

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

ФАКУЛЬТЕТ ІНФОРМАТИКИ ТА ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ

КАФЕДРА ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ

ЛАБОРАТОРНА РОБОТА №5

З КУРСУ

«ТЕХНОЛОГІЯ ПРОЕКТУВАННЯ КОМП’ЮТЕРНИХ СИСТЕМ»

Виконав:

студент IV курсу

групи ІО-21

Журо Георгій Олександрович

Київ – 2015

**Тема:** Автоматизація синтезу таблиці переходів.

**Мета:** Здобуття навичок з аналізу графових структур і автоматизації процедури побудови таблиці переходів.

**Завдання:**

1. **Тип тригера:** D.
2. Розробити модуль генерації таблиці переходів і функцій збудження тригерів на основі закодованого графу переходів.
3. Реалізувати засоби відображення таблиці та її збереження у файлі.

**Опис програми:**

Для побудови таблиці переходів та функцій збудження тригерів мною був розроблений клас AutomatTableModel. Він приймає в конструкторі об’єкт класу CodedMooreAutomat, що містить всі дані про граф. З цих даних і будується таблиця. Для її відображення мною був використаний клас JTable.

Для побудови таблиці переходів необхідно після кодування графу (л.р. №4) натиснути кнопку «Build Table Of Transitions», після чого відкриється нова вкладка «Table Of Transitions», в якій і буде відображена побудована таблиця. Наприклад, для даного графу (рисунок 1) таблиця переходів зображена на рисунку 2.

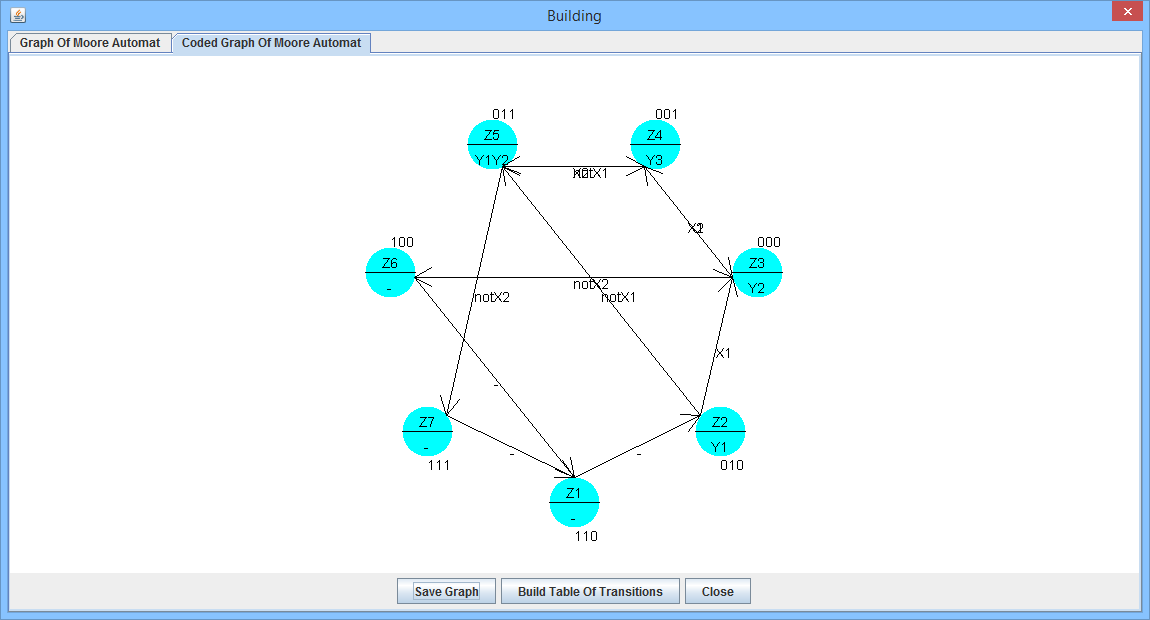


Рисунок 1

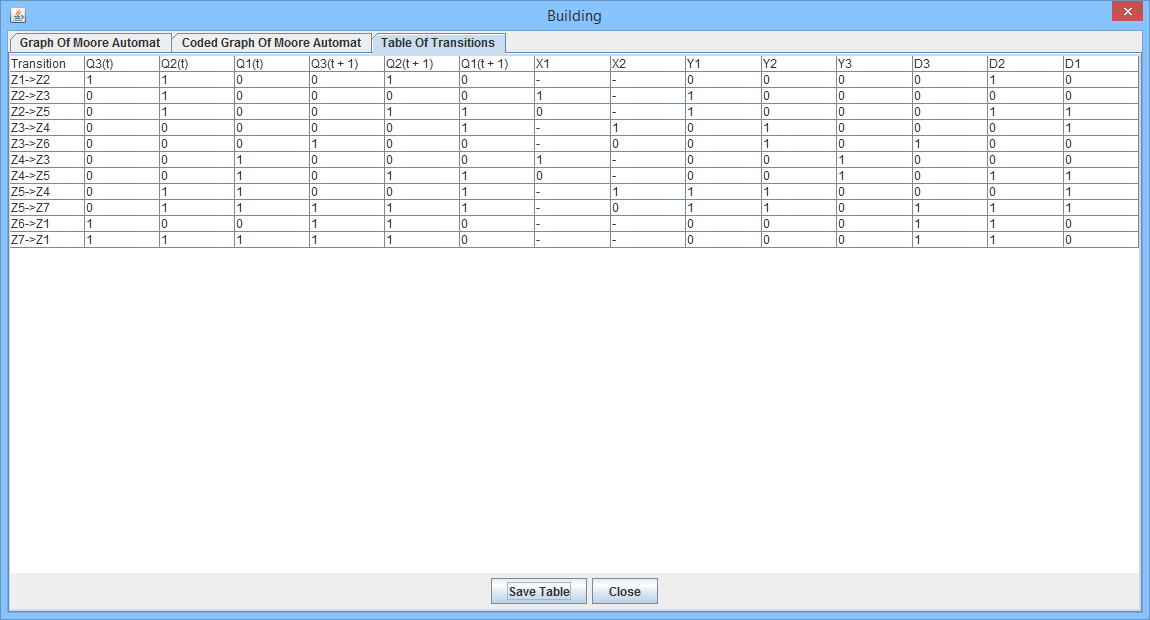


Рисунок 2

Для збереження таблиці у текстовому файлі необхідно натиснути на кнопку «Save Table» і вибрати необхідний файл у діалоговому вікні. Вміст файлу для таблиці з рисунку 2 показаний на рисунку 3.

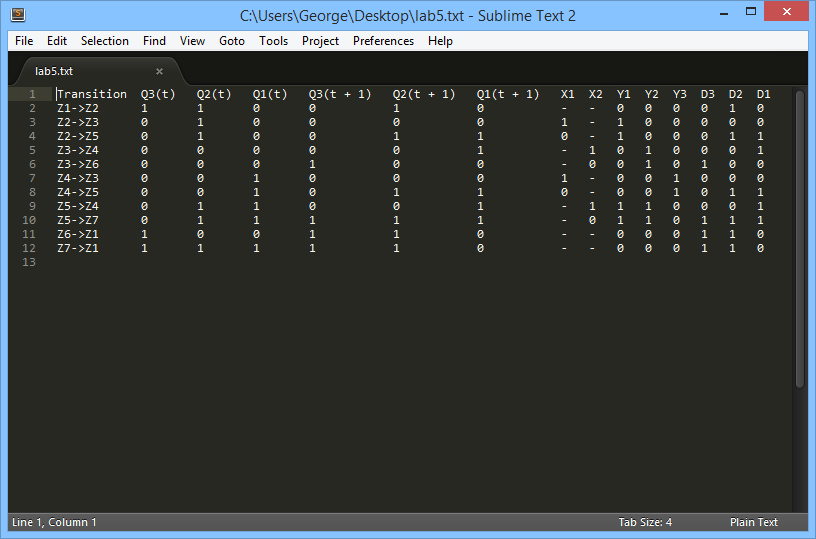


Рисунок 3

**Лістинг програми:**

package face;

import automat.moore.\*;

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.io.File;

import java.io.IOException;

class BuildFrame extends JDialog {

private MainFrame mainFrame;

private JTabbedPane tabbedPane;

private GraphPanel graphPanel;

private CodedGraphPanel codedGraphPanel;

private JButton codeGraphButton;

private AutomatTableModel tableModel;

private JButton buildTableButton;

public BuildFrame(MainFrame frame, Rectangle bounds, MooreAutomat automat) {

super(frame);

mainFrame = frame;

setBounds(bounds);

setMinimumSize(bounds.getSize());

setResizable(true);

setModal(true);

setTitle("Building");

tabbedPane = new JTabbedPane();

add(tabbedPane);

JPanel mooreGraphPanel = new JPanel();

mooreGraphPanel.setLayout(new BorderLayout());

graphPanel = new GraphPanel(new GraphModel(automat));

JPanel mooreGraphButtonsPanel = new JPanel();

JButton saveGraphButton = new JButton(new SaveGraphAction(this));

saveGraphButton.setText("Save Graph");

codeGraphButton = new JButton(new CodeGraphAction(this));

codeGraphButton.setText("Code Graph");

JButton closeButton = new JButton(new AbstractAction() {

public void actionPerformed(ActionEvent e) {

setVisible(false);

}

});

closeButton.setText("Close");

mooreGraphButtonsPanel.add(saveGraphButton);

mooreGraphButtonsPanel.add(codeGraphButton);

mooreGraphButtonsPanel.add(closeButton);

mooreGraphPanel.add(mooreGraphButtonsPanel, BorderLayout.SOUTH);

mooreGraphPanel.add(graphPanel);

tabbedPane.addTab("Graph Of Moore Automat", mooreGraphPanel);

}

public BuildFrame(MainFrame frame, Rectangle bounds, CodedMooreAutomat automat) {

super(frame);

mainFrame = frame;

setBounds(bounds);

setMinimumSize(bounds.getSize());

setResizable(true);

setModal(true);

setTitle("Building");

tabbedPane = new JTabbedPane();

add(tabbedPane);

JPanel mooreCodedGraphPanel = new JPanel();

mooreCodedGraphPanel.setLayout(new BorderLayout());

codedGraphPanel = new CodedGraphPanel(new GraphModel(automat));

JPanel mooreGraphButtonsPanel = new JPanel();

JButton saveCodedGraphButton = new JButton(new SaveCodedGraphAction(this));

saveCodedGraphButton.setText("Save Graph");

buildTableButton = new JButton(new BuildTableAction(this));

buildTableButton.setText("Build Table Of Transitions");

JButton closeButton = new JButton(new AbstractAction() {

public void actionPerformed(ActionEvent e) {

setVisible(false);

}

});

closeButton.setText("Close");

mooreGraphButtonsPanel.add(saveCodedGraphButton);

mooreGraphButtonsPanel.add(buildTableButton);

mooreGraphButtonsPanel.add(closeButton);

mooreCodedGraphPanel.add(mooreGraphButtonsPanel, BorderLayout.SOUTH);

mooreCodedGraphPanel.add(codedGraphPanel);

tabbedPane.addTab("Coded Graph Of Moore Automat", mooreCodedGraphPanel);

}

private class SaveGraphAction extends AbstractAction {

private BuildFrame frame;

public SaveGraphAction(BuildFrame frame) {

this.frame = frame;

}

public void actionPerformed(ActionEvent e) {

JFileChooser chooser = mainFrame.getChooser();

chooser.resetChoosableFileFilters();

chooser.addChoosableFileFilter(new GraphFileFilter());

int result = chooser.showSaveDialog(frame);

if (result == JFileChooser.APPROVE\_OPTION) {

if (!chooser.getSelectedFile().getName().endsWith(GraphFileFilter.GRAPH\_EXTENSION)) {

chooser.setSelectedFile(new File(chooser.getSelectedFile().getAbsolutePath() + GraphFileFilter.GRAPH\_EXTENSION));

}

try {

MooreAutomat.writeToFile(chooser.getSelectedFile(), graphPanel.getModel().getAutomat());

} catch (IOException e1) {

JOptionPane.showMessageDialog(frame, "Error! Can't create file.",

"Error", JOptionPane.ERROR\_MESSAGE);

}

}

}

}

private class SaveCodedGraphAction extends AbstractAction {

private BuildFrame frame;

public SaveCodedGraphAction(BuildFrame frame) {

this.frame = frame;

}

public void actionPerformed(ActionEvent e) {

JFileChooser chooser = mainFrame.getChooser();

chooser.resetChoosableFileFilters();

chooser.addChoosableFileFilter(new CodedGraphFileFilter());

int result = chooser.showSaveDialog(frame);

if (result == JFileChooser.APPROVE\_OPTION) {

if (!chooser.getSelectedFile().getName().endsWith(CodedGraphFileFilter.CODED\_GRAPH\_EXTENSION)) {

chooser.setSelectedFile(new File(chooser.getSelectedFile().getAbsolutePath() + CodedGraphFileFilter.CODED\_GRAPH\_EXTENSION));

}

try {

CodedMooreAutomat.writeToFile(chooser.getSelectedFile(), (CodedMooreAutomat) codedGraphPanel.getModel().getAutomat());

} catch (IOException e1) {

JOptionPane.showMessageDialog(frame, "Error! Can't create file.",

"Error", JOptionPane.ERROR\_MESSAGE);

}

}

}

}

private class CodeGraphAction extends AbstractAction {

private BuildFrame frame;

public CodeGraphAction(BuildFrame frame) {

this.frame = frame;

}

public void actionPerformed(ActionEvent e) {

JPanel mooreCodedGraphPanel = new JPanel();

mooreCodedGraphPanel.setLayout(new BorderLayout());

codedGraphPanel = new CodedGraphPanel(new GraphModel(graphPanel.getModel().getAutomat()));

JPanel mooreGraphButtonsPanel = new JPanel();

JButton saveCodedGraphButton = new JButton(new SaveCodedGraphAction(frame));

saveCodedGraphButton.setText("Save Graph");

buildTableButton = new JButton(new BuildTableAction(frame));

buildTableButton.setText("Build Table Of Transitions");

JButton closeButton = new JButton(new AbstractAction() {

public void actionPerformed(ActionEvent e) {

setVisible(false);

}

});

closeButton.setText("Close");

mooreGraphButtonsPanel.add(saveCodedGraphButton);

mooreGraphButtonsPanel.add(buildTableButton);

mooreGraphButtonsPanel.add(closeButton);

mooreCodedGraphPanel.add(mooreGraphButtonsPanel, BorderLayout.SOUTH);

mooreCodedGraphPanel.add(codedGraphPanel);

tabbedPane.addTab("Coded Graph Of Moore Automat", mooreCodedGraphPanel);

tabbedPane.setSelectedIndex(1);

codeGraphButton.setEnabled(false);

}

}

private class SaveTableAction extends AbstractAction {

private BuildFrame frame;

public SaveTableAction(BuildFrame frame) {

this.frame = frame;

}

public void actionPerformed(ActionEvent e) {

JFileChooser chooser = mainFrame.getChooser();

chooser.resetChoosableFileFilters();

chooser.addChoosableFileFilter(new TextFileFilter());

int result = chooser.showSaveDialog(frame);

if (result == JFileChooser.APPROVE\_OPTION) {

if (!chooser.getSelectedFile().getName().endsWith(TextFileFilter.TEXT\_FILE\_EXTENSION)) {

chooser.setSelectedFile(new File(chooser.getSelectedFile().getAbsolutePath() + TextFileFilter.TEXT\_FILE\_EXTENSION));

}

try {

tableModel.writeToFile(chooser.getSelectedFile());

} catch (IOException e1) {

JOptionPane.showMessageDialog(frame, "Error! Can't create file.",

"Error", JOptionPane.ERROR\_MESSAGE);

}

}

}

}

private class BuildTableAction extends AbstractAction {

private BuildFrame frame;

public BuildTableAction(BuildFrame frame) {

this.frame = frame;

}

public void actionPerformed(ActionEvent e) {

JPanel tablePanel = new JPanel();

tablePanel.setLayout(new BorderLayout());

tableModel = new AutomatTableModel((CodedMooreAutomat) codedGraphPanel.getModel().getAutomat());

JTable table = new JTable(tableModel);

JPanel tableButtonsPanel = new JPanel();

JButton saveTableButton = new JButton(new SaveTableAction(frame));

saveTableButton.setText("Save Table");

JButton closeButton = new JButton(new AbstractAction() {

public void actionPerformed(ActionEvent e) {

setVisible(false);

}

});

closeButton.setText("Close");

tableButtonsPanel.add(saveTableButton);

tableButtonsPanel.add(closeButton);

tablePanel.add(tableButtonsPanel, BorderLayout.SOUTH);

tablePanel.add(table);

tabbedPane.addTab("Table Of Transitions", tablePanel);

tabbedPane.setSelectedIndex(tabbedPane.getTabCount() - 1);

buildTableButton.setEnabled(false);

}

}

}

**Висновки:** При виконанні даної лабораторної роботи я здобув навички з аналізу графових структур і автоматизації процедури побудови таблиці переходів. Мною був розроблений модуль для генерації таблиці переходів. А також я реалізував засоби для візуального відображення сгенерованої таблиці переходів автомату. Для цього мною був використаний клас JTable мови програмування Java.