsLinkedTables
29         self.inputBtnMode = 'input'
30
31     def _get_and_check(self):
32         '''
33         Считывает данные из полей для ввода и выполняет проверки на корректность
34         ↪ введённых данных
35         '''
36         for name, table in self.tables.items():
37             table.toNumpy()
38             if not table.data_good:
39                 raise ValueError(f"{name}. Неверно введённые данные")
40
41         for name, lineEdit in self.lineEdits.items():
42             text = lineEdit.text()
43             if len(text) == 0:
44                 raise ValueError(f"{name}. Должно быть введено значение")
45             self.lineEditsTexts[name] = text
46
47         if name not in self.lineEditsLinkedTables:
```



---

```

47         return
48
49         table_name = self.lineEditsLinkedTables[name]
50         theorMin = self.tables[table_name].theorMin
51         if float(text) < theorMin:
52             raise ValueError(f"{name}. Значение не может быть меньше
53                 ↳ теоретического минимума равного {theorMin}")
54
55 def catch_input_errors(self) -> bool:
56     try:
57         self._get_and_check()
58         msg = TemplateMessageBox("Данные успешно введены", "",
59             ↳ QMessageBox.Icon.Information)
60         msg.exec()
61         return False
62     except ValueError as e:
63         msg = TemplateMessageBox("Неверно введённые данные", str(e),
64             ↳ QMessageBox.Icon.Warning)
65         msg.exec()
66         return True

```

---

### Листинг 3. tableHandlers.py

---

```

1  from PyQt6.QtWidgets import QTableWidgetItem, QTableWidgetItem, QMessageBox
2  from PyQt6.QtGui import QColor
3  from userInfo import TemplateMessageBox
4  import numpy as np
5
6  class TableHandler:
7      valueErrorTitle: str = 'Введены неверные данные!'
8      valueErrorMessages: dict = {
9          'empty_cell': TemplateMessageBox(valueErrorTitle, 'Заполните все ячейки
10              ↳ матрицы.', QMessageBox.Icon.Warning),
11          'non_float': TemplateMessageBox(valueErrorTitle, 'Все значения должны быть
12              ↳ числовыми.', QMessageBox.Icon.Warning),
13          'negative_value': TemplateMessageBox(valueErrorTitle, 'Значение не может
14              ↳ быть отрицательным.', QMessageBox.Icon.Warning)
15      }
16
17 def __init__(self, table: QTableWidgetItem, defaultValues: np.array = None):
18     self.table: QTableWidgetItem = table
19     self.rows = self.table.rowCount()
20     self.columns = self.table.columnCount()
21     self.matrix: np.ndarray = defaultValues
22     self.data_good = True
23     self.theorMin = None
24
25     if self.matrix is None:
26         self.data_good = False
27         return
28
29     if (self.rows, self.columns) != self.matrix.shape:

```

```

27         raise ValueError("Table shape does not match passed defaultValues:
           ↳ np.array shape")
28
29     self.toTable(self.matrix)
30
31     @staticmethod
32     def floatValidate(item: QTableWidgetItem):
33         '''
34         Валидирует значение в соответствии со следующими критериями: значение
           ↳ непустое, вещественное типа float, неотрицательное. Возвращает код
           ↳ результата проверки и само проверенное значение, если оно прошло
           ↳ валидацию.
35         '''
36         if not item:
37             return ('empty_cell', None)
38
39         try:
40             item_f = float(item.text())
41         except ValueError:
42             return ('non_float', None)
43
44         if item_f < 0:
45             return ('negative_value', None)
46
47         return ('good', item_f)
48
49     @staticmethod
50     def floatValidateAndMessage(item: QTableWidgetItem):
51         '''
52         Валидирует введённое значение и выводит сообщение, если значение не прошло
           ↳ валидацию. Проверяемые критерии определяются floatValidate()
53         '''
54         status, value = TableHandler.floatValidate(item)
55         if status != 'good':
56             TableHandler.valueErrorMessages[status].exec()
57             return ('err', None)
58
59         return ('good', value)
60
61     def _calcTheorMin(self):
62         '''
63         Расчёт теоретического суммарного минимума для набора из единичных значений
           ↳ из каждой строки
64         '''
65         self.theorMin = sum(self.matrix.min(1))
66
67     def toNumpy(self):
68         '''
69         Запись значений из таблицы QTableWidgetItem в матрицу np.array
70         '''
71         matrix = []
72         for i in range(self.table.rowCount()):
73             row = []
74             for j in range(self.table.columnCount()):

```

```

75         item = self.table.item(i, j)
76
77         status, value = TableHandler.floatValidateAndMessage(item)
78         if status == 'err':
79             self.table.setCurrentCell(i, j)
80             self.data_good = False
81             return None
82
83         row += [value,]
84         matrix += [row,]
85
86         np_matrix = np.array(matrix, dtype=np.float16)
87         self.matrix = np_matrix
88         self.data_good = True
89         self._calcTheorMin()
90         return np_matrix
91
92     def toTable(self, matrix: np.array):
93         """
94         Запись значений из матрицы np.array в таблицу QTableWidgetItem
95         """
96         if (self.rows, self.columns) != matrix.shape:
97             raise ValueError("Table shape does not match passed np.array shape")
98
99         for i in range(self.rows):
100             for j in range(self.columns):
101                 value = QTableWidgetItem(str(matrix[i,j]))
102                 self.table.setItem(i, j, value)
103
104     def colorize_cells(self, mask: np.array):
105         if (self.table.rowCount(), self.table.columnCount()) != mask.shape:
106             raise ValueError("Table shape does not match passed np.array shape")
107
108         for i in range(self.table.rowCount()):
109             for j in range(self.table.columnCount()):
110                 if mask[i,j] == True:
111                     self.table.item(i, j).setBackground(QColor(200,0,0))
112
113     def decolorize_cells(self):
114         for i in range(self.table.rowCount()):
115             for j in range(self.table.columnCount()):
116                 self.table.item(i, j).setBackground(QColor(255,255,255))

```

#### Листинг 4. optimizer.py

```

1 from PyQt6.QtWidgets import QMessageBox
2 from tableHandlers import TableHandler
3 import numpy as np
4
5 class Tree:
6     def __init__(self, parent: 'Tree' = None, fixed_nodes: list = []):
7         self.parent: Tree = parent

```

```

8         self.children = dict()
9         self.fixed_nodes = fixed_nodes
10        self.status_good = True
11
12    def add_child(self, fixed_nodes, c_sum, t_sum) -> 'Tree':
13        parent_node = self.find_child(fixed_nodes[:-1])
14        child_node = Tree(parent_node, fixed_nodes.copy())
15        child_node.set_metrics(c_sum, t_sum)
16        last_node = fixed_nodes[-1]
17        parent_node.children[last_node] = child_node
18
19        return child_node
20
21    def find_child(self, fixed_nodes) -> 'Tree':
22        if self.fixed_nodes == fixed_nodes:
23            return self
24
25        next_node = fixed_nodes[len(self.fixed_nodes)]
26        res = self.children[next_node].find_child(fixed_nodes)
27        return res
28
29    def set_metrics(self, c_sum, t_sum):
30        self.c_sum = c_sum
31        self.t_sum = t_sum
32
33    def set_status(self, status_good: bool):
34        self.status_good = status_good
35
36    def print_node(self):
37        print(f'{{self.c_sum:5>}}|{{self.t_sum:5>}} good:{{repr(self.status_good)}}')
38
39    def print_tree(self, level = 1):
40        if level == 1:
41            self.print_node()
42        for node, child in self.children.items():
43            print('| '*level + f'{{node}}' + '-->', end='')
44            child.print_node()
45            child.print_tree(level+1)
46
47    def _calc_x_for_node(self, x_place):
48        x_offset = (x_place - 1.5) * self.nodes_x_step
49        x_offset + (self.image_width-self.node_width)/2
50        x = x_offset + (self.image_width-self.node_width)/2
51        return x
52
53    def _calc_x_for_text(self, x_place):
54        x_offset = (x_place - 1.5) * self.nodes_x_step
55        x = x_offset + self.image_width/2
56        return x
57
58    def nodeSvg(self, node: 'Tree', x_place=1.5, level=0):
59        if node.status_good:
60            color = 'white'
61        else:

```

```

62         color = 'gray'
63         y_baseline = level * self.level_step
64
65         svg = f'''<rect x='{self._calc_x_for_node(x_place)}' y='{y_baseline}'
        ↳ width='{self.node_width}' height = '{self.node_height}' stroke='black'
        ↳ fill='{color}'/>
66 <text x='{self._calc_x_for_text(x_place)}' y='{y_baseline + self.node_height/2+3}'
        ↳ dominant-baseline='middle' text-anchor='middle'
        ↳ font-size='{self.f_size}'>{node.c_sum}|{node.t_sum}</text>\n'''
67         return svg
68
69     def nodesLevelSvg(self, children, level):
70         y_baseline = level * self.level_step
71         svg = f'''<text x='0' y='{y_baseline + self.node_height/2}'
        ↳ font-size='{self.f_size}'>{level} уровень</text>\n''
72         for j, node in children.items():
73             svg += self.nodeSvg(node, j, level)
74         return svg
75
76
77     def toSvg(self, file_name: str = 'out.svg'):
78         self.image_width = 800*1.13
79         self.image_height = 250*1.13
80         self.node_width = 50
81         self.node_height = 30
82         self.f_size = '12px'
83         self.level_step = 50
84         self.nodes_x_step = self.node_width + 30
85
86         svg = f'''<?xml version = '1.0' encoding='UTF-8'?>
87 <svg width='{self.image_width}' height='{self.image_height}' viewBox='0 0
        ↳ {self.image_width} {self.image_height}' xmlns='http://www.w3.org/2000/svg'>
88 <text x='0' y='{self.node_height/2}' font-size='{self.f_size}'>0 уровень</text>
89 '''
90         svg += self.nodeSvg(self)
91         valid_node = self
92         level = 1
93         while valid_node.children:
94             svg += self.nodesLevelSvg(valid_node.children, level)
95             level += 1
96             y_baseline = (level-1) * self.level_step
97             parent_y = y_baseline - self.level_step + self.node_height
98             if valid_node.fixed_nodes:
99                 parent_x = self._calc_x_for_text(valid_node.fixed_nodes[-1])
100             else:
101                 parent_x = self._calc_x_for_text(1.5)
102
103             for node in valid_node.children.values():
104                 child_y = y_baseline
105                 child_x = self._calc_x_for_text(node.fixed_nodes[-1])
106                 svg += f'''<line x1='{parent_x}' y1='{parent_y}' x2='{child_x}'
        ↳ y2='{child_y}' stroke='black' />\n''
107                 if node.status_good:
108                     valid_node = node

```

```

109
110
111     svg += '</svg>'
112     with open(file_name, 'w') as fp:
113         fp.write(svg)
114
115
116
117 class MatrixOptimizer:
118     def __init__(self, cTable: TableHandler, tTable: TableHandler, max_T: float):
119         self.max_T = max_T
120         self.cTable: TableHandler = cTable
121         self.tTable: TableHandler = tTable
122         self.cMatrix: np.array = cTable.matrix
123         self.tMatrix: np.array = tTable.matrix
124         self.rows, self.columns = self.cMatrix.shape
125         self.eliminated = np.zeros((self.rows, self.columns), dtype=bool)
126         self.treeDone = False
127
128     def refreshValues(self):
129         self.cMatrix = self.cTable.matrix
130         self.tMatrix = self.tTable.matrix
131         self.rows, self.columns = self.cMatrix.shape
132         self.eliminated = np.zeros((self.rows, self.columns), dtype=bool)
133
134     def optimization1(self):
135         c_argmin = self.cMatrix.argmax(1)
136         eliminated = []
137         for i in range(self.rows):
138             t_min = self.tMatrix[i, c_argmin[i]]
139             c_min = self.cMatrix[i, c_argmin[i]]
140             eliminated.append([ self.tMatrix[i, j] > t_min or (self.tMatrix[i, j] ==
141                 ↪ t_min and self.cMatrix[i, j] > c_min) for j in range(self.columns)
142                 ↪ ])
143
144         self.eliminated = np.logical_or(self.eliminated, np.array(eliminated,
145             ↪ dtype=bool))
146         return self.eliminated
147
148     def optimization2(self):
149         t_min = self.tMatrix.min(1)
150         t_min_sum = sum(t_min)
151         eliminated = []
152         for i in range(self.rows):
153             eliminated.append([ t_min_sum - t_min[i] + self.tMatrix[i, j] >
154                 ↪ self.max_T for j in range(self.columns) ])
155
156         self.eliminated = np.logical_or(self.eliminated, np.array(eliminated,
157             ↪ dtype=bool))
158         return self.eliminated
159
160     def tree_add_current_row_nodes(self, fixed_nodes, row, c_argmin, t_argmin):
161         current_row_good_nodes = []
162         for j in range(self.columns):

```

```

158         if self.eliminated[row,j]:
159             continue
160
161         fixed_nodes[row] = j
162         c_sum = sum(self.cMatrix[range(row+1),fixed_nodes])
163         c_sum += sum(self.cMatrix[range(row+1, self.rows),c_argmin[row+1:]])
164         t_sum = sum(self.tMatrix[range(row+1),fixed_nodes])
165         t_sum += sum(self.tMatrix[range(row+1, self.rows),t_argmin[row+1:]])
166         node = self.root.add_child(fixed_nodes, c_sum, t_sum)
167         if t_sum > self.max_T:
168             node.status_good = False
169         else:
170             current_row_good_nodes.append(node)
171     return current_row_good_nodes
172
173
174 def tree_optimization(self):
175     c_argmin = self.cMatrix.argmin(1)
176     t_argmin = self.tMatrix.argmin(1)
177
178     fixed_nodes = []
179     c_sum = sum(self.cMatrix[range(self.rows),c_argmin])
180     t_sum = sum(self.tMatrix[range(self.rows),t_argmin])
181     self.root = Tree()
182     self.root.set_metrics(c_sum, t_sum)
183     self.root.set_status(status_good=True)
184
185     for i in range(self.rows):
186         fixed_nodes.append(0)
187         current_row_good_nodes = self.tree_add_current_row_nodes(fixed_nodes,
188             ↪ i, c_argmin, t_argmin)
189
190         min_c = current_row_good_nodes[0].c_sum
191         ind = current_row_good_nodes[0].fixed_nodes[-1]
192         for i, node in enumerate(current_row_good_nodes):
193             if i == 0:
194                 continue
195
196             if node.c_sum < min_c:
197                 min_c = node.c_sum
198                 ind = node.fixed_nodes[-1]
199
200         fixed_nodes[-1] = ind
201         for node in current_row_good_nodes:
202             if node.fixed_nodes[-1] != ind:
203                 node.set_status(False)
204
205     self.root.print_tree()
206     self.treeDone = True
207     self.root.toSvg()
208     msg = QMessageBox()
209     msg.setText('Дерево успешно построено')

```

210 msg.exec()

## Листинг 5. main\_window.ui

```

1  <?xml version="1.0" encoding="UTF-8"?>
2  <ui version="4.0">
3      <class>MainWindow</class>
4      <widget class="QMainWindow" name="MainWindow">
5          <property name="geometry">
6              <rect>
7                  <x>0</x>
8                  <y>0</y>
9                  <width>800</width>
10                 <height>700</height>
11             </rect>
12         </property>
13         <property name="sizePolicy">
14             <sizepolicy hsize="Preferred" vsizetype="Preferred">
15                 <horstretch>0</horstretch>
16                 <verstretch>0</verstretch>
17             </sizepolicy>
18         </property>
19         <property name="windowTitle">
20             <string>Распределение задач по узлам управления</string>
21         </property>
22         <widget class="QWidget" name="centralwidget">
23             <layout class="QGridLayout" name="gridLayout">
24                 <item row="1" column="0">
25                     <widget class="QFrame" name="frame">
26                         <property name="frameShape">
27                             <enum>QFrame::NoFrame</enum>
28                         </property>
29                         <property name="frameShadow">
30                             <enum>QFrame::Plain</enum>
31                         </property>
32                         <layout class="QHBoxLayout" name="horizontalLayout">
33                             <item>
34                                 <spacer name="horizontalSpacer">
35                                     <property name="orientation">
36                                         <enum>Qt::Horizontal</enum>
37                                     </property>
38                                     <property name="sizeHint" stdset="0">
39                                         <size>
40                                             <width>40</width>
41                                             <height>20</height>
42                                         </size>
43                                     </property>
44                                 </spacer>
45                             </item>
46                             <item>
47                                 <layout class="QVBoxLayout" name="verticalLayout_2">
48                                     <item>

```



```

49 <widget class="QLabel" name="cLabel">
50 <property name="sizePolicy">
51 <sizepolicy hsize="Preferred" vsize="Minimum">
52 <horstretch>0</horstretch>
53 <verstretch>0</verstretch>
54 </sizepolicy>
55 </property>
56 <property name="text">
57 <string>Матрица C</string>
58 </property>
59 <property name="alignment">
60 <set>Qt::AlignCenter</set>
61 </property>
62 </widget>
63 </item>
64 <item>
65 <widget class="QTableWidget" name="cMatrix">
66 <property name="sizePolicy">
67 <sizepolicy hsize="Minimum" vsize="Minimum">
68 <horstretch>0</horstretch>
69 <verstretch>0</verstretch>
70 </sizepolicy>
71 </property>
72 <property name="minimumSize">
73 <size>
74 <width>0</width>
75 <height>0</height>
76 </size>
77 </property>
78 <property name="verticalScrollBarPolicy">
79 <enum>Qt::ScrollBarAlwaysOff</enum>
80 </property>
81 <property name="horizontalScrollBarPolicy">
82 <enum>Qt::ScrollBarAlwaysOff</enum>
83 </property>
84 <property name="sizeAdjustPolicy">
85 <enum>QAbstractScrollArea::AdjustToContents</enum>
86 </property>
87 <property name="selectionMode">
88 <enum>QAbstractItemView::SingleSelection</enum>
89 </property>
90 <property name="showGrid">
91 <bool>true</bool>
92 </property>
93 <property name="wordWrap">
94 <bool>false</bool>
95 </property>
96 <property name="cornerButtonEnabled">
97 <bool>true</bool>
98 </property>
99 <property name="rowCount">
100 <number>5</number>
101 </property>
102 <property name="columnCount">

```

```

103         <number>4</number>
104     </property>
105     <attribute name="horizontalHeaderVisible">
106         <bool>true</bool>
107     </attribute>
108     <attribute name="horizontalHeaderCascadingSectionResizes">
109         <bool>>false</bool>
110     </attribute>
111     <attribute name="horizontalHeaderDefaultSectionSize">
112         <number>60</number>
113     </attribute>
114     <attribute name="horizontalHeaderMinimumSectionSize">
115         <number>60</number>
116     </attribute>
117     <attribute name="horizontalHeaderStretchLastSection">
118         <bool>>false</bool>
119     </attribute>
120     <attribute name="verticalHeaderVisible">
121         <bool>true</bool>
122     </attribute>
123     <row/>
124     <row/>
125     <row/>
126     <row/>
127     <row/>
128     <column/>
129     <column/>
130     <column/>
131     <column/>
132 </widget>
133 </item>
134 </layout>
135 </item>
136 <item>
137     <layout class="QVBoxLayout" name="verticalLayout_3">
138         <item>
139             <widget class="QLabel" name="tLabel">
140                 <property name="sizePolicy">
141                     <sizepolicy hstretch="Preferred" vsizetype="Minimum">
142                         <hstretch>0</hstretch>
143                         <verstretch>0</verstretch>
144                     </sizepolicy>
145                 </property>
146                 <property name="text">
147                     <string>Матрица T</string>
148                 </property>
149                 <property name="alignment">
150                     <set>Qt::AlignCenter</set>
151                 </property>
152             </widget>
153         </item>
154         <item>
155             <widget class="QTableWidget" name="tMatrix">
156                 <property name="sizePolicy">

```

```
157     <sizepolicy hsizetype="Minimum" vsizetype="Minimum">
158         <horstretch>0</horstretch>
159         <verstretch>0</verstretch>
160     </sizepolicy>
161 </property>
162 <property name="minimumSize">
163     <size>
164         <width>0</width>
165         <height>0</height>
166     </size>
167 </property>
168 <property name="autoFillBackground">
169     <bool>>false</bool>
170 </property>
171 <property name="lineWidth">
172     <number>1</number>
173 </property>
174 <property name="verticalScrollBarPolicy">
175     <enum>Qt::ScrollBarAlwaysOff</enum>
176 </property>
177 <property name="horizontalScrollBarPolicy">
178     <enum>Qt::ScrollBarAlwaysOff</enum>
179 </property>
180 <property name="sizeAdjustPolicy">
181     <enum>QAbstractScrollArea::AdjustToContents</enum>
182 </property>
183 <property name="autoScrollMargin">
184     <number>16</number>
185 </property>
186 <property name="selectionMode">
187     <enum>QAbstractItemView::SingleSelection</enum>
188 </property>
189 <property name="textElideMode">
190     <enum>Qt::ElideNone</enum>
191 </property>
192 <property name="showGrid">
193     <bool>true</bool>
194 </property>
195 <property name="wordWrap">
196     <bool>>false</bool>
197 </property>
198 <property name="cornerButtonEnabled">
199     <bool>true</bool>
200 </property>
201 <property name="rowCount">
202     <number>5</number>
203 </property>
204 <property name="columnCount">
205     <number>4</number>
206 </property>
207 <attribute name="horizontalHeaderVisible">
208     <bool>true</bool>
209 </attribute>
210 <attribute name="horizontalHeaderCascadingSectionResizes">
```

```

211     <bool>false</bool>
212 </attribute>
213 <attribute name="horizontalHeaderDefaultSectionSize">
214     <number>60</number>
215 </attribute>
216 <attribute name="horizontalHeaderMinimumSectionSize">
217     <number>60</number>
218 </attribute>
219 <attribute name="horizontalHeaderStretchLastSection">
220     <bool>false</bool>
221 </attribute>
222 <attribute name="verticalHeaderVisible">
223     <bool>true</bool>
224 </attribute>
225 <row/>
226 <row/>
227 <row/>
228 <row/>
229 <row/>
230 <column/>
231 <column/>
232 <column/>
233 <column/>
234 </widget>
235 </item>
236 </layout>
237 </item>
238 <item>
239     <spacer name="horizontalSpacer_2">
240         <property name="orientation">
241             <enum>Qt::Horizontal</enum>
242         </property>
243         <property name="sizeHint" stdset="0">
244             <size>
245                 <width>40</width>
246                 <height>20</height>
247             </size>
248         </property>
249     </spacer>
250 </item>
251 </layout>
252 </widget>
253 </item>
254 <item row="2" column="0">
255     <layout class="QHBoxLayout" name="horizontalLayout_3">
256         <property name="spacing">
257             <number>10</number>
258         </property>
259         <property name="leftMargin">
260             <number>0</number>
261         </property>
262         <property name="rightMargin">
263             <number>0</number>
264         </property>

```

```

265 <item>
266   <spacer name="horizontalSpacer_5">
267     <property name="orientation">
268       <enum>Qt::Horizontal</enum>
269     </property>
270     <property name="sizeHint" stdset="0">
271       <size>
272         <width>40</width>
273         <height>20</height>
274       </size>
275     </property>
276   </spacer>
277 </item>
278 <item>
279   <widget class="QLabel" name="T_zLabel">
280     <property name="text">
281       <string>Значение Тз</string>
282     </property>
283   </widget>
284 </item>
285 <item>
286   <widget class="QLineEdit" name="T_max_lEdit">
287     <property name="maximumSize">
288       <size>
289         <width>100</width>
290         <height>16777215</height>
291       </size>
292     </property>
293   </widget>
294 </item>
295 <item>
296   <spacer name="horizontalSpacer_6">
297     <property name="orientation">
298       <enum>Qt::Horizontal</enum>
299     </property>
300     <property name="sizeHint" stdset="0">
301       <size>
302         <width>40</width>
303         <height>20</height>
304       </size>
305     </property>
306   </spacer>
307 </item>
308 </layout>
309 </item>
310 <item row="5" column="0">
311   <layout class="QHBoxLayout" name="horizontalLayout_2">
312     <item>
313       <widget class="QPushButton" name="optimize1">
314         <property name="enabled">
315           <bool>false</bool>
316         </property>
317         <property name="minimumSize">
318           <size>

```

```

319         <width>0</width>
320         <height>60</height>
321     </size>
322 </property>
323 <property name="text">
324     <string>1 ОПТИМИЗАЦИЯ</string>
325 </property>
326 </widget>
327 </item>
328 <item>
329     <widget class="QPushButton" name="optimize2">
330         <property name="enabled">
331             <bool>>false</bool>
332         </property>
333         <property name="minimumSize">
334             <size>
335                 <width>0</width>
336                 <height>60</height>
337             </size>
338         </property>
339         <property name="text">
340             <string>2 ОПТИМИЗАЦИЯ</string>
341         </property>
342     </widget>
343 </item>
344 <item>
345     <widget class="QPushButton" name="decisionTreeBtn">
346         <property name="enabled">
347             <bool>>false</bool>
348         </property>
349         <property name="sizePolicy">
350             <sizepolicy hstretch="Minimum" vsizetype="Fixed">
351                 <horstretch>0</horstretch>
352                 <verstretch>0</verstretch>
353             </sizepolicy>
354         </property>
355         <property name="minimumSize">
356             <size>
357                 <width>0</width>
358                 <height>60</height>
359             </size>
360         </property>
361         <property name="text">
362             <string>Построить
363     дерево решений</string>
364         </property>
365     </widget>
366 </item>
367 </layout>
368 </item>
369 <item row="3" column="0">
370     <layout class="QHBoxLayout" name="horizontalLayout_5">
371         <item>
372             <spacer name="horizontalSpacer_7">

```

```

373     <property name="orientation">
374         <enum>Qt::Horizontal</enum>
375     </property>
376     <property name="sizeHint" stdset="0">
377         <size>
378             <width>40</width>
379             <height>20</height>
380         </size>
381     </property>
382 </spacer>
383 </item>
384 <item>
385     <widget class="QPushButton" name="dataInput">
386         <property name="minimumSize">
387             <size>
388                 <width>30</width>
389                 <height>50</height>
390             </size>
391         </property>
392         <property name="maximumSize">
393             <size>
394                 <width>999999</width>
395                 <height>16777215</height>
396             </size>
397         </property>
398         <property name="layoutDirection">
399             <enum>Qt::LeftToRight</enum>
400         </property>
401         <property name="text">
402             <string>Ввод данных</string>
403         </property>
404     </widget>
405 </item>
406 <item>
407     <spacer name="horizontalSpacer_8">
408         <property name="orientation">
409             <enum>Qt::Horizontal</enum>
410         </property>
411         <property name="sizeHint" stdset="0">
412             <size>
413                 <width>40</width>
414                 <height>20</height>
415             </size>
416         </property>
417     </spacer>
418 </item>
419 </layout>
420 </item>
421 <item row="0" column="0">
422     <layout class="QHBoxLayout" name="horizontalLayout_4">
423         <item>
424             <spacer name="horizontalSpacer_4">
425                 <property name="orientation">
426                     <enum>Qt::Horizontal</enum>

```

```

427     </property>
428     <property name="sizeHint" stdset="0">
429         <size>
430             <width>40</width>
431             <height>20</height>
432         </size>
433     </property>
434 </spacer>
435 </item>
436 <item>
437     <widget class="QLabel" name="instructionLabel">
438         <property name="font">
439             <font>
440                 <pointsize>12</pointsize>
441             </font>
442         </property>
443         <property name="text">
444             <string>Введите данные</string>
445         </property>
446         <property name="scaledContents">
447             <bool>false</bool>
448         </property>
449         <property name="alignment">
450             <set>Qt::AlignCenter</set>
451         </property>
452     </widget>
453 </item>
454 <item>
455     <spacer name="horizontalSpacer_3">
456         <property name="orientation">
457             <enum>Qt::Horizontal</enum>
458         </property>
459         <property name="sizeHint" stdset="0">
460             <size>
461                 <width>40</width>
462                 <height>20</height>
463             </size>
464         </property>
465     </spacer>
466 </item>
467 </layout>
468 </item>
469 <item row="4" column="0">
470     <spacer name="verticalSpacer">
471         <property name="orientation">
472             <enum>Qt::Vertical</enum>
473         </property>
474         <property name="sizeType">
475             <enum>QSizePolicy::Fixed</enum>
476         </property>
477         <property name="sizeHint" stdset="0">
478             <size>
479                 <width>20</width>
480                 <height>15</height>

```



```
481     </size>
482   </property>
483 </spacer>
484 </item>
485 <item row="6" column="0">
486   <widget class="QWidget" name="svgView" native="true">
487     <property name="sizePolicy">
488       <sizepolicy hstypе="Preferred" vstypе="Expanding">
489         <horstretch>0</horstretch>
490         <verstretch>0</verstretch>
491       </sizepolicy>
492     </property>
493   </widget>
494 </item>
495 </layout>
496 </widget>
497 <widget class="QMenuBar" name="menubar">
498   <property name="geometry">
499     <rect>
500       <x>0</x>
501       <y>0</y>
502       <width>800</width>
503       <height>21</height>
504     </rect>
505   </property>
506 </widget>
507 <widget class="QStatusBar" name="statusbar"/>
508 </widget>
509 <resources/>
510 <connections/>
511 </ui>
```

---