**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**Чорноморський національний університет   
імені Петра Могили**

**Факультет комп’ютерних наук**

**Кафедра "Інтелектуальних інформаційних систем"**

**ЗВІТ**

*з практичної роботи № 1*

***Варіант № 22***

"Реалізація методу агрегації трапеційних лінгвістичних термів на основі експертних знань для задач багатокритерійного прийняття рішень"

Дисципліна "Знання-орієнтовані технології обчислювального інтелекту"

Спеціальність: 122 Комп’ютерні науки

122-ЛР.1-501.22450122

***Cтудентка*** *А. Ю. Цимбал*

*21.09.2024*

***Викладач*** Є. В. Сіденко

**Миколаїв – 2024**

**Практична робота №1**

**Тема:** Реалізація методу агрегації трапеційних лінгвістичних термів на основі експертних знань для задач багатокритерійного прийняття рішень.

**Мета:** Розробка методу агрегації трапеційних ЛТ для підвищення ефективності процесу вибору найкращого рішення серед існуючих альтернативних варіантів за визначеним переліком критеріїв при коливних лінгвістичних оцінках експертів.

**Теоретичні відомості**

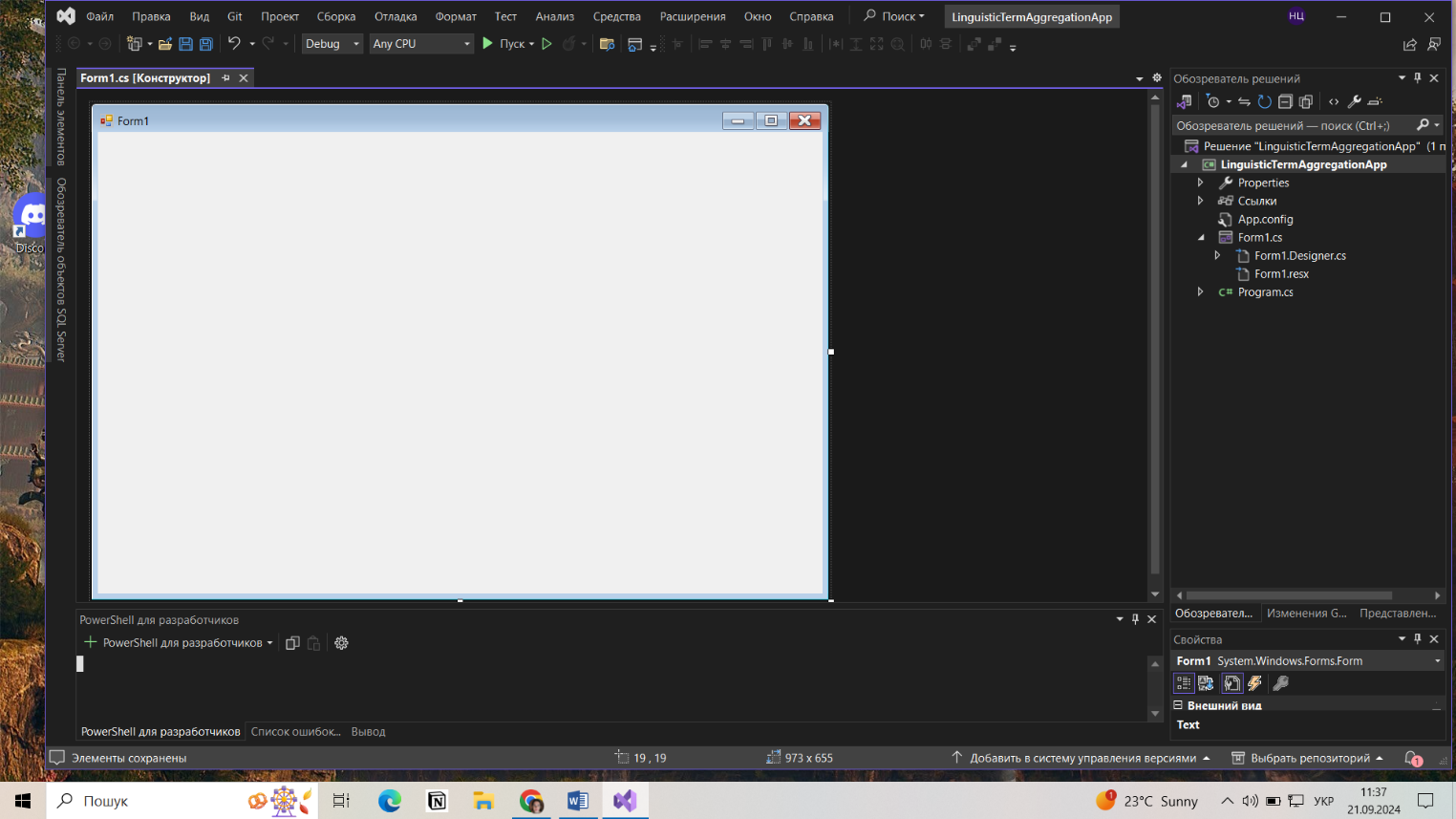
Процес багатокритерійного прийняття рішень полягає у виборі найкращого рішення серед альтернативних варіантів за визначеним переліком критеріїв. Моделювання на основі експертних знань про систему представляє собою підхід, в основу якого покладено знання та досвід людини, яка виступає експертом в питаннях, що пов’язані з реальною системою. Пошук рішення багатокритеріальної задачі не представляє особливих труднощів, якщо перевага по одному з критеріїв призводить до такої ж переваги за іншим критерієм, тобто якщо критерії кооперуються. Складність виникає в тому випадку, коли оцінки рішення за відповідними критеріями не порівняльні між собою. При цьому виникають ситуації, в яких експерту складно оцінити альтернативне рішення, використовуючи лише кількісну шкалу оцінювання. Це пов’язано з тим, що судження експерта в більшості випадків приймають мовну форму у вигляді нечітких множин, правил та граматик. Так, наприклад, експерту простіше оцінити транспортну компанію за критерієм «рівень страхування вантажу» в нечіткій формі (за допомогою лінгвістичних оцінок – термів), наприклад «рівень страхування вантажу – низький або нижче середнього», ніж дати точну кількісну оцінку. Розв’язання задач багатокритерійного прийняття рішень в умовах невизначеності, при яких оцінки експерта коливаються в межах декількох лінгвістичних значень оцінювального параметру, є актуальною проблемою на сьогоднішній день. Це пов’язано зі складністю розробки моделей, які враховують відповідні умови невизначеності, оскільки експертні оцінки в більшості випадків представляються у вигляді нечітких інтервальних ЛТ. При цьому експерт зазвичай вказує оцінку в межах декількох ЛТ для кожного альтернативного рішення за відповідним критерієм.

**Завдання:**

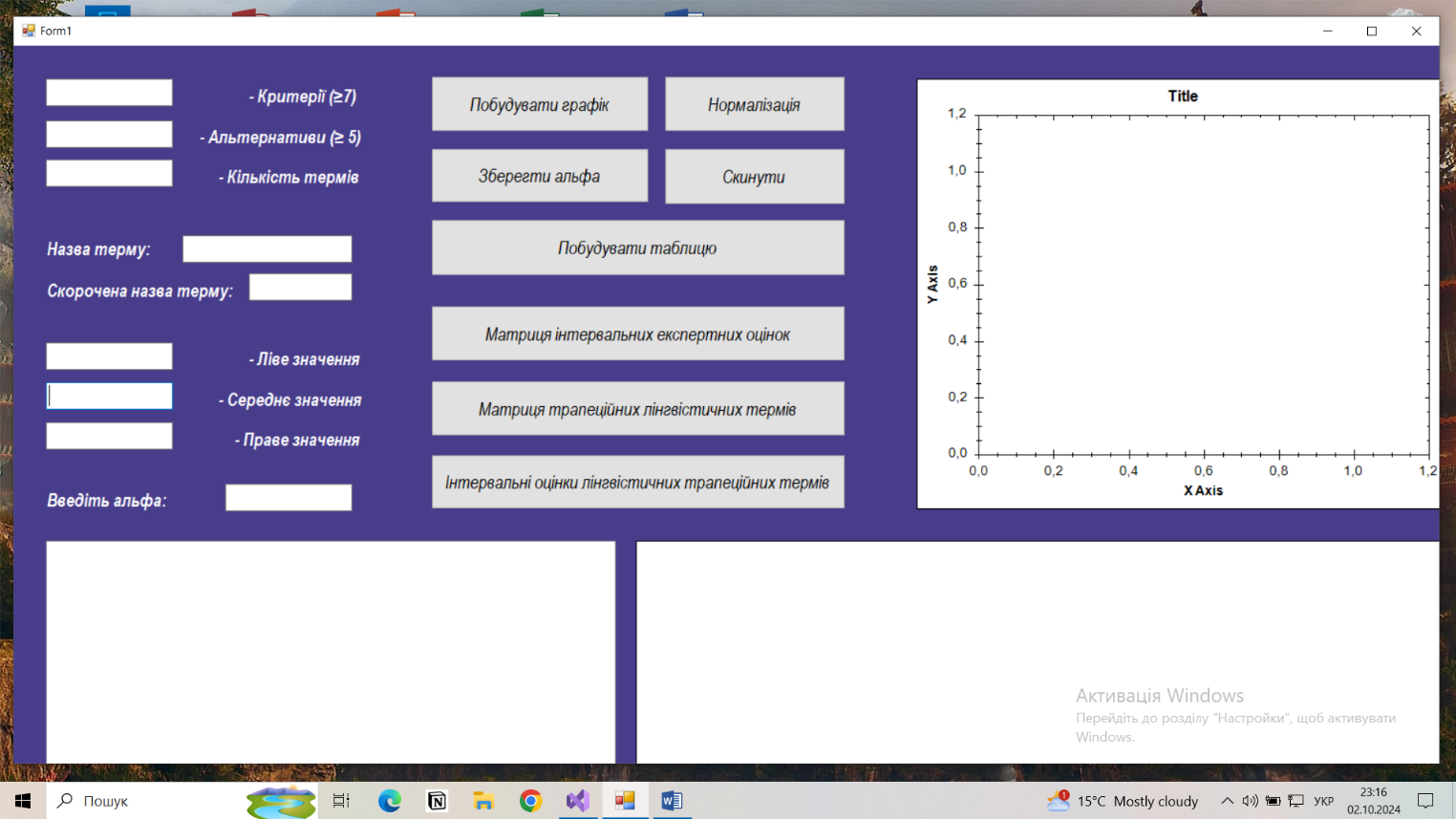
Необхідно розробити програмний застосунок для вирішення обраної студентом задачі (кількість альтернатив не менше 5, кількість критеріїв не менше 7) з використанням методу агрегації трапеційних лінгвістичних термів (для песимістичної, оптимістичної та нейтральної (узагальненої) позицій ЛПР).

**Розв’язання:**

Створимо проект на C# з WinForms у Visual Studio. Проект LinguisticTermAggregationApp:



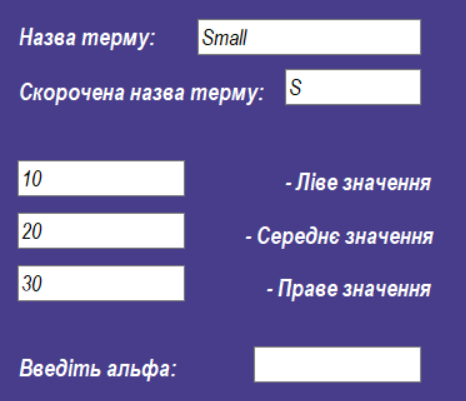
Додаємо усі необхідні компоненти з панелі елементів та стилізуємо:



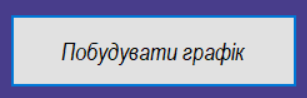
Спочатку вводимо необхідну кількість критеріїв, альтернатив та термів:



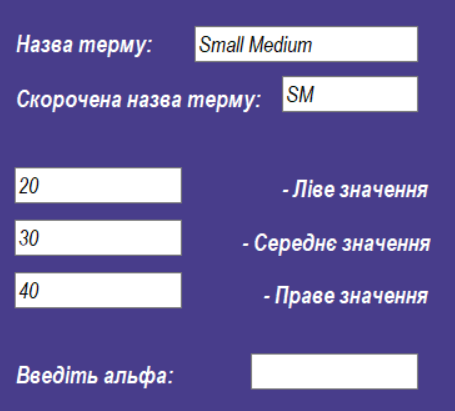
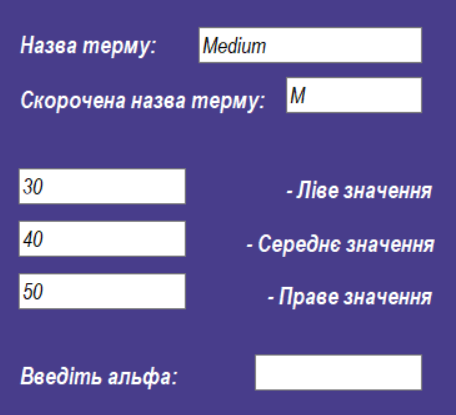
Потім для першого трикутного числа вводимо назву терму, його скорочену назву та параметри лівої, правої і середньої границь:

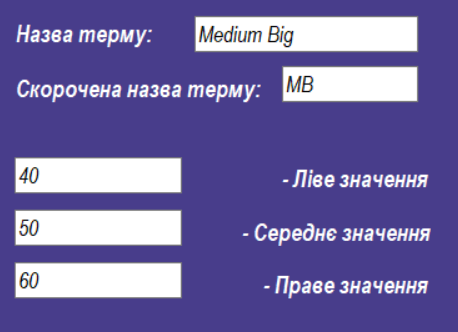
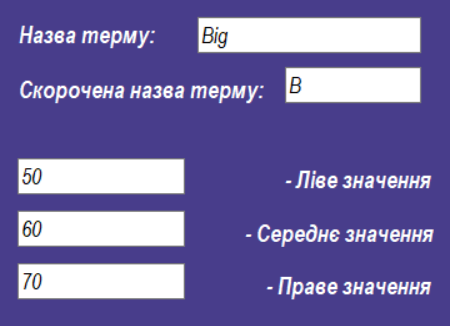


Натискаємо на кнопку побудови графіку і на ньому будується перше трикутне число.

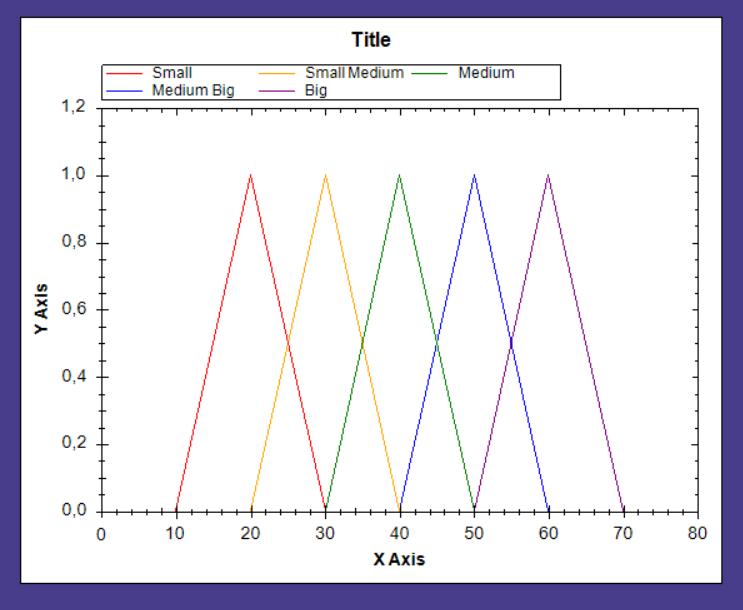


Аналогічно, вводимо параметри для інших трикутних чисел:

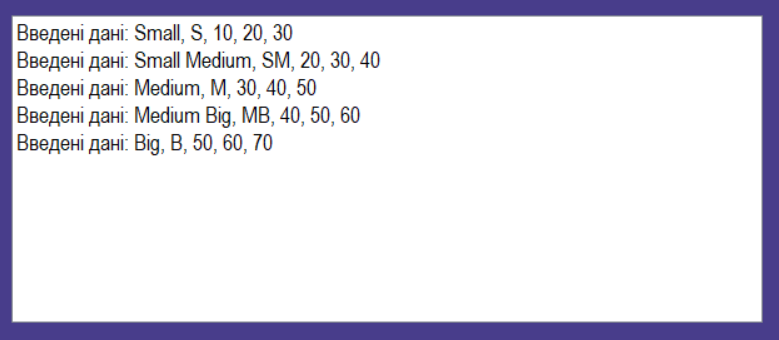
 

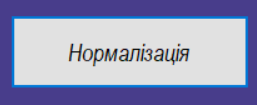
Переглядаємо побудовані на графіку трикутні числа:



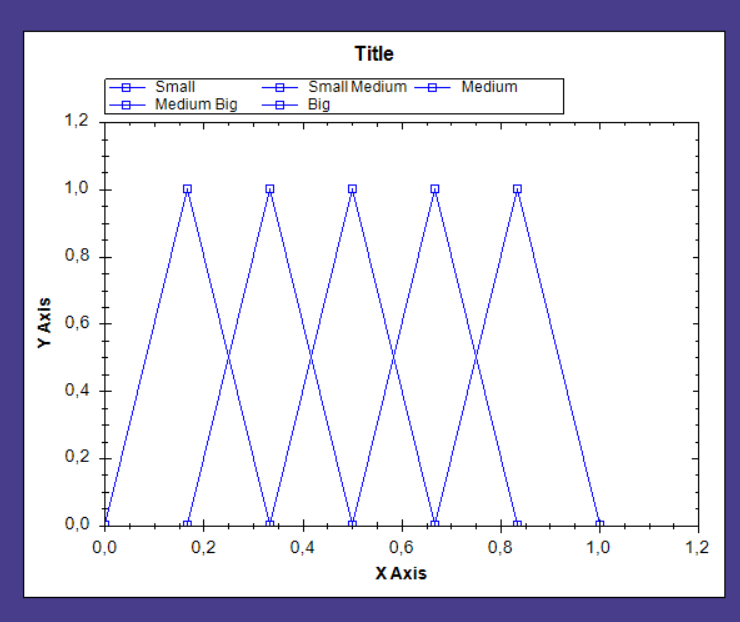
Введені дані також відображаються у списку:



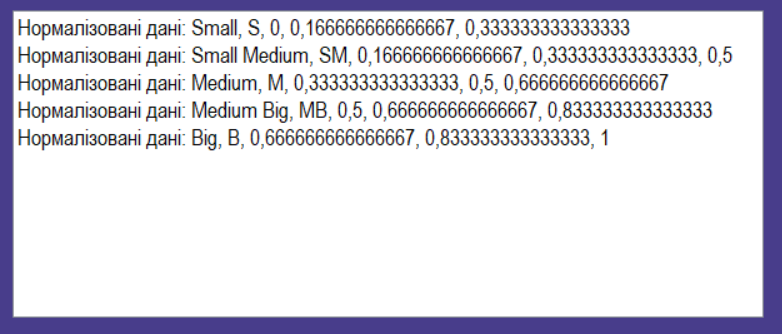
Для нормалізації трикутних чисел натискаємо на кнопку нормалізації:



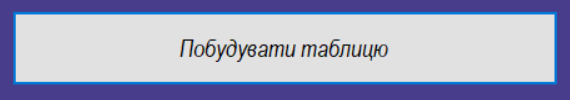
Переглядаємо нормалізацію на графіку:



Її також можна переглянути у списку, де введені дані стали нормалізованими:



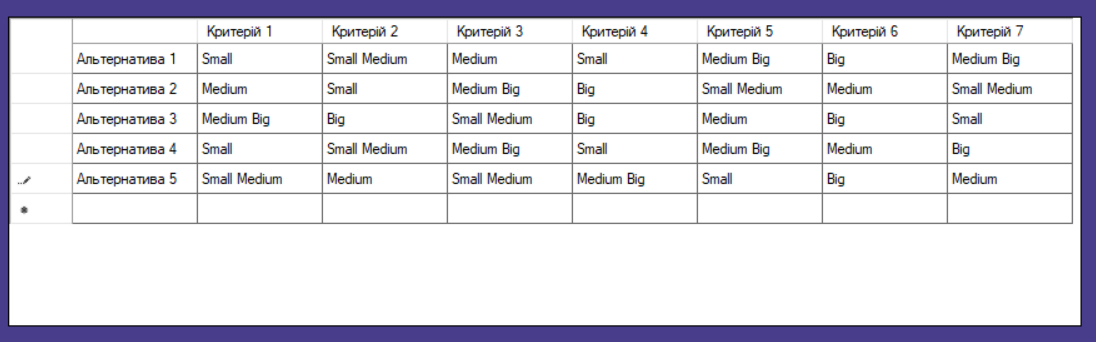
Далі натискаємо на кнопку побудови таблиці:



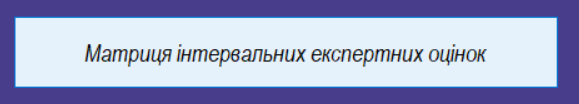
Відображається пуста таблиця зі вказаною кількістю критеріїв та альтернатив:



Заповнюємо дану таблицю введеними термами, тобто повними назвами термів:



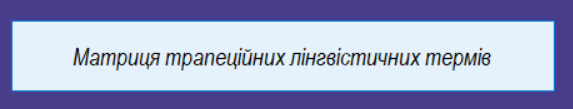
Після заповнення натискаємо на кнопку перетворення даної таблиці у матрицю інтервальних експертних оцінок:



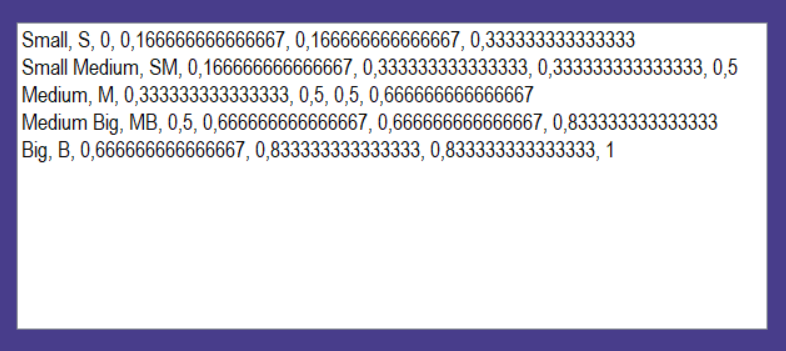
Після цього бачимо дану матрицю інтервальних експертних оцінок, яка містить скорочені назви термів:



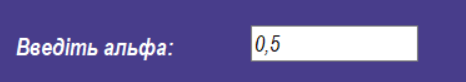
Потім натискаємо на перетворення на матрицю трапеційних лінгвістичних термів:



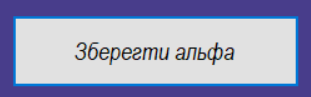
Дана матриця відображається в списку:



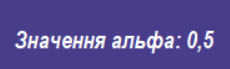
Потім вводимо значення альфа:



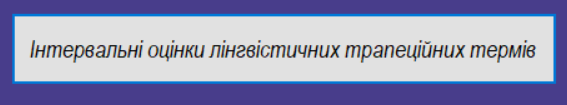
Зберігаємо його:



Після збереження дане значення відображається у текстовому форматі:



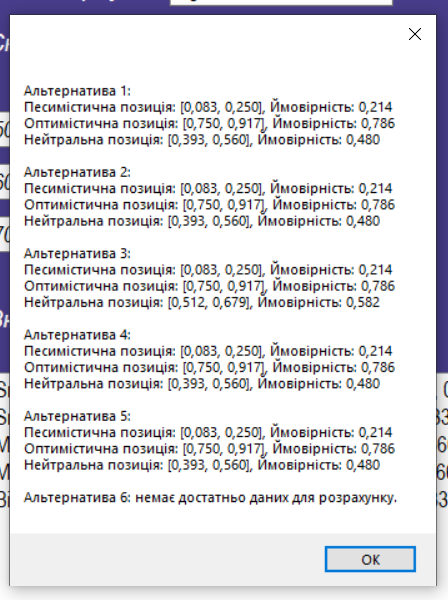
Після чого натискаємо на перетворення матриці трапеційних ЛТ у інтервальні оцінки цих лінгвістичних трапеційних термів:



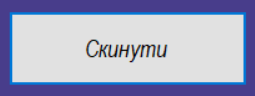
Інтервальні оцінки трапеційних ЛТ відображаються в таблиці:



Крім того, в результаті відкривається вікно з оптимістичними, песимістичними та нейтральними позиціями для кожної альтернативи:



Можна також видалити всі дані з програми, не завершуючи її, натиснувши кнопку скидання:



Таким чином, програма виводить графіки для кожного введеного трикутного числа, нормалізує їх, будує таблицю, виводить необхідні дані в таблицю та в окремий список, будує інтервальні експертні оцінки, матрицю трапеційних ЛТ та матрицю інтервальних експертних оцінок трапеційних ЛТ, а також виводить результати щодо позицій та ймовірностей для кожної альтернативи.

**Висновки:**

Запропонований метод агрегації узагальнених трапеційних лінгвістичних термів дозволяє спростити процес вибору найкращого альтернативного варіанту та підвищити ефективність, зокрема швидкодію процесів багатокритерійного прийняття рішень. Дане твердження базується на основі порівняльного аналізу часової тривалості обчислювальних операцій при реалізації відповідних методів. Апробація методу агрегації узагальнених трапеційних термів підтверджує високу ефективність, в т.ч. при розв’язанні багатокритерійної задачі певного вибору з множини існуючих альтернативних варіантів.

**Лістинг коду:**

using System;

using System.Collections.Generic;

using System.Drawing;

using System.Linq;

using System.Windows.Forms;

using ZedGraph;

using System.Globalization;

using System.Text;

namespace LR1Stasiia

{

public partial class Form1 : Form

{

private Dictionary<string, string> termMapping = new Dictionary<string, string>();

private List<TrapezoidalNumber> trapezoidalNumbers = new List<TrapezoidalNumber>();

public Form1()

{

InitializeComponent();

}

private void buttonBuild\_Click(object sender, EventArgs e)

{

float left = float.TryParse(textBoxLeft.Text, out float l) ? l : 0;

float middle = float.TryParse(textBoxMiddle.Text, out float m) ? m : 0;

float right = float.TryParse(textBoxRight.Text, out float r) ? r : 0;

string termName = textBoxTermName.Text.Trim();

string shortName = textBoxShortName.Text.Trim();

if (string.IsNullOrWhiteSpace(termName))

{

MessageBox.Show("Введіть назву терму.", "Помилка", MessageBoxButtons.OK, MessageBoxIcon.Error);

return;

}

if (!termMapping.ContainsKey(termName))

{

termMapping[termName] = shortName;

}

trapezoidalNumbers.Add(new TrapezoidalNumber(termName, shortName, left, middle, right));

Color[] colors = { Color.Red, Color.Orange, Color.Green, Color.Blue, Color.Purple };

int triangleCount = zedGraphControl1.GraphPane.CurveList.Count;

Color color = colors[triangleCount % colors.Length];

PointPairList trianglePoints = new PointPairList();

trianglePoints.Add(left, 0);

trianglePoints.Add(middle, 1);

trianglePoints.Add(right, 0);

LineItem triangleCurve = zedGraphControl1.GraphPane.AddCurve(termName, trianglePoints, color, SymbolType.None);

zedGraphControl1.AxisChange();

zedGraphControl1.Invalidate();

string displayText = $"Введені дані: {termName}, {shortName}, {left}, {middle}, {right}";

listBoxNormalizedIntervals.Items.Add(displayText);

}

private void BuildGraph()

{

zedGraphControl1.GraphPane.CurveList.Clear();

foreach (var trapezoidal in trapezoidalNumbers)

{

double[] x = new double[] { trapezoidal.Left, trapezoidal.Middle, trapezoidal.Right };

double[] y = new double[] { 0, 1, 0 };

zedGraphControl1.GraphPane.AddCurve(trapezoidal.Name, x, y, System.Drawing.Color.Blue);

}

zedGraphControl1.AxisChange();

zedGraphControl1.Invalidate();

}

private void buttonCreateTable\_Click(object sender, EventArgs e)

{

int criteriaCount = int.TryParse(textBoxCriteria.Text.Trim(), out int criteria) ? criteria : 0;

int alternativesCount = int.TryParse(textBoxAlternatives.Text.Trim(), out int alternatives) ? alternatives : 0;

dataGridView1.Columns.Clear();

dataGridView1.Rows.Clear();

dataGridView1.Columns.Add("", "");

for (int i = 0; i < criteriaCount; i++)

{

dataGridView1.Columns.Add($"Criterion{i + 1}", $"Критерій {i + 1}");

}

for (int i = 0; i < alternativesCount; i++)

{

dataGridView1.Rows.Add($"Альтернатива {i + 1}");

}

if (dataGridView1.Rows.Count > 0)

{

dataGridView1.Rows[0].Cells[0].Value = "Альтернатива 1";

}

if (termMapping.Count == 0)

{

MessageBox.Show("Не введено жодного терму!", "Попередження", MessageBoxButtons.OK, MessageBoxIcon.Warning);

}

}

private void buttonNormalize\_Click(object sender, EventArgs e)

{

if (trapezoidalNumbers.Count == 0)

{

MessageBox.Show("Не знайдено жодного трикутного числа для нормалізації!", "Попередження", MessageBoxButtons.OK, MessageBoxIcon.Warning);

return;

}

double min = double.MaxValue;

double max = double.MinValue;

foreach (var trapezoidal in trapezoidalNumbers)

{

min = Math.Min(min, trapezoidal.Left);

min = Math.Min(min, trapezoidal.Middle);

min = Math.Min(min, trapezoidal.Right);

max = Math.Max(max, trapezoidal.Left);

max = Math.Max(max, trapezoidal.Middle);

max = Math.Max(max, trapezoidal.Right);

}

listBoxNormalizedIntervals.Items.Clear();

for (int i = 0; i < trapezoidalNumbers.Count; i++)

{

var trapezoidal = trapezoidalNumbers[i];

double normalizedLeft = (trapezoidal.Left - min) / (max - min);

double normalizedMiddle = (trapezoidal.Middle - min) / (max - min);

double normalizedRight = (trapezoidal.Right - min) / (max - min);

trapezoidalNumbers[i] = new TrapezoidalNumber(trapezoidal.Name, trapezoidal.ShortName, normalizedLeft, normalizedMiddle, normalizedRight);

listBoxNormalizedIntervals.Items.Add($"Нормалізовані дані: {trapezoidal.Name}, {trapezoidal.ShortName}, {normalizedLeft}, {normalizedMiddle}, {normalizedRight}");

}

BuildGraph();

}

private void buttonReset\_Click(object sender, EventArgs e)

{

textBoxCriteria.Clear();

textBoxAlternatives.Clear();

textBoxTerms.Clear();

textBoxLeft.Clear();

textBoxMiddle.Clear();

textBoxRight.Clear();

textBoxTermName.Clear();

textBoxShortName.Clear();

textBoxAlpha.Clear();

zedGraphControl1.GraphPane.CurveList.Clear();

zedGraphControl1.Invalidate();

listBoxNormalizedIntervals.Items.Clear();

trapezoidalNumbers.Clear();

foreach (DataGridViewRow row in dataGridView1.Rows)

{

foreach (DataGridViewCell cell in row.Cells)

{

cell.Value = string.Empty;

}

}

}

private void buttonConvertToShortNames\_Click(object sender, EventArgs e)

{

for (int i = 0; i < dataGridView1.Rows.Count; i++)

{

for (int j = 1; j < dataGridView1.Columns.Count; j++)

{

string fullTermName = dataGridView1.Rows[i].Cells[j].Value?.ToString();

if (!string.IsNullOrEmpty(fullTermName) && termMapping.ContainsKey(fullTermName))

{

string shortTermName = termMapping[fullTermName];

dataGridView1.Rows[i].Cells[j].Value = shortTermName;

}

else if (!string.IsNullOrEmpty(fullTermName) && !termMapping.ContainsKey(fullTermName))

{

MessageBox.Show($"Терм '{fullTermName}' не знайдено.", "Попередження", MessageBoxButtons.OK, MessageBoxIcon.Warning);

}

}

}

}

private void buttonConvertToTrapezoidal\_Click(object sender, EventArgs e)

{

listBoxNormalizedIntervals.Items.Clear();

HashSet<string> addedTerms = new HashSet<string>();

for (int i = 0; i < dataGridView1.Rows.Count; i++)

{

for (int j = 1; j < dataGridView1.Columns.Count; j++)

{

string shortTermName = dataGridView1.Rows[i].Cells[j].Value?.ToString();

if (!string.IsNullOrEmpty(shortTermName) && termMapping.ContainsValue(shortTermName))

{

string fullTermName = termMapping.FirstOrDefault(x => x.Value == shortTermName).Key;

var trapezoidal = trapezoidalNumbers.FirstOrDefault(t => t.Name == fullTermName);

if (trapezoidal != null && !addedTerms.Contains(trapezoidal.Name))

{

addedTerms.Add(trapezoidal.Name);

double trapezoidalLeft = trapezoidal.Left;

double trapezoidalMiddle = trapezoidal.Middle;

double trapezoidalRight = trapezoidal.Right;

string displayText = $"{trapezoidal.Name}, {trapezoidal.ShortName}, {trapezoidalLeft}, {trapezoidalMiddle}, {trapezoidalMiddle}, {trapezoidalRight}";

listBoxNormalizedIntervals.Items.Add(displayText);

}

}

}

}

}

private double alpha;

private void buttonTrapezIntervalRatings\_Click(object sender, EventArgs e)

{

StringBuilder resultBuilder = new StringBuilder();

for (int i = 0; i < dataGridView1.Rows.Count; i++)

{

List<double> leftValues = new List<double>();

List<double> rightValues = new List<double>();

for (int j = 1; j < dataGridView1.Columns.Count; j++)

{

string shortTermName = dataGridView1.Rows[i].Cells[j].Value?.ToString();

if (!string.IsNullOrEmpty(shortTermName) && termMapping.ContainsValue(shortTermName))

{

string fullTermName = termMapping.FirstOrDefault(x => x.Value == shortTermName).Key;

var trapezoidal = trapezoidalNumbers.FirstOrDefault(t => t.Name == fullTermName);

if (trapezoidal != null)

{

double trapezoidalLeft = trapezoidal.Left;

double trapezoidalMiddle = trapezoidal.Middle;

double trapezoidalRight = trapezoidal.Right;

double newLeft = (alpha \* (trapezoidalMiddle - trapezoidalLeft)) + trapezoidalLeft;

double newRight = trapezoidalRight - (alpha \* (trapezoidalRight - trapezoidalMiddle));

leftValues.Add(newLeft);

rightValues.Add(newRight);

string newCellValue = $"{newLeft:F3}, {newRight:F3}";

dataGridView1.Rows[i].Cells[j].Value = newCellValue;

}

}

}

if (leftValues.Count > 0 && rightValues.Count > 0)

{

double minLeft = leftValues.Min();

double minRight = rightValues.Min();

double pessimisticProbability = CalculateProbability(minLeft, minRight);

double maxLeft = leftValues.Max();

double maxRight = rightValues.Max();

double optimisticProbability = CalculateProbability(maxLeft, maxRight);

double avgLeft = leftValues.Average();

double avgRight = rightValues.Average();

double neutralProbability = CalculateProbability(avgLeft, avgRight);

resultBuilder.AppendLine($"Альтернатива {i + 1}:");

resultBuilder.AppendLine($"Песимістична позиція: [{minLeft:F3}, {minRight:F3}], Ймовірність: {pessimisticProbability:F3}");

resultBuilder.AppendLine($"Оптимістична позиція: [{maxLeft:F3}, {maxRight:F3}], Ймовірність: {optimisticProbability:F3}");

resultBuilder.AppendLine($"Нейтральна позиція: [{avgLeft:F3}, {avgRight:F3}], Ймовірність: {neutralProbability:F3}");

resultBuilder.AppendLine();

}

else

{

resultBuilder.AppendLine($"Альтернатива {i + 1}: немає достатньо даних для розрахунку.");

}

}

MessageBox.Show(resultBuilder.ToString());

}

private double CalculateProbability(double left, double right)

{

return Math.Max(1 - Math.Max((1 - left) / (right - left + 1), 0), 0);

}

private void textBoxAlpha\_TextChanged(object sender, EventArgs e)

{

if (double.TryParse(textBoxAlpha.Text, NumberStyles.Any, CultureInfo.CurrentCulture, out double parsedAlpha))

{

if (parsedAlpha >= 0 && parsedAlpha <= 1)

{

alpha = parsedAlpha;

}

else

{

alpha = 0.5;

}

}

else

{

alpha = 0.5;

}

}

private void listBoxNormalizedIntervals\_SelectedIndexChanged(object sender, EventArgs e)

{

if (listBoxNormalizedIntervals.SelectedItem != null)

{

textBoxAlpha.Text = listBoxNormalizedIntervals.SelectedItem.ToString();

}

}

private void buttonSaveAlpha\_Click(object sender, EventArgs e)

{

if (string.IsNullOrWhiteSpace(textBoxAlpha.Text))

{

MessageBox.Show("Будь ласка, введіть значення альфа.");

return;

}

double alpha;

if (double.TryParse(textBoxAlpha.Text, out alpha))

{

if (alpha < 0 || alpha > 1)

{

MessageBox.Show("Будь ласка, введіть значення альфа в межах від 0 до 1.");

return;

}

listBoxNormalizedIntervals.Items.Add($"Значення альфа: {alpha}");

textBoxAlpha.Clear();

}

else

{

MessageBox.Show("Будь ласка, введіть коректне числове значення для альфа.");

}

}

private void buttonAlphaPositionValue\_Click(object sender, EventArgs e)

{

labelAlphaValue.Text = $"Значення альфа: {alpha}";

}

}

public class TrapezoidalNumber

{

public string Name { get; set; }

public string ShortName { get; set; }

public double Left { get; set; }

public double Middle { get; set; }

public double Right { get; set; }

public TrapezoidalNumber(string name, string shortName, double left, double middle, double right)

{

Name = name;

ShortName = shortName;

Left = left;

Middle = middle;

Right = right;

}

}

}