Практичне заняття №2

Тема: Проектування СППР на основі нечіткої логіки. Порівняльне дослідження роботи алгоритмів Мамдані та Сугено в однотипних додатках

Мета: Вміти проектувати СППР на основі нечіткої логіки, а саме системи нечіткого виводу на основі алгоритму Мамдані та Сугено.

Теоретична частина

Створення та настройка експертної системи з використанням Fuzzy Logic Toolbox

Fuzzy Logic Toolbox – це пакет прикладних програм, що входять до складу середовища MatLab. Він дозволяє створювати системи нечіткого логічного виведення і нечіткої класифікації в рамках середовища MatLab з можливістю їх інтеграції в Simulink.

Основні властивості:

- визначення змінних, нечітких правил і функцій належності;
- інтерактивний перегляд нечіткого логічного виведення;
- сучасні методи: адаптивне нечітке виведення з використанням нейронних мереж, нечітка кластеризація;
- інтерактивне динамічне моделювання в Simulink;
- генерація переносного *C* коду за допомогою Real-Time Workshop.

Пакет *Fuzzy Logic* містить п’ять графічних редакторів для представлення необхідної інформації в процесі проектування, створення і тестування нечітких моделей.

Пакет *Fuzzy Logic* містить сучасні методи нечіткого моделювання, включаючи:

– адаптивне нечітке виведення з використанням нейронних мереж для автоматичного формування функції належності в процесі навчання їх на вхідних даних;

– нечітку логіку і кластеризацію для задач розпізнавання образів;

– можливість вибору широко відомого метода Мамдані або метода Сугено для створення гібридних нечітких систем.

Пакет дозволяє роботу:

- у режимі графічного інтерфейсу;
- у режимі командного рядка;
- з використанням блоків та прикладів пакета Simulink.

Базовим поняттям *Fuzzy Logic Toolbox* є *FIS-структура* – система нечіткого виведення (*Fuzzy Inference System*). *FIS-структура* містить усі необхідні дані для реалізації функціонального відображення “входи-виходи” на основі нечіткого логічного виведення згідно зі схемою, наведеною на рис. 1.

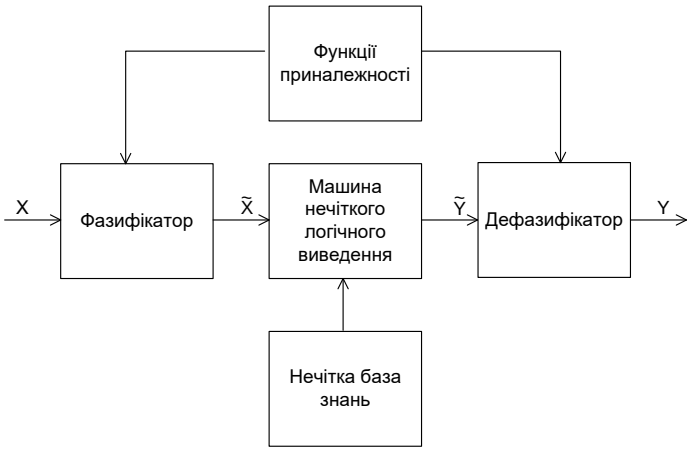

Рисунок 1 – Нечітке логічне виведення

Позначення: X – вхідний чіткий вектор; $\tilde{X}$ – вектор нечітких множин, що відповідає вхідному вектору X; $\tilde{Y}$ – результат логічного виведення у вигляді вектора нечітких множин; Y – вихідний чіткий вектор.

Склад графічного інтерфейсу

Fuzzy Logic Toolbox містить наступні редактори:

– редактор нечіткої системи виведення *Fuzzy Inference System Editor* (*FIS Editor* або *FIS-редактор*) разом з додатковими програмами – редактором функцій належності (*Memberhip Function Editor*), редактором правил (*Rule Editor*), вікно перегляду правил (*Rule Viewer*) і вікном перегляду поверхні відгуку (*SurfaceViewer*);

– редактор гібридних систем (*ANFIS Editor, ANFIS-редактор*);

– програма знаходження кластерів (програма *Clustering* – кластеризація).

Редактор нечіткої системи виведення

Командою (функцією) *fuzzy* з режиму командного рядка запускається основна інтерфейсна програма пакета *Fuzzy Logic* – редактор нечіткої системи виведення. Головне вікно наведено на рисунку 2.

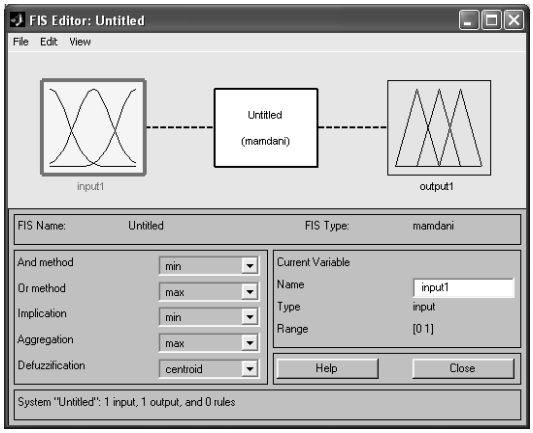

Рисунок 2 – Вигляд вікна *FIS Editor*

Графічний інтерфейс гібридних мереж

Головне вікно редактора *ANFIS Editor* викликається командою *anfisedit* з командного рядка, вигляд якого наведено на рисунку 1.3.

За допомогою даного редактора виконується створення або завантаження гібридної системи, перегляд структури, настроювання її параметрів, перевірка якості функціонування такої системи.

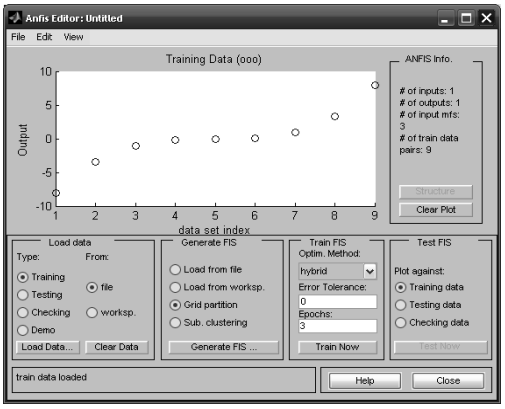

Рисунок 3 – Вікно редактора гібридних систем

Графічний інтерфейс програми кластеризації

Програма *Clustering* (кластеризація) дозволяє виявляти центри кластерів, тобто точки в багатовимірному просторі даних, біля яких групуються (скупчуються) експериментальні дані.

Запуск програми *Clustering* виконується командою *findcluster*. На рисунку 4 наведено приклад використання програми.

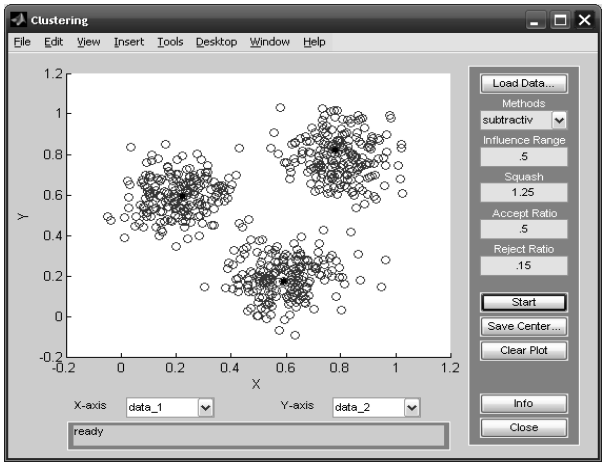

Рисунок 4 – Результат роботи програми *Clustering*

Робота в редакторі нечіткої системи виведення Fuzzy Inference System Editor

Для завантаження основного *fis-редактора* надрукуємо слово *fuzzy* в командному рядку. Після цього відкриється нове графічне вікно, зображене на рисунку 1.2. Для того щоб додати нову вхідну змінну, необхідно в меню *Edit* вибрати команду *Add Variable…\Input*. Для зміни імені змінної необхідно ввести нове ім’я в полі *Name* і натиснути клавішу *Enter*. Для того щоб задати ім’я системі, необхідно в меню *File* вибрати в підменю *Export* команду *To File* і ввести ім'я файла.

Щоб перейти в редактор функцій приналежності, необхідно двічі натиснути на будь-якій з функцій, де можна вибирати властивості конкретної, вікно відображено на рисунку 5:

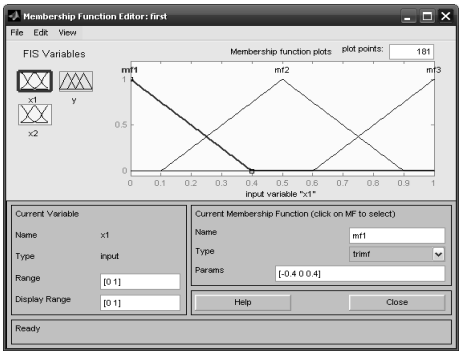

Рисунок 5 – Редактор функцій належності

Внизу вікна вказуються наступні властивості функції:

Current Variable:

- Name – ім’я функції;
- Type – тип (вхідна чи вихідна);
- Range – діапазон змінної;
- Display Range – відображуваний діапазон.

Current Membership Function:

- Name – ім’я поточної функції належності;
- Type – тип терму функції належності – вибирається з переліку (трикутна, трапецеїдальна, ґауссові 1 та 2-го порядку та інші);
- Params – числові значення терму функції належності.

Для задання нових функцій належності для змінної необхідно в меню *Edit* вибрати команду *Add MFs...* У результаті з'явитися діалогове вікно (рисунок 6) вибору типу і кількості функцій належності.

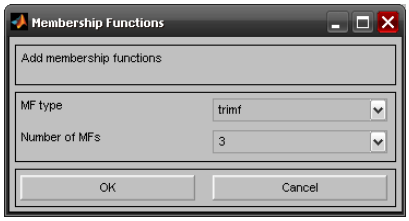

Рисунок 6 – Задання функцій належності

Вибравши необхідну кількість термів та їх тип, натиснути OK.

Ім’я та числове значення термів можна змінити, виділивши необхідний, і задати у відповідних полях області *Current Membership Function* нові значення.

Редактор бази знань RuleEditor

Для виклику редактора необхідно вибрати в меню *Edit* команду ***Rules...***, відобразиться головне вікно, зображене на рисунку 7.

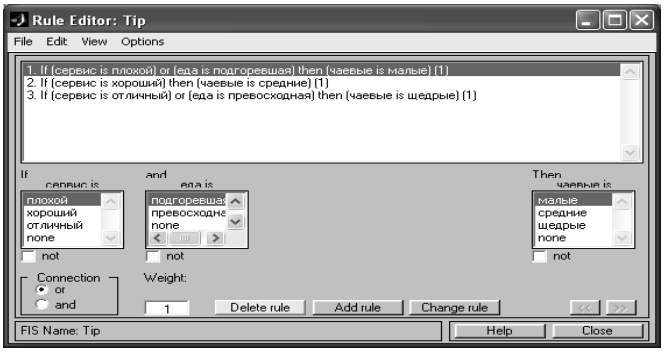

Рисунок 7 – Редактор правил

Для створення нових правил необхідно вибрати відповідну комбінацію термів і залежностей, вибрати тип зв’язку: *or* або *and*, вагу правила *Weight*, значення вихідної змінної та натиснути кнопку *Add rule*.

Для перегляду вікна візуалізації нечіткого логічного виведення викликаємо його командою *View rules...* меню *View*.

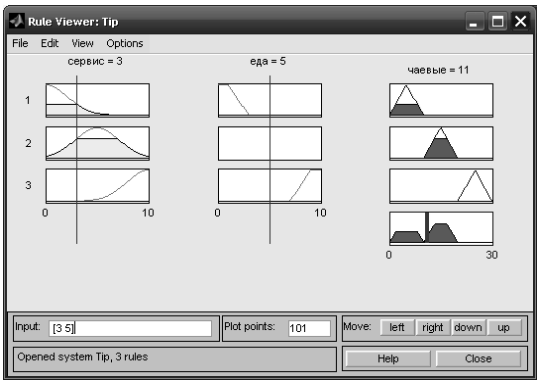

Рисунок 8 – Візуалізація нечіткого логічного виведення в *RuleViewer*

Можна переглянути поверхню “входи-виход”, відповідну синтезованій нечіткій системі. Для виведення цього вікна необхідно використовувати команду *View surface...* меню *View*.

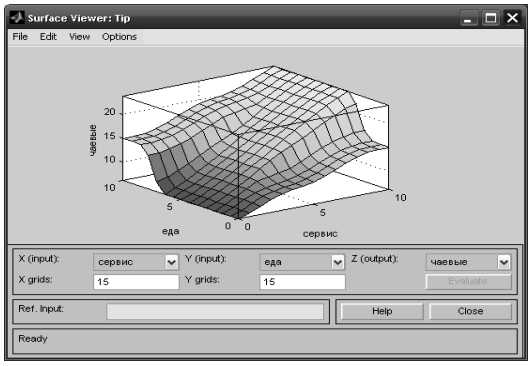

Рисунок 9 – Поверхня відгуку нечіткої системи

Алгоритми нечіткого виведення різняться, головним чином, видом використовуваних правил, логічних операцій і різновидом методу дефазифікації. Розроблені моделі нечіткого виведення Мамдані, Сугено, Ларсена, Цукамото. При розгляді алгоритмів для спрощення припустимо, що базу знань організують два нечітких правила вигляду:

$П_1$: якщо x є A_1 та y є B_1, то z є C_1,

$П_2$: якщо x є A_2 та y є B_2, то z є C_2,

де x і y – імена вхідних змінних, z – ім'я змінної виведення, A_1, A_2, B_1, B_2, C_1, C_2 – деякі задані функції належності, при цьому чітке значення z_0 необхідно визначити на основі наведеної інформації та чітких значень x_0 і y_0.

Алгоритм Мамдані (Mamdani)

Алгоритм Мамдані є одним з перших, який знайшов застосування в системах нечіткого виведення. Він був запропонований 1975 р. англійським математиком Е. Мамдані (Ebrahim Mamdani) як метод для керування паровим двигуном. Формально *алгоритм Мамдані* може бути визначений таким чином.

1. Процедура фазифікації: визначаються ступені істинності, тобто значення функцій належності для лівих частин кожного правила (передумов): $A_1(x_0)$, $A_2(x_0)$, $B_1(y_0)$, $B_2(y_0)$.
2. Нечітке виведення: знаходяться рівні відтинання для передумов кожного з правил з використанням операції мінімум:
   $\alpha_1 = A_1(x_0) \wedge B_1(y_0)$,
   $\alpha_2 = A_2(x_0) \wedge B_2(y_0)$,

де через «$\wedge$» позначена операція логічного мінімуму (min), потім знаходяться «зрізані» функції належності

$C_1'(z) = (\alpha_1 \wedge C_1(z))$,

$C_2'(z) = (\alpha_2 \wedge C_2(z))$.

3. Композиція: з використанням операції максимуму (max, позначення: «$\vee$») виконується об'єднання знайдених зрізаних функцій, що приводить до отримання підсумкової нечіткої підмножини для змінної виходу з функцією належності

$$\mu\Sigma(z) = C(z) = C1'(z) \vee C2'(z) = (\alpha 1 \wedge C1(z)) \vee (\alpha 2 \wedge C2(z)).$$

4. Приведення до чіткості (для знаходження z0) проводиться, наприклад, центроїдним методом (як x – координата центра ваги функції належності підсумкової нечіткої підмножини для змінної виходу):

$$z_0 = \frac{\int_{\Omega} z \cdot \mu_{\Sigma}(z) dz}{\int_{\Omega} \mu_{\Sigma}(z) dz}$$

Алгоритм ілюструється (рис. 10):

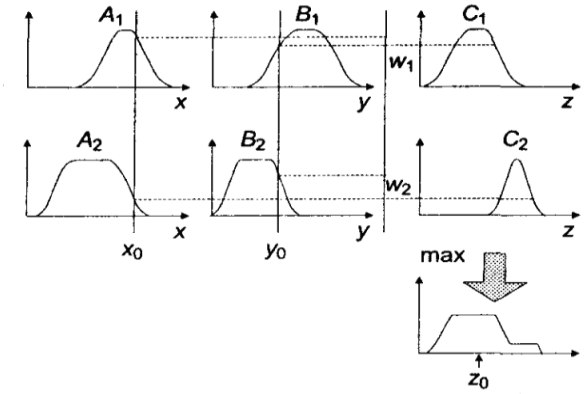

Рисунок 10 – Графічна реалізація

Алгоритм Сугено (Sugeno)

Формально алгоритм Сугено, запропонований Сугено і Такагі, може бути визначений таким чином.

1. Перший етап – як в алгоритмі Мамдані.
2. На другому етапі знаходяться $\alpha 1 = A1(x0) \wedge B1(y0)$, $\alpha 2 = A2(x0) \wedge B2(y0)$ та індивідуальні виходи правил:

$$z^*_1 = a_1x_0 + b_1y_0,$$
$$z^*_2 = a_2x_0 + b_2y_0,$$

3. На третьому етапі визначається чітке значення змінної виведення:

$$z_0 = \frac{\alpha_1 z_1^* + \alpha_2 z_2^*}{\alpha_1 + \alpha_2}$$

Алгоритм ілюструється на рисунку 11:

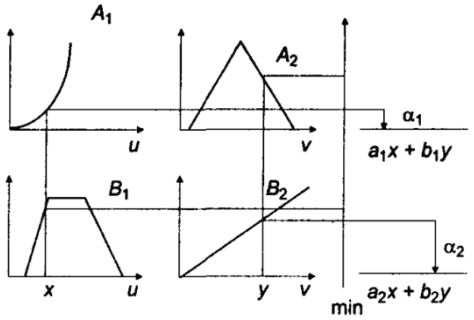

Рисунок 11 – Графічна реалізація алгоритму Сугено

Завдання до виконання

1. Порівняти алгоритми Мамдані і Сугено на прикладі створення системи нечіткого логічного виведення, що моделює залежність $y = x_1^2 \cdot \sin(x_2 - 1)$, $x_1 \in [-7,\ 3]$, $x_2 \in [-4.4,\ 1.7]$. Проектування системи нечіткого логічного виведення необхідно провести на основі графічного зображення вказаної залежності. (**Кожен вибирає свою залежність і проходить пункти 2-12**).

2. Для побудови тривимірного зображення функції $y = x_1^2 \cdot \sin(x_2 - 1)$ в межах $x_1 \in [-7,\ 3]$, $x_2 \in [-4.4,\ 1.7]$ необхідно скласти наступну програму, прописавши її в m-файлі:

```
%Побудова графіка функції y = x1^2*sin(x2-1) в межах x1є[-7, 3] і x2є[-4.4, 1.7].
n = 15;                    % кількість точок
x1 = -7:10/(n-1):3;              % задання параметрів змінної x1
x2 = -4.4:6.1/(n-1):1.7;         % задання параметрів змінної x1
y = zeros (n, n);  % формування нульового масиву
  % розміром n×n для вихідної змінної
for j = 1:n
y (j,:) = x1.^2*sin(x2(j)-1);
```

```
end
surf (x1, x2, y)          % зображення поверхні функції
xlabel ('x1')
ylabel ('x2')
zlabel ('y')
title ('Target');
```

У результаті виконання програми отримаємо графічне зображення, наведене на рис. 12.

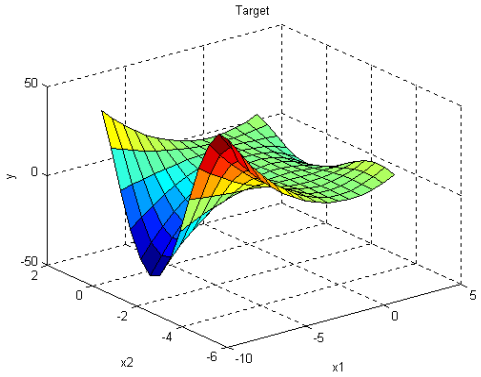

Рисунок 12 – Еталонна поверхня

3. Реалізуйте дві нечіткі системи для заданої функції, вибравши для першої тип системи Сугено, для другої – Мамдані.

4. Для створення першої системи завантажте fis-редактор. Виберіть тип системи – Sugeno. Додайте другу вхідну змінну та назвіть усі змінні відповідними іменами, а саме першу вхідну змінну перейменуйте на x1, другу – на x2, а вихідну – на y.

5. Перейдіть у редактор функцій належності. Задайте діапазон змінення змінної x1 та створіть для неї функції належності, вказавши ім’я, тип та числові значення термів:

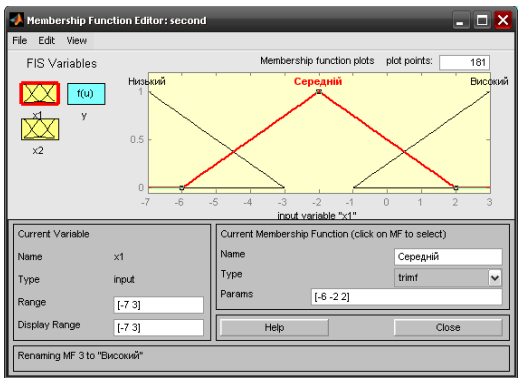

Рисунок 13 – Функції належності змінної x1

6. Аналогічно задайте діапазон змінення змінної x2 та створіть для неї функції належності, вказавши ім’я, тип та числові значення термів:

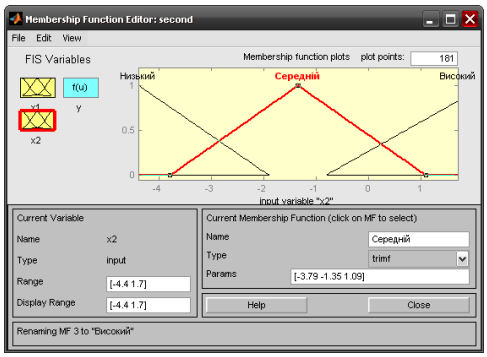

Рисунок 14– Функції приналежності змінної x2

7. Для алгоритму Сугено для вихідної змінної задаються лінійні залежності між входами і виходом, що мають міститися в базі знань. У базі знань вказано 5 різних залежностей: y=50; y=4x1-x2; y=2x1+2x2+1; y=8x1+2x2+8; y=0. Тому додайте ще дві залежності шляхом вибору команди *Add Mfs* меню *Edit*. У діалоговому вікні в полі *Number of MFs* виберіть 2 і натисніть кнопку OK. Задайте найменування і параметри лінійних залежностей. Для цього виберіть першу залежність mf1. Надрукуйте найменування залежності, наприклад 50, у полі *Name* і встановіть тип залежності - константа шляхом вибору опції Constant в меню *Type*. Після цього введіть значення параметра 50 у полі *Params*. Аналогічно для другої залежності mf2 введіть найменування залежності, наприклад 8+8x1+2x2. Потім вкажіть лінійний тип залежності шляхом вибору опції Linear у меню *Type* і введіть параметри залежності 8 2 8 в полі *Params*. Для лінійної залежності порядок параметрів наступний: перший параметр – коефіцієнт при першій змінній, другий, – при другій і т. д., останній параметр – вільний член залежності. У результаті маєте отримати графічне вікно, зображене на рис.15.

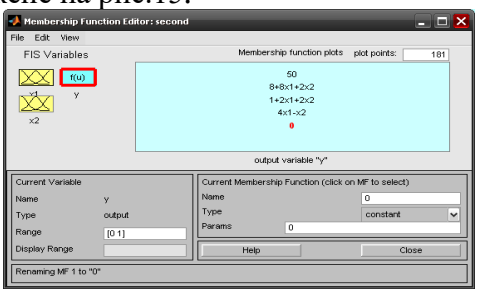

Рисунок 15– Вікно лінійних залежностей “входи-вихід”

8. Аналізуючи еталонну поверхню, можна скласти наступні залежності та правила:

якщо x1=середній, то y=0;

якщо x1=високий і x2=високий, то y=2x1+2x2+1;

якщо x1=високий і x2=низький, то y=4x1-x2;

якщо x1=низький і x2=середний, то y=8x1+2x2+8;

якщо x1=низький і x2=низький, то y=50;

якщо x1=низький і x2=високий, то y=50.

9. Перейдіть у редактор бази знань *RuleEditor* і введіть правила бази знань, що наведені вище.

10. Перегляньте вікно візуалізації нечіткого логічного виведення, а також поверхню “входи–вихід” для синтезованої нечіткої системи.

11. Реалізуйте нечітку логічну систему для заданої функції, використовуючи алгоритм Мамдані. Самостійно складіть правила для відповідних функцій.

12. Порівняйте отримані різними методами поверхні з еталонною поверхнею. Зробіть висновки щодо ефективності кожної з них.

Зміст звіту

1. Указати номер, тему й мету лабораторної роботи.
2. Зобразити FIS-структури для розроблених систем за різними алгоритмами.
3. Навести перелік правил.
4. Відобразити отримані результати – перехідні процеси, поверхні відгуку.
5. Зробити порівняльні висновки стосовно роботи системи з різним настройками.