

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Західноукраїнський національний університет
Факультет комп'ютерних інформаційних технологій
Кафедра комп'ютерної інженерії

Кудибин Марина Сергіївна

**Нечіткий модуль оцінки вартості ІТ-проєктів
на основі функціональних точок / Fuzzy
module for estimating the cost of IT projects
based on functional points**

спеціальність: 123 – Комп'ютерна інженерія
освітньо-професійна програма – Комп'ютерна інженерія

Кваліфікаційна робота

Виконала: студентка групи КІ-41
Кудибин Марина Сергіївна

Науковий керівник
к.т.н. доцент Савка Н.Я.

ТЕРНОПІЛЬ - 2026

АНОТАЦІЯ

Кваліфікаційна робота присвячена розробленню нечіткої моделі оцінювання вартості IT-проєкту на основі поєднання методу функціональних точок (IFPUG FPA) із нечітким виводом за алгоритмом Мамдані. У роботі проаналізовано предметну область оцінювання вартості програмного забезпечення, виконано порівняльний огляд класичних підходів (COCOMO, COCOMO II, метод Альбрехта, метод Патнема), обґрунтовано доцільність застосування нечіткої логіки для умов невизначеності початкових даних.

Спроектовано архітектуру нечіткої системи типу MISO з двома вхідними лінгвістичними змінними («розмір ПЗ у функціональних точках» і «коефіцієнт складності проєкту») та вихідною змінною «вартість». Сформовано базу з п'ятнадцяти правил типу «якщо–то», визначено параметри функцій належності та реалізовано всі етапи нечіткого виводу — фазифікацію, активацію правил, агрегування і дефазифікацію методом центру ваги — вручну, без використання Fuzzy Logic Toolbox, що забезпечує сумісність із MATLAB, Octave та середовищем Simulations4All. Проведено експериментальне дослідження адекватності моделі та аналіз її чутливості до зміни коефіцієнта складності.

Кваліфікаційна робота містить вступ, три розділи, висновки, список використаних джерел і додаток із програмною реалізацією моделі.

Ключові слова: НЕЧІТКА ЛОГІКА, ВИВІД ЗА МАМДАНІ, ФУНКЦІОНАЛЬНІ ТОЧКИ, ОЦІНЮВАННЯ ВАРТОСТІ, IT-ПРОЄКТ, IFPUG, КОЕФІЦІЄНТ СКЛАДНОСТІ, ДЕФАЗИФІКАЦІЯ.

					КР.КІ. 10658707.00.00.000.ПЗ	Арк.
						9
Змн.	Арк.	№ докум.	Підпис	Дата		

ANNOTATION

The qualification thesis is devoted to the development of a fuzzy model for IT project cost estimation based on a combination of the IFPUG function point analysis method and the Mamdani fuzzy inference algorithm. The subject area of software cost estimation is analysed; classical approaches (COCOMO, COCOMO II, Albrecht's method, Putnam's method) are reviewed and the appropriateness of fuzzy logic for conditions of input data uncertainty is justified.

A MISO-type fuzzy system architecture is designed with two input linguistic variables ("software size in function points" and "project complexity factor") and one output variable ("cost"). A rule base consisting of nine if-then rules is formed, membership function parameters are defined, and all stages of the fuzzy inference (fuzzification, rule activation, aggregation, and centre-of-gravity defuzzification) are implemented manually, without using the Fuzzy Logic Toolbox, which ensures compatibility with MATLAB, Octave, and the Simulations4All environment. An experimental study of the model's adequacy and a sensitivity analysis with respect to the complexity factor are carried out.

The thesis consists of an introduction, three chapters, conclusions, a list of references, and an appendix with the program implementation of the model.

Keywords: FUZZY LOGIC, MAMDANI INFERENCE, FUNCTION POINTS, COST ESTIMATION, IT PROJECT, IFPUG, COMPLEXITY FACTOR, DEFUZZIFICATION.

					КР.КІ. 10658707.00.00.000.ПЗ	Арк.
						10
Змн.	Арк.	№ докум.	Підпис	Дата		

ЗМІСТ

Перелік умовних позначень	5
Вступ.....	6
1 Особливості формування ціни ІТ-проектів	9
1.1 Аналіз особливостей розробки ІТ-проектів	9
1.2 Методи оцінки вартості ІТ-проектів	12
1.3 Засоби моделювання	16
1.4 Постановка задачі кваліфікаційної роботи.....	19
2 Алгоритм оцінки вартості за функціональними точками	22
2.1 Метод функціональних точок	22
2.2 Алгоритм побудови нечітких моделей	25
2.3 Нечіткий алгоритм оцінки вартості ІТ-проектів.....	28
3 Нечіткий модуль оцінки вартості ІТ-проектів	31
3.1 Формалізація вхідних і вихідних даних.....	31
3.2 Нечітка модель та база правил.....	35
3.3 Експериментальні дослідження розробленої моделі.....	39
Висновки	44
Список використаних джерел	46
Додаток А Світлокопія публікації.....	50
Додаток Б Лістинг програмного коду	54
Додаток В Техніко-економічне обґрунтування	57

					КР.КІ. 10658707.00.00.000 ПЗ			
Змн.	Лист	№ докум.	Підпис	Дата				
Розробив	Кудибин М.С.				НЕЧІТКИЙ МОДУЛЬ ОЦІНКИ ВАРТОСТІ ІТ-ПРОЄКТІВ НА ОСНОВІ ФУНКЦІОНАЛЬНИХ ТОЧОК	Літ.	Арк.	Акрушів
Перевір.	Савка Н.Я.						9	67
Консульт.	Савка Н.Я.					ЗУНУ.ФКІТ. КІ-41		
Н. Контр.	Дубчак Л.О.							
Затвердив	Дубчак Л.О.							

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

COCOMO – COnstructive COst MOdel (конструктивна модель витрат);
FPA – Function Point Analysis (аналіз функціональних точок);
GSC – General System Characteristics (загальні характеристики системи);
IFPUG – International Function Point Users Group;
ILF – Internal Logical File (внутрішній логічний файл);
EIF – External Interface File (зовнішній інтерфейсний файл);
EI – External Input (зовнішнє введення);
EO – External Output (зовнішнє виведення);
EQ – External Inquiry (зовнішній запит);
ISO – International Organization for Standardization;
MAPE – Mean Absolute Percentage Error (середня абсолютна відсоткова похибка);
MISO – Multiple Input — Single Output (багато входів — один вихід);
PERT – Program Evaluation and Review Technique;
SDLC – Software Development Life Cycle (життєвий цикл розробки ПЗ);
UFP – Unadjusted Function Points (некориговані функціональні точки);
VAF – Value Adjustment Factor (коефіцієнт коригування вартості);
ВКР – випускна кваліфікаційна робота;
ЗУНУ – Західноукраїнський національний університет;
ІТ – інформаційні технології;
ОПЗ – оцінювання вартості програмного забезпечення;
ПЗ – програмне забезпечення;
ФТ – функціональні точки.

					КР.КІ. 10658707.00.00.000.ПЗ	Арк.
						12
Змн.	Арк.	№ докум.	Підпис	Дата		

ВСТУП

Управління вартістю є однією з ключових задач управління проєктами у сфері інформаційних технологій (ІТ-проєктами). Планування бюджету – один із вирішальних чинників успіху ІТ-проєкту: за даними звіту Standish Group CHAOS, понад 60 % програмних проєктів перевищують початковий кошторис, а основною причиною є помилки оцінювання вартості на ранніх стадіях, коли вимоги до системи ще не визначені остаточно. Тому розроблення методів, які дають змогу прогнозувати вартість розробки в умовах неповної та неточної інформації, є актуальною задачею програмної інженерії.

Поширеним інструментом оцінювання розміру програмного забезпечення є метод функціональних точок (Function Point Analysis, FPA) за стандартом міжнародної групи користувачів функціональних точок IFPUG (International Function Point Users Group), який вимірює обсяг функціональності продукту незалежно від мови програмування й застосовний ще до початку кодування. Проте класичні детерміновані методи погано враховують невизначеність: складність системи, кваліфікація команди та ризики реалізації мають радше якісний, ніж точний числовий характер.

За способом отримання оцінки методи оцінювання вартості поділяють на експертні, аналогові, параметричні та інтелектуальні (докладніше їх розглянуто в першому розділі). Спільним недоліком детермінованих методів є те, що вони потребують точних числових вхідних даних і повертають єдину точкову оцінку, тоді як реальна складність проєкту найчастіше описується якісно, у природномовних термінах.

Ефективним засобом роботи з такими якісно заданими параметрами є апарат нечіткої логіки, запропонований Л. Заде. Нечіткі моделі оперують лінгвістичними змінними («низька складність», «висока вартість») і базою правил «якщо–то», що дає змогу формалізувати експертні знання та отримувати обґрунтовані оцінки

					КР.КІ. 10658707.00.00.000.ПЗ	Арк.
						13
Змн.	Арк.	№ докум.	Підпис	Дата		

навіть за неповних вихідних даних. Поєднання методу функціональних точок із нечітким виводом Мамдані становить основу цієї роботи.

Тому для усунення зазначених недоліків доцільно поєднати об'єктивне вимірювання розміру методом функціональних точок з апаратом нечіткої логіки: такий гібридний підхід дає змогу враховувати якісні експертні оцінки складності й формувати обґрунтовану оцінку вартості за умов невизначеності.

Метою кваліфікаційної роботи є підвищення обґрунтованості оцінювання вартості ІТ-проектів на ранніх стадіях шляхом розроблення нечіткої моделі, що поєднує метод функціональних точок із нечітким виводом за алгоритмом Мамдані.

Для досягнення поставленої мети розв'язано такі завдання:

- проаналізувати сучасні методи оцінювання вартості програмних продуктів, виявити їхні обмеження та обґрунтувати вибір середовища нечіткого моделювання;
- розглянути метод функціональних точок як спосіб вимірювання розміру програмного забезпечення;
- вивчити засади нечіткої логіки та алгоритм нечіткого виводу Мамдані;
- розробити нечітку модель оцінювання вартості ІТ-проекту та реалізувати її у середовищі Simulations4All;
- провести експериментальні дослідження розробленої моделі (аналіз адекватності та чутливості) й оцінити отримані результати.

Об'єктом дослідження є процес оцінювання вартості програмних проектів.

Предметом дослідження є моделі та методи оцінювання вартості ІТ-проектів, що поєднують метод функціональних точок із апаратом нечіткої логіки.

Практична цінність роботи полягає в можливості застосування розробленої моделі для формування попередніх кошторисів ІТ-проектів, порівняння архітектурних варіантів та планування резервного бюджету на ризики ще до початку розробки.

Результати кваліфікаційної роботи опубліковано у тезах доповіді на ІХ Міжнародна студентська наукова конференція «Міждисциплінарні наукові дослідження та перспективи їх використання». Із копією публікації можна ознайомитись у додатку А.

					КР.КІ. 10658707.00.00.000.ПЗ	Арк.
						14
Змн.	Арк.	№ докум.	Підпис	Дата		

1 ОСОБЛИВОСТІ ФОРМУВАННЯ ЦІНИ ІТ-ПРОЄКТІВ

1.1 Аналіз особливостей розробки ІТ-проектів

Інформаційно-технологічний проєкт – це обмежена в часі діяльність, спрямована на створення унікального програмного продукту, послуги або результату цифрової трансформації організації. На відміну від проєктів у будівництві чи виробництві, де продукт має чітко вимірювані фізичні характеристики, ІТ-проект оперує нематеріальними сутностями: вимогами, архітектурою, кодом, документацією. Це визначає принципово інший характер планування, ризиків та оцінювання витрат.

Життєвий цикл ІТ-проекту охоплює п'ять основних фаз: ініціювання, планування, виконання, моніторинг і завершення. На фазі ініціювання формується бізнес-обґрунтування, окреслюється загальний обсяг та верхньорівневі вимоги. Фаза планування передбачає декомпозицію робіт, оцінку трудомісткості та формування бюджету. Фаза виконання охоплює проєктування, кодування, тестування і впровадження. Фаза моніторингу забезпечує контроль прогресу й управління ризиками. Завершення передбачає здачу-приймання продукту та архівацію проєктної документації.

Учасниками ІТ-проекту виступають замовник, керівник проєкту, бізнес-аналітики, архітектори, розробники, інженери з якості, тестувальники та фахівці з впровадження. У сучасних гнучких методологіях розробки склад команди динамічно змінюється, а ролі частково перетинаються. Це підвищує вимоги до точності планування трудовитрат і ускладнює застосування жорстко детермінованих методів оцінювання.

Особливістю ІТ-сфери є висока невизначеність вхідних даних на ранніх стадіях проєкту. Бізнес-вимоги уточнюються в процесі розробки, технологічний стек може змінюватися, а складність окремих компонентів часто стає очевидною лише після початку реалізації. За даними звіту Standish Group CHAOS, понад 60 % програмних проєктів перевищують початковий бюджет, понад 50 % виходять за

					КР.КІ. 10658707.00.00.000.ПЗ	Арк.
						15
Змн.	Арк.	№ докум.	Підпис	Дата		

межі запланованих термінів, а близько 20 % припиняють розробку через нестачу фінансування.

Окремою проблемою є залежність продуктивності ІТ-команди від великої кількості важко формалізованих чинників: кваліфікації розробників, новизни технологій, якості вимог, рівня комунікації в команді, наявності апаратної інфраструктури. Незначні відхилення у будь-якому з цих чинників можуть призводити до суттєвих змін у строках і вартості проєкту. Це робить класичні параметричні моделі оцінювання недостатньо точними та обумовлює потребу в застосуванні апарату нечіткої логіки, який природно відображає невизначеність експертних оцінок у вигляді лінгвістичних змінних.

Сучасні ІТ-проєкти прийнято класифікувати за характером кінцевого результату на кілька типів[1]. Проєкти замовної розробки передбачають створення унікального програмного продукту під конкретні вимоги замовника. Продуктові проєкти спрямовані на розроблення тиражованого програмного продукту для широкого ринку. Інтеграційні проєкти забезпечують поєднання наявних інформаційних систем у єдине середовище, а міграційні – перенесення наявних систем на нові платформи чи технології. Кожен тип має власну структуру витрат і власні чинники невизначеності, проте всім їм притаманна спільна проблема – складність достовірного оцінювання вартості на ранніх стадіях.

Структура витрат ІТ-проєкту суттєво відрізняється від проєктів матеріального виробництва. Основну частку (за різними оцінками від 60 до 75 %) становлять витрати на оплату праці фахівців, тоді як матеріальні витрати є незначними. Це означає, що точність оцінювання вартості майже повністю визначається точністю прогнозування трудомісткості – кількості людино-годин, потрібних для реалізації проєкту. Тому більшість методів оцінювання вартості ІТ-проєктів фактично є методами оцінювання трудомісткості з подальшим перерахунком у грошовий вираз.

Особливу роль відіграє так званий конус невизначеності – відома в інженерії програмного забезпечення закономірність, згідно з якою похибка оцінювання вартості є найбільшою на початку проєкту й поступово зменшується з уточненням

					КР.КІ. 10658707.00.00.000.ПЗ	Арк.
						16
Змн.	Арк.	№ докум.	Підпис	Дата		

вимог. На етапі формулювання концепції похибка оцінки може сягати чотириразового відхилення в обидва боки, на етапі затвердження вимог вона зменшується приблизно до півтораразового, і лише на етапі детального проектування наближається до прийнятних 10–15 %[2]. Парадокс полягає в тому, що рішення про доцільність і бюджет проєкту доводиться ухвалювати саме на ранніх стадіях, коли похибка є найбільшою.

Суттєвий вплив на процес оцінювання вартості справляє обрана модель життєвого циклу розробки. У межах каскадної (водоспадної) моделі вартість оцінюють одноразово на етапі планування на основі повного переліку вимог, що дає змогу сформувавши фіксований бюджет, проте робить оцінку вразливою до подальших змін вимог. Натомість гнучкі ітеративно-інкрементні методології, зокрема Scrum і Kanban, передбачають періодичне переоцінювання обсягу робіт наприкінці кожної ітерації, коли частину невизначеності вже усунено. Такий підхід підвищує адаптивність планування, проте ускладнює прогнозування підсумкової вартості всього проєкту на його старті. Незалежно від обраної методології базова проблема залишається незмінною: на момент ухвалення рішення про фінансування проєкту наявних даних недостатньо для точного детермінованого розрахунку, що актуалізує застосування методів, здатних коректно опрацьовувати неповну й якісну інформацію.

Для впорядкованого оцінювання витрат обсяг робіт проєкту декомпонують за допомогою ієрархічної структури робіт (Work Breakdown Structure, WBS), яка розбиває проєкт на керовані пакети робіт із власною трудомісткістю. Сукупні витрати ІТ-проєкту прийнято поділяти на прямі та непрямі. Прямі витрати безпосередньо пов'язані з розробленням продукту й охоплюють насамперед оплату праці членів проєктної команди, тоді як непрямі, або накладні, витрати включають утримання робочих місць, амортизацію обладнання, ліцензії на програмне забезпечення, адміністративні та комунальні видатки[3]. Оскільки в ІТ-проєктах домінує саме трудова складова, ключовим завданням оцінювання вартості стає достовірне прогнозування трудомісткості, яка згодом перераховується у грошовий вираз через вартість людино-години та коефіцієнт накладних витрат.

					КР.КІ. 10658707.00.00.000.ПЗ	Арк.
						17
Змн.	Арк.	№ докум.	Підпис	Дата		

Регламентацию самого процесса оценивания обеспечивают международные стандарты управления проектами, зокрема зведення знань PMBOK і стандарт ISO 21500, які розглядають оцінювання вартості як окремий процес у складі управління вартістю проекту.

Точність будь-якого методу оцінювання безпосередньо залежить від якості формалізації вимог. Неповні, суперечливі або нестабільні вимоги є основним джерелом похибки оцінок і подальшого перевищення бюджету. Тому сучасна практика передбачає попереднє оброблення вимог – їх структурування, усунення суперечностей і ранжування за пріоритетністю – ще до початку кількісного оцінювання. Проте навіть якісно опрацьовані вимоги зберігають залишкову невизначеність щодо складності реалізації окремих функцій, яку неможливо повністю усунути на ранніх стадіях[4]. Саме ця залишкова якісна невизначеність найкраще піддається опису засобами нечіткої логіки, що й визначає напрям подальшого дослідження.

Таким чином, специфіка ІТ-проектів – нематеріальний характер результату, висока невизначеність вхідних даних, залежність продуктивності від формалізованих і неформальних чинників – вимагає розробки гібридних методів оцінювання вартості, що поєднують об'єктивні кількісні метрики з гнучким якісним описом ризиків. Огляд таких методів подано у підрозділі 1.2.

1.2 Методи оцінки вартості ІТ-проектів

Оцінювання вартості ІТ-проекту – це процес прогнозування трудомісткості й фінансових витрат на його реалізацію, що виконується ще до початку розробки. Основна складність полягає у неповноті вихідних даних на ранніх стадіях життєвого циклу, коли вимоги ще уточнюються, а склад команди й технологічний стек можуть змінюватися. У практиці управління ІТ-проектами сформувалися

					КР.КІ. 10658707.00.00.000.ПЗ	Арк.
						18
Змн.	Арк.	№ докум.	Підпис	Дата		

чотири основні групи методів оцінювання вартості: експертні, аналогові, параметричні та змішані.

Експертні методи спираються на досвід кваліфікованих спеціалістів, які оцінюють трудомісткість окремих робіт на основі знань про схожі проєкти. Найвідомішими формалізаціями експертного підходу є метод Делфі та планувальний покер [29]. Перевагою є гнучкість і застосовність за майже повної відсутності кількісних даних; недоліком – виражена суб'єктивність і залежність якості оцінки від кваліфікації експерта.

Серед експертних методів широкого визнання набув метод PERT, згідно з яким для кожної роботи експерт надає три оцінки трудомісткості – оптимістичну, найімовірнішу та песимістичну, – а підсумкова оцінка обчислюється як їхнє зважене середнє. Такий підхід частково компенсує суб'єктивність, проте не усуває залежності від кваліфікації експерта. У гнучких методологіях розробки поширення набув планувальний покер, у якому члени команди незалежно оцінюють складність задач у відносних одиницях, а розбіжності в оцінках обговорюються до досягнення консенсусу.

Аналогові методи визначають вартість нового проєкту шляхом порівняння з раніше завершеними проєктами, що мають схожі характеристики – тип продукту, технологічний стек, обсяг функціональності. Точність методу прямо залежить від наявності репрезентативної бази історичних даних, що обмежує його застосування для організацій без накопиченої статистики.

Параметричні (алгоритмічні) методи встановлюють математичну залежність між кількісно вимірюваними характеристиками продукту й трудомісткістю його розробки. Найвідомішою є модель COSOMO та її розвиток COSOMO II [2, 3], що пов'язують трудомісткість з прогнозованим обсягом коду в тисячах рядків (KLOC). Спільним обмеженням таких моделей є необхідність прогнозувати обсяг коду на етапі, коли коду ще не існує, що на ранніх стадіях призводить до значної похибки.

Базова модель COSOMO встановлює степеневу залежність трудомісткості від обсягу коду, у якій коефіцієнти залежать від типу проєкту – органічного, напіврозподіленого або вбудованого. Модель COSOMO II уточнює цю залежність,

					КР.КІ. 10658707.00.00.000.ПЗ	Арк.
						19
Змн.	Арк.	№ докум.	Підпис	Дата		

уводячи множники витрат і масштабні чинники, що враховують досвід команди, складність продукту, обмеження платформи та інші параметри. Попри математичну строгість, точність параметричних моделей обмежена необхідністю прогнозувати обсяг коду на етапі, коли архітектура ще не визначена.

Ще однією відомою параметричною моделлю є модель Патнема, відома також під назвою SLIM (Software Lifecycle Management). В її основі лежить програмне рівняння, що пов'язує розмір продукту, сукупні трудовитрати й тривалість розробки через чинник продуктивності та криву розподілу зусиль Рейлі–Нордена. Характерною особливістю моделі є нелінійна залежність між тривалістю проєкту й трудомісткістю: спроба скоротити терміни розробки спричиняє непропорційно велике зростання витрат. Модель Патнема ефективна для великих проєктів із тривалим життєвим циклом, проте, як і інші параметричні методи, потребує надійної оцінки розміру та каліброваних на історичних даних коефіцієнтів продуктивності.

Із розвитком технологій аналізу даних поширення набули методи оцінювання вартості на основі машинного навчання. До них належать регресійні моделі, дерева рішень, штучні нейронні мережі та метод міркувань за прецедентами, які виявляють залежність між характеристиками проєкту й фактичними витратами на основі накопиченої статистики завершених проєктів. За наявності великого й репрезентативного набору історичних даних такі методи здатні забезпечити високу точність прогнозу. Проте їх застосовність обмежена кількома суттєвими чинниками: необхідністю значного обсягу якісних навчальних даних, складністю інтерпретації одержаних моделей та чутливістю до неоднорідності проєктів. Для організацій без накопиченої статистики й для оцінювання на найраніших стадіях ці методи виявляються малопридатними.

Зазначеного недоліку позбавлений метод функціональних точок (Function Point Analysis, FPA), який вимірює розмір продукту за обсягом наданої користувачеві функціональності [16], а не за обсягом коду. Метод незалежний від мови програмування й технології реалізації та застосовний уже на етапі

					КР.КІ. 10658707.00.00.000.ПЗ	Арк.
						20
Змн.	Арк.	№ докум.	Підпис	Дата		

формулювання вимог. Саме ці властивості зумовили його вибір як основи для розроблюваної в роботі моделі оцінювання вартості.

Спільною властивістю всіх класичних детермінованих методів є те, що вони повертають точкову оцінку – одне число, тоді як реальна невизначеність на ранніх стадіях проєкту краще описується діапазоном або якісною шкалою. Це обґрунтовує доцільність поєднання об'єктивного вимірювання розміру методом функціональних точок з апаратом нечіткої логіки [14], здатним опрацьовувати якісні характеристики складності проєкту.

Спільним недоліком як класичних, так і інтелектуальних методів залишається складність опрацювання якісної експертної інформації про складність проєкту, яка зазвичай формулюється у природномовних термінах – «низька», «середня», «висока». Класичні методи потребують переведення таких оцінок у точні числа, що спричиняє втрату інформації, а методи машинного навчання – попереднього навчання на великих вибірках. Апарат нечіткої логіки дає змогу безпосередньо опрацьовувати лінгвістичні оцінки, зберігаючи їхню якісну природу, що робить його природним доповненням до об'єктивного вимірювання розміру методом функціональних точок.

Таблиця 1.1 – Порівняльна характеристика методів оцінювання вартості ІТ-проєктів

Група методів	Потреба в даних	Рання застосовність	Об'єктивність	Урахування невизначеності
Експертні	Низька	Висока	Низька	Часткове
Аналогові	Висока	Середня	Середня	Низьке
Параметричні (СОСОМО)	Висока	Низька	Висока	Низьке
Функціональні точки	Середня	Висока	Висока	Низьке

Продовження таблиці 1.1

Гібридні (нечіткі)	Середня	Висока	Висока	Високе
-----------------------	---------	--------	--------	--------

Наведене порівняння показує, що метод функціональних точок поєднує об'єктивність і застосовність на ранніх стадіях, проте, як і інші детерміновані методи, повертає точкову оцінку без урахування невизначеності. Натомість гібридний підхід, що доповнює функціональні точки нечіткою логікою, здатний водночас забезпечити об'єктивність вимірювання розміру й гнучке врахування якісної невизначеності складності, що обґрунтовує його вибір для цієї роботи.

1.3 Засоби моделювання

Для програмної реалізації нечіткої моделі потрібне середовище математичного моделювання, що підтримує побудову систем нечіткого виводу та візуалізацію результатів. Розглянуто три типові середовища: інженерне середовище MATLAB разом із модулем Fuzzy Logic Toolbox, вільне середовище GNU Octave та бібліотеки мови Python (scikit-fuzzy), а також хмарну навчально-дослідницьку платформу Simulations4All. Порівняльну характеристику середовищ наведено в таблиці 1.2.

Таблиця 1.2 – Порівняльна характеристика середовищ нечіткого моделювання

Критерій	Simulations4All	MATLAB + Fuzzy Toolbox	Python (scikit- fuzzy)
Ліцензія	Безкоштовна, академічна	Платна (комерційна)	Відкрита (BSD)
Поріг входження	Низький	Середній	Високий

Продовження таблиці 1.2

Підтримка виводу Мамдані	Так	Так	Так
Візуалізація результатів	Вбудована	Вбудована	Через matplotlib
Встановлення	Не потрібне (хмара)	Локальне	Локальне

MATLAB – комерційне інженерне середовище математичних і наукових обчислень, що є фактичним галузевим стандартом. Спеціалізований модуль Fuzzy Logic Toolbox надає високорівневі функції побудови систем нечіткого виводу Мамдані з графічним редактором. Основним обмеженням є платна ліцензія, що ускладнює використання в навчальному процесі.

Графічний інструментарій Fuzzy Logic Toolbox містить редактор функцій належності, редактор бази правил та засоби перегляду поверхні виводу в реальному часі, що робить MATLAB зручним для навчання й прототипування. Проте висока вартість комерційної ліцензії робить його малодоступним для індивідуального навчального використання. Середовище GNU Octave відтворює значну частину синтаксису MATLAB і є безкоштовним, однак нечітка логіка в ньому реалізується стороннім пакетом з обмеженою функціональністю, який нерегулярно оновлюється.

Екосистема мови Python пропонує бібліотеки scikit-fuzzy та simplful, які надають програмний інтерфейс для побудови нечітких систем і добре інтегруються із засобами аналізу даних. Такий підхід забезпечує максимальну гнучкість і відтворюваність обчислень, проте потребує впевненого володіння програмуванням і не має графічного редактора, що ускладнює швидке прототипування для користувачів-початківців. Таким чином, кожне із загальнодоступних середовищ має певні обмеження для навчально-дослідницького застосування.

GNU Octave – вільне середовище зі синтаксисом, наближеним до MATLAB. Воно підтримує імпорт m-файлів, проте сторонній модуль нечіткої логіки має

					КР.КІ. 10658707.00.00.000.ПЗ	Арк.
						23
Змн.	Арк.	№ докум.	Підпис	Дата		

обмежену функціональність і не завжди сумісний з останніми версіями. Бібліотеки мови Python (scikit-fuzzy) надають максимальну гнучкість і добре інтегруються з екосистемою аналізу даних, однак потребують вищого рівня програмістської підготовки.

Хмарну платформу Simulations4All обрано середовищем реалізації за сукупністю критеріїв: повна безкоштовність для навчального використання, відсутність потреби встановлення локального програмного забезпечення, кросплатформність (робота безпосередньо у браузері) та вбудовані засоби візуалізації. Мова опису моделей SimLab є предметно-орієнтованою мовою процедурного типу, синтаксис якої наближено до MATLAB та Octave. Платформа не містить готового модуля нечіткого виводу, тому всі етапи алгоритму Мамдані реалізовано вручну, що додатково поглиблює розуміння його роботи. Платформа підтримує побудову двовимірних графіків функцій, стовпчастих діаграм і тривимірних поверхонь, що достатньо для візуалізації функцій належності, поверхні виводу та результатів аналізу чутливості моделі.

Архітектурно платформа Simulations4All побудована за клієнт-серверною моделлю: клієнтська частина є однострінковим вебзастосунком, що виконується у браузері, а обчислення здійснюються на сервері. Робочий простір організовано за трипанельним принципом – редактор коду мовою SimLab, область виведення результатів і графіків та довідкова панель. Відсутність спеціалізованого модуля нечіткого виводу для навчально-дослідницьких цілей має й позитивний бік: необхідність самостійно реалізувати всі етапи алгоритму Мамдані забезпечує глибше розуміння математичного апарату, ніж застосування готових функцій.

Для обґрунтованого вибору середовища реалізації сформульовано систему критеріїв порівняння: вартість і умови ліцензування, доступність для індивідуального навчального використання, наявність вбудованих засобів нечіткого виводу, якість і достатність інструментів візуалізації, кросплатформність, а також придатність середовища для глибокого засвоєння математичного апарату. За критерієм вартості перевагу мають безкоштовні рішення – GNU Octave, бібліотеки Python та Simulations4All; за критерієм зручності

					КР.КІ. 10658707.00.00.000.ПЗ	Арк.
						24
Змн.	Арк.	№ докум.	Підпис	Дата		

прототипування лідирує MATLAB завдяки графічному редактору; за критерієм переносності коду найкращі показники забезпечує процедурна реалізація, що не спирається на специфічні бібліотеки. Зіставлення середовищ за цими критеріями, наведене в таблиці 1.2, показало, що жодне з них не є оптимальним за всіма параметрами одночасно, тому остаточний вибір зроблено на основі пріоритетності критеріїв саме для навчально-дослідницької задачі.

Мова SimLab, якою реалізовано модель, належить до процедурних предметно-орієнтованих мов і підтримує основні типи даних, потрібні для нечіткого моделювання: скалярні величини, вектори та матриці. Її синтаксис передбачає оголошення змінних, арифметичні й логічні операції, умовні конструкції та цикли, а також визначення власних функцій, що дає змогу оформити кожен етап алгоритму Мамдані у вигляді окремої функції. Функції належності лінгвістичних змінних описуються як параметричні залежності, а їхні параметри й база правил зберігаються у вигляді таблиць, винесених окремо від обчислювального коду. Такий поділ на дані й алгоритм спрощує налаштування моделі та її подальше супроводження.

Важливою перевагою обраного підходу з ручною реалізацією алгоритму Мамдані є незалежність моделі від конкретного середовища виконання. Оскільки алгоритмічне ядро використовує лише базові операції – мінімум, максимум, арифметичні дії та чисельне інтегрування для дефазифікації, – воно не потребує жодних спеціалізованих бібліотек нечіткого виводу й без структурних змін відтворюється у MATLAB, GNU Octave чи Python. Натомість рішення, побудовані на основі готових модулів на кшталт Fuzzy Logic Toolbox, виявляються прив'язаними до конкретного комерційного середовища. Таким чином, обране поєднання платформи Simulations4All із процедурною реалізацією забезпечує водночас безкоштовність, переносність і прозорість обчислень, що повністю відповідає вимогам навчально-дослідницької роботи.

					КР.КІ. 10658707.00.00.000.ПЗ	Арк.
						25
Змн.	Арк.	№ докум.	Підпис	Дата		

1.4 Постановка задачі кваліфікаційної роботи

На основі проведеного аналізу методів і засобів оцінювання вартості для розв'язання поставленої задачі обрано метод функціональних точок IFPUG як спосіб об'єктивного вимірювання розміру програмного забезпечення. Вибір методу зумовлений тим, що він застосовний на ранніх стадіях проєкту за переліком функціональних вимог, не залежить від мови програмування й технології реалізації та має добру кореляцію з реальною трудомісткістю розробки.

Для опрацювання коефіцієнта складності проєкту – якісної характеристики, яка погано піддається точному вимірюванню, – обрано апарат нечіткої логіки з виводом за алгоритмом Мамдані. Поєднання двох підходів дає змогу одночасно враховувати об'єктивний функціональний розмір продукту й нечіткі експертні оцінки складності, формуючи обґрунтовану оцінку вартості за умов неповноти вихідних даних.

Середовищем програмної реалізації обрано хмарну платформу Simulations4All. Найпоширеніший інструмент нечіткого моделювання – пакет Fuzzy Logic Toolbox середовища MATLAB – вимагає платної ліцензії та реєстрації, що унеможливило його використання в навчальних умовах. Тому перевагу віддано платформі Simulations4All, яка є безкоштовною для навчального використання, не потребує встановлення локального програмного забезпечення (працює у браузері), підтримує кросплатформність і містить вбудовані засоби візуалізації, достатні для побудови й дослідження нечіткої моделі. Це й зумовило остаточний вибір середовища реалізації.

Сутність поставленої задачі полягає в тому, щоб за відомими кількостями функціональних компонентів програмного продукту та якісною оцінкою складності проєкту сформулювати числову оцінку вартості його розробки, придатну для попереднього планування бюджету. Особливість задачі зумовлена різноманітністю вхідних даних: розмір продукту вимірюється об'єктивно у функціональних точках, тоді як складність є якісною характеристикою, яку експерт

					КР.КІ. 10658707.00.00.000.ПЗ	Арк.
						26
Змн.	Арк.	№ докум.	Підпис	Дата		

оцінює наближено. Через це задача не зводиться до прямого застосування детермінованої формули й потребує апарату, здатного коректно опрацьовувати неповну та неточну інформацію.

Доцільність обраного гібридного підходу обґрунтовується взаємодоповнюваністю двох методів. Метод функціональних точок надає стандартизовану, незалежну від технології метрику обсягу функціональності, що усуває суб'єктивність в оцінюванні розміру. Нечіткий вивід за алгоритмом Мамдані, своєю чергою, дає змогу оперувати лінгвістичними оцінками складності («низька», «середня», «висока») і плавно відображати їх у вартість через базу правил «якщо–то». Поєднання цих методів забезпечує водночас об'єктивність вимірювання розміру й гнучкість урахування якісної невизначеності, недосяжну для жодного з методів окремо.

Метою кваліфікаційної роботи є підвищення обґрунтованості оцінювання вартості ІТ-проектів на ранніх стадіях шляхом розроблення нечіткої моделі, що поєднує метод функціональних точок із нечітким виводом за алгоритмом Мамдані.

Для досягнення поставленої мети необхідно розв'язати такі задачі:

- дослідити метод функціональних точок і сформулювати алгоритм обчислення функціонального розміру програмного забезпечення;
- обґрунтувати алгоритм побудови нечіткої моделі за Мамдані та розробити гібридний нечіткий алгоритм оцінювання вартості (розділ 2);
- розробити структуру програмного модуля, побудувати функції належності лінгвістичних змінних і базу з п'ятнадцяти правил нечіткого виводу та реалізувати модель у середовищі Simulations4All;
- провести експериментальні дослідження розробленої моделі — обчислювальні експерименти та аналіз чутливості (розділ 3).

					КР.КІ. 10658707.00.00.000.ПЗ	Арк.
						27
Змн.	Арк.	№ докум.	Підпис	Дата		

2 АЛГОРИТМ ОЦІНКИ ВАРТОСТІ ЗА ФУНКЦІОНАЛЬНИМИ ТОЧКАМИ

2.1 Метод функціональних точок

Метод функціональних точок (Function Point Analysis, FPA) запропонував А. Албрехт, співробітник компанії IBM [16], як універсальний спосіб вимірювання розміру програмного забезпечення, незалежний від мови програмування й технології реалізації. Згодом метод стандартизовано міжнародною організацією користувачів функціональних точок IFPUG [1, 8], керівництво якої й нині визначає правила підрахунку.

Базова ідея методу – вимірювати функціональний розмір системи з погляду користувача. Будь-який програмний продукт надає користувачам певний набір функціональних можливостей: введення даних, одержання звітів, виконання запитів, обмін інформацією з іншими системами. Метод виокремлює п'ять типів функціональних компонентів, кожен з яких відображає певний аспект взаємодії системи з користувачем або зовнішнім середовищем.

Процедура підрахунку функціональних точок виконується у кілька кроків. Спочатку для розроблюваної системи визначають межі застосування й перелік функціональних компонентів кожного з п'яти типів. Потім кожному компоненту присвоюють рівень складності за критеріями кількості елементів даних і кількості пов'язаних файлів: зовнішній вхід вважають простим, якщо він звертається до одного-двох файлів і містить незначну кількість полів, і складним – за більшої кількості файлів або полів. Помноживши кількість компонентів кожного типу й рівня на відповідний ваговий коефіцієнт і підсумувавши результати, одержують некориговані функціональні точки.

Зовнішні входи (External Inputs, EI) – операції, через які дані вводяться до системи: форми реєстрації та редагування записів, форми завантаження файлів. Зовнішні виходи (External Outputs, EO) – операції формування звітів, повідомлень та експортних файлів, що передбачають обробку даних. Зовнішні запити (External

					КР.КІ. 10658707.00.00.000.ПЗ	Арк.
						28
Змн.	Арк.	№ докум.	Підпис	Дата		

Inquiries, EQ) – операції одержання інформації без її модифікації: перегляд деталей запису, пошук за критеріями.

Внутрішні логічні файли (Internal Logical Files, ILF) – групи логічно пов'язаних даних, що зберігаються в самій системі й управляються нею; у реляційних базах даних кожен ILF зазвичай відповідає таблиці або групі пов'язаних таблиць. Зовнішні інтерфейсні файли (External Interface Files, EIF) – групи даних, якими система обмінюється з іншими інформаційними системами, але які управляються поза нею.

Кожен функціональний компонент відносять до одного з трьох рівнів складності – низького, середнього або високого – за критеріями офіційного керівництва IFPUG. Кожному рівню складності кожного типу компонента присвоєно ваговий коефіцієнт, що відображає його внесок у загальну трудомісткість реалізації. Стандартні значення вагових коефіцієнтів IFPUG [8] наведено в таблиці 2.1.

Таблиця 2.1 – Вагові коефіцієнти функціональних компонентів за IFPUG

Тип функціонального компонента	Низька	Середня	Висока
Зовнішні входи (EI)	3	4	6
Зовнішні виходи (EO)	4	5	7
Зовнішні запити (EQ)	3	4	6
Внутрішні логічні файли (ILF)	7	10	15
Зовнішні інтерфейсні файли (EIF)	5	7	10

Некориговані функціональні точки (Unadjusted Function Points, UFP) обчислюються як сума добутків кількостей функціональних компонентів на відповідні вагові коефіцієнти. Для врахування загальносистемних характеристик, які впливають на трудомісткість, але не описуються переліком компонентів, у методі введено коефіцієнт коригування VAF (Value Adjustment Factor), що

обчислюється на основі чотирнадцяти загальних характеристик системи. Скориговані функціональні точки (Adjusted Function Points, AFP) визначаються як добуток UFP та VAF за формулою (2.1):

До чотирнадцяти загальних характеристик системи належать інтенсивність обміну даними, розподіленість обробки, вимоги до продуктивності, складність обробки, повторне використання, зручність установлення та експлуатації, багатоплатформність та інші. Кожній характеристиці експерт присвоює ступінь впливу від нуля (відсутній) до п'яти (сильний). Сума цих ступенів утворює загальний ступінь впливу, за яким коефіцієнт коригування обчислюється у межах від 0,65 до 1,35, тобто скоригований розмір може відрізнятися від некоригованого не більш ніж на 35 % в обидва боки.

$$AFP = UFP \cdot VAF \quad (2.1)$$

Перевагами методу функціональних точок є незалежність від мови програмування й технологічного стеку, застосовність на ранніх стадіях проєкту та добра кореляція з реальною трудомісткістю. Обмеженням є певна суб'єктивність віднесення компонентів до рівнів складності й оцінювання загальних характеристик системи. Саме природна суб'єктивність цих оцінок робить доцільним поєднання методу функціональних точок з апаратом нечіткої логіки, у якому якісні характеристики опрацьовуються лінгвістичними змінними.

Для переходу від функціональних точок до трудомісткості застосовують показник продуктивності команди – середню кількість функціональних точок, що реалізується одним розробником за людину-місяць. За даними галузевих досліджень, цей показник коливається від 8 до 20 функціональних точок на людину-місяць залежно від технологічного стеку, кваліфікації команди та складності предметної області. Помноживши трудомісткість у людину-місяцях на вартість людину-місяця, одержують орієнтовну вартість розробки. Саме на цьому етапі детерміновані методи виявляють основну ваду – необхідність задавати єдине

					КР.КІ. 10658707.00.00.000.ПЗ	Арк.
						30
Змн.	Арк.	№ докум.	Підпис	Дата		

точкове значення продуктивності, тоді як насправді воно є невизначеним і залежить від низки якісних чинників.

2.2 Алгоритм побудови нечітких моделей

Нечітка логіка – це розширення класичної двозначної логіки на випадок, коли належність об'єкта до множини визначається не однозначно (істина або хиба, 1 або 0), а певним числом у діапазоні $[0; 1]$, що називається ступенем належності. Цей апарат запропонував Л. Заде [11] для формалізації нечітких, виражених природною мовою понять.

Базовим поняттям апарату є нечітка множина, яка задається функцією належності $\mu(x)$, що приймає значення у відрізьку $[0; 1]$. Тісно пов'язане поняття – лінгвістична змінна, значеннями якої є не числа, а слова або словосполучення природної мови, які називають термами. Наприклад, лінгвістична змінна «складність проєкту» може приймати значення «низька», «середня», «висока»; кожному терму відповідає окрема функція належності.

Найпоширеніші форми функцій належності – трикутні, трапецієподібні й гаусові [4, 23]. Над нечіткими множинами визначено операції, що узагальнюють класичні операції теорії множин; зокрема, перетин нечітких множин A і B (відповідник логічного «І») у класичній моделі задається мінімумом їхніх функцій належності за формулою (2.2):

Окрім перетину, над нечіткими множинами визначено операцію об'єднання (відповідник логічного «АБО»), що задається максимумом функцій належності, та операцію доповнення (відповідник заперечення), що обчислюється як одиниця мінус функція належності. Ці операції є основою механізму нечіткого виводу: перетин застосовується для обчислення ступеня виконання правила з кількома передумовами, а об'єднання – для агрегування результатів кількох правил у єдину вихідну множину.

					КР.КІ. 10658707.00.00.000.ПЗ	Арк.
						31
Змн.	Арк.	№ докум.	Підпис	Дата		

Трапецієподібна функція належності задається чотирма параметрами й має пласку вершину, що зручно для опису крайніх термів лінгвістичної змінної. Гаусова функція задається центром і шириною та забезпечує гладку, нескінченно диференційовну характеристику, проте складніша в обчисленні. У цій роботі для крайніх термів обрано трапецієподібні функції, а для проміжних – трикутні, що забезпечує простоту обчислень за достатньої гнучкості опису.

$$\mu_{A \cap B}(x) = \min(\mu_A(x), \mu_B(x)) \quad (2.2)$$

Власне нечітка модель – це сукупність нечітких множин для вхідних і вихідних лінгвістичних змінних та бази правил «якщо–то», що описують залежність вихідних змінних від вхідних. Процес обчислення значення вихідної змінної за відомими значеннями вхідних називається нечітким виводом. Для задачі оцінювання вартості ІТ-проектів обрано алгоритм виводу Мамдані [5], оскільки він найбільш інтуїтивно інтерпретується експертами, а результат у вигляді нечіткої множини додатково ілюструє ступінь невизначеності оцінки.

Поділ кожної вхідної змінної на три терми обрано як компроміс між точністю опису та кількістю правил: збільшення кількості термів підвищило б деталізацію, але призвело б до квадратичного зростання кількості правил і ускладнило б їх експертне обґрунтування. За трьох термів кожної з двох вхідних змінних базова кількість правил становить дев'ять; за потреби її доповнюють правилами для особливих випадків, зокрема неповних вхідних даних, як це зроблено в розділі 3.

Алгоритм Мамдані складається з чотирьох послідовних етапів, узагальнену блок-схему яких наведено на рисунку 2.1.

Перший етап – фазифікація: для заданих числових значень вхідних змінних (у цій роботі – розміру програмного забезпечення у функціональних точках і коефіцієнта складності проекту) обчислюються ступені належності до кожного терму відповідної змінної. Другий етап – активація правил: для кожного правила бази ступінь його виконання визначається як мінімум ступенів належності передумов. Третій етап – імплікація та агрегування: функцію належності вихідного

					КР.КІ. 10658707.00.00.000.ПЗ	Арк.
						32
Змн.	Арк.	№ докум.	Підпис	Дата		

терму кожного активованого правила обмежують зверху рівнем його виконання, після чого одержані функції об'єднують операцією максимуму в одну результуючу нечітку множину вартості.

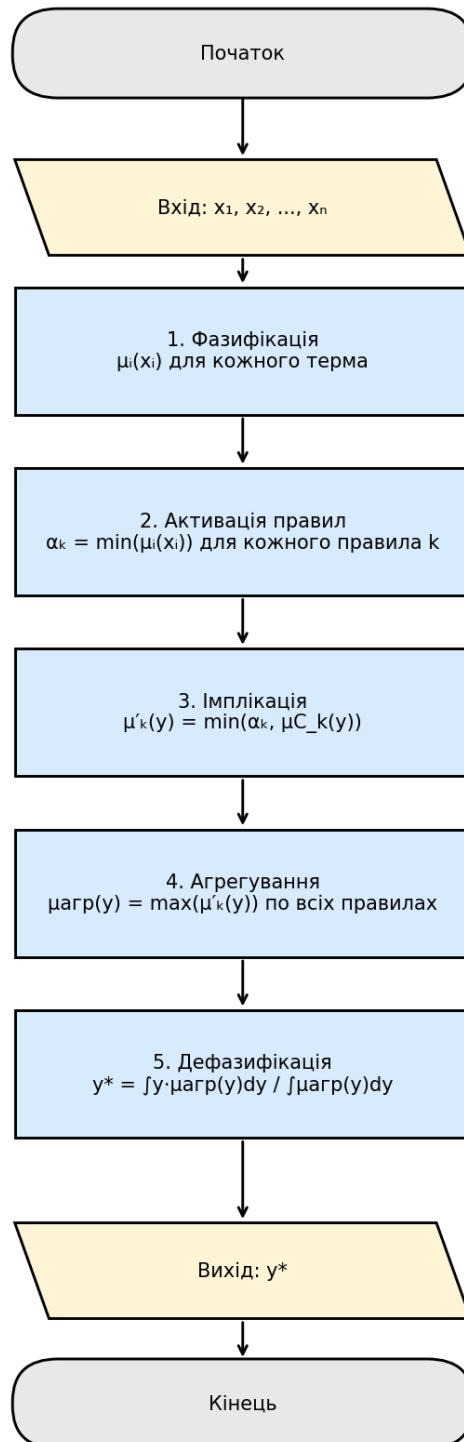


Рисунок 2.1 – Блок-схема алгоритму нечіткого виводу за Мамдані

Четвертий етап – дефазифікація: результуюча нечітка множина перетворюється на одне числове значення вартості методом центру ваги. Саме цю послідовність етапів реалізовано у програмному модулі, описаному в третьому розділі.

2.3 Нечіткий алгоритм оцінки вартості ІТ-проектів

Поєднання методу функціональних точок із нечітким виводом за алгоритмом Мамдані утворює гібридний алгоритм оцінювання вартості ІТ-проектів [14, 27]. Функціональні точки забезпечують об'єктивне, стандартизоване вимірювання розміру програмного забезпечення на ранніх стадіях, а нечітка логіка дає змогу гнучко враховувати якісну характеристику складності проекту, яка погано піддається точному кількісному вимірюванню.

Узагальнений гібридний алгоритм складається з послідовності етапів: обчислення некоригованих функціональних точок UFP за методом IFPUG; задання експертного коефіцієнта складності проекту; фазифікація обох вхідних змінних; нечіткий вивід вартості за базою правил «якщо–то» за алгоритмом Мамдані; дефазифікація результату й формування підсумкової числової оцінки вартості. Блок-схему гібридного алгоритму наведено на рисунку 2.2.

Розглянемо узагальнений гібридний алгоритм оцінювання вартості детальніше. На першому кроці на основі переліку функціональних вимог визначають кількості п'яти типів функціональних компонентів і обчислюють некориговані функціональні точки. На другому кроці експерт задає коефіцієнт складності проекту в діапазоні $[0; 1]$ на основі якісної оцінки архітектурної та технологічної складності. На третьому кроці виконують фазифікацію обох вхідних змінних, на четвертому – активацію правил та імплікацію, на п'ятому – агрегування, на шостому – дефазифікацію результату методом центру ваги. У вигляді псевдокоду основний цикл алгоритму подано нижче:

					КР.КІ. 10658707.00.00.000.ПЗ	Арк.
						34
Змн.	Арк.	№ докум.	Підпис	Дата		

$UFP \leftarrow \sum (fp[i] \cdot w[i])$ // обчислення розміру
 для кожного правила r з бази:
 $\alpha[r] \leftarrow \min(\mu_{\text{розмір}}(UFP), \mu_{\text{складність}}(c))$ // ступінь виконання
 $B'[r] \leftarrow \min(\alpha[r], B_{\text{вих}}[r])$ // імплікація (відсікання)
 $Aggr \leftarrow \max(B'[r])$ за всіма r // агрегування
 $\text{Вартість} \leftarrow \sum (x \cdot Aggr(x)) / \sum Aggr(x)$ // дефазифікація

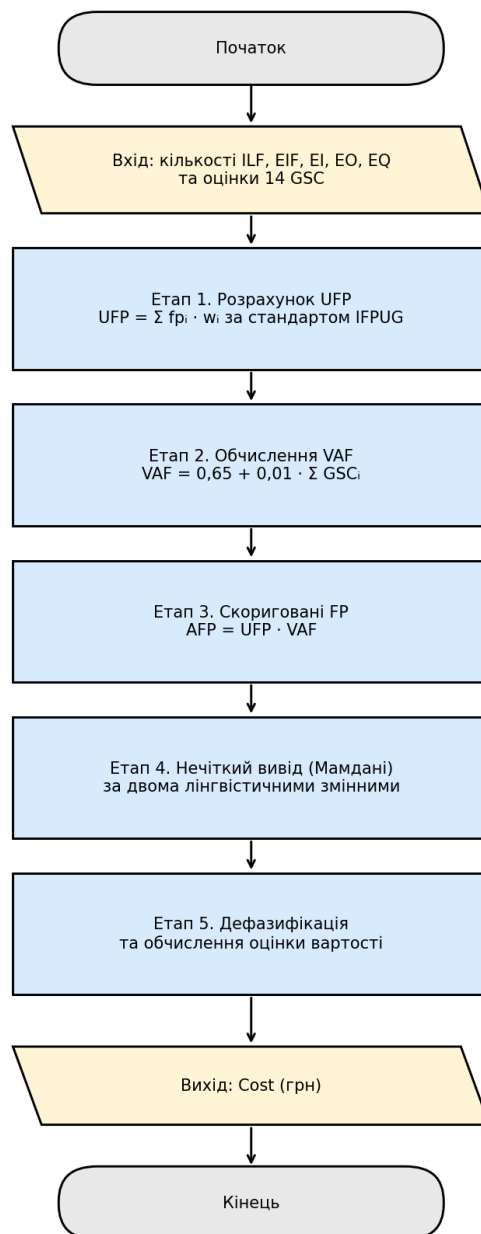


Рисунок 2.2 – Блок-схема нечіткого алгоритму оцінки вартості ІТ-проектів на основі функціональних точок

На вхід гібридний алгоритм отримує кількості п'яти типів функціональних елементів та експертний коефіцієнт складності. Блок обчислення розміру визначає функціональний розмір UFP, блоки фазифікації перетворюють розмір і складність на ступені належності до термів, блок нечіткого виводу за базою правил формує нечітку оцінку вартості, а блок дефазифікації перетворює її на підсумкове число. Програмну реалізацію цього алгоритму описано в третьому розділі.

Вибір методу центру ваги для дефазифікації зумовлений тим, що він враховує всю форму результуючої нечіткої множини, а не лише її окремі точки, і забезпечує плавну, неперервну залежність вихідного значення від входів. Альтернативні методи дефазифікації (метод максимуму, метод центру максимумів) є простішими в обчисленні, проте можуть давати розривні характеристики, що небажано для задачі оцінювання вартості, де очікується монотонна й плавна реакція моделі на зміну вхідних параметрів.

Досліджено метод функціональних точок IFPUG як спосіб об'єктивного вимірювання розміру програмного продукту та розглянуто апарат нечіткої логіки з алгоритмом нечіткого виводу Мамдані. Сформовано гібридний алгоритм оцінювання вартості IT-проектів, що поєднує кількісне вимірювання розміру з нечітким опрацюванням якісної складності. Побудовані блок-схеми алгоритму Мамдані та гібридного алгоритму є основою для програмної реалізації нечіткого модуля в третьому розділі.

					КР.КІ. 10658707.00.00.000.ПЗ	Арк.
						36
Змн.	Арк.	№ докум.	Підпис	Дата		

3 НЕЧІТКИЙ МОДУЛЬ ОЦІНКИ ВАРТОСТІ ІТ-ПРОЄКТІВ

3.1 Формалізація вхідних і вихідних даних

Програмний модуль реалізує функцію перетворення кількісних і якісних характеристик ІТ-проєкту в орієнтовну оцінку вартості його розробки. Вхідними даними є кількості функціональних елементів за методом функціональних точок IFPUG та коефіцієнт складності проєкту; вихідним результатом є прогнозована вартість розробки у тисячах гривень. Структурно модуль складається з чотирьох послідовних блоків: блоку обчислення функціонального розміру за методом функціональних точок, блоку фазифікації вхідних змінних, блоку нечіткого виводу за алгоритмом Мамдані та блоку дефазифікації, що формує підсумкову числову оцінку вартості.

Модель оперує двома вхідними лінгвістичними змінними – розміром програмного забезпечення та складністю проєкту – і однією вихідною змінною – вартістю розробки. Розмір вимірюється у функціональних точках і обчислюється з кількостей п'яти типів функціональних компонентів (EI, EO, EQ, ILF, EIF) за формулою (3.1), тобто є проміжною величиною, що подається на перший вхід; складність задає експерт у діапазоні $[0; 1]$. Діапазони, терми та форми функцій належності змінних наведено в таблиці 3.1.

Таблиця 3.1 – Лінгвістичні змінні нечіткої моделі

Змінна	Роль	Діапазон	Терми (значення)
Розмір ПЗ (UFP)	вхідна	0...600 ф.т.	мала, середня, велика
Складність (с)	вхідна	0...1	низька, середня, висока
Вартість	вихідна	0...3000 тис. грн	дуже низька, низька, середня, висока, дуже висока

Для крайніх термів кожної лінгвістичної змінної застосовано трапецієподібні функції належності, а для проміжних – трикутні.

Таким чином, на вхід моделі подаються лише дві величини – розмір ПЗ у функціональних точках і коефіцієнт складності, – а на виході формується прогнозована вартість розробки у тисячах гривень разом з її лінгвістичною інтерпретацією. Функціональні компоненти EI, EO, EQ, ILF, EIF та вагові коефіцієнти w_i є проміжними величинами, що використовуються лише для обчислення розміру й не подаються на вхід нечіткого виводу безпосередньо.

Блок обчислення розміру приймає кількості функціональних компонентів і повертає некориговані функціональні точки. Блок фазифікації приймає числові значення розміру й складності та повертає вектори ступенів належності до термів вхідних змінних. Блок нечіткого виводу застосовує базу правил і повертає агреговану нечітку множину вартості, яку блок дефазифікації перетворює на підсумкове числове значення.

Перший блок приймає на вхід кількості п'яти типів функціональних елементів (зовнішніх входів EI, зовнішніх виходів EO, зовнішніх запитів EQ, внутрішніх логічних файлів ILF та зовнішніх інтерфейсних файлів EIF) і обчислює некориговані функціональні точки UFP як зважену суму цих кількостей за формулою (3.1):

$$UFP = \sum_i f p_i \cdot w_i \quad (3.1)$$

де $f p_i$ – кількість елементів i -го типу;

w_i – відповідний ваговий коефіцієнт IFPUG.

Для демонстрації роботи модуля обрано тестовий проєкт корпоративної інформаційної системи управління ресурсами підприємства (ERP-модуль середнього рівня) з такими характеристиками: EI = 15, EO = 10, EQ = 8, ILF = 5, EIF = 3. Із застосуванням стандартних середніх ваг IFPUG $w = (4, 5, 4, 10, 7)$ функціональний розмір продукту становить $UFP = 15 \cdot 4 + 10 \cdot 5 + 8 \cdot 4 + 5 \cdot 10 + 3 \cdot 7 = 213$ функціональних точок.

					КР.КІ. 10658707.00.00.000.ПЗ	Арк.
						38
Змн.	Арк.	№ докум.	Підпис	Дата		

Другим вхідним параметром модуля є коефіцієнт складності проєкту complexity у діапазоні $[0; 1]$, який інтегрально характеризує архітектурну та технологічну складність системи: кількість зовнішніх інтеграцій, новизну застосованих технологій, вимоги до надійності й продуктивності. На відміну від розміру, цей параметр має виражено якісну природу й погано піддається точному вимірюванню, тому його доцільно опрацьовувати засобами нечіткої логіки. Для тестового проєкту експертно прийнято значення складності $complexity = 0,6$.

Модуль забезпечує такі можливості: обчислення функціонального розміру за методом IFPUG; фазифікацію вхідних змінних за заданими функціями належності; нечіткий вивід вартості за базою правил «якщо–то» за алгоритмом Мамдані; дефазифікацію результату методом центру ваги; візуалізацію функцій належності, поверхні виводу та чутливості моделі. На виході модуль повертає одне числове значення – прогнозовану вартість розробки у тисячах гривень, яке використовується для попереднього планування бюджету проєкту на ранніх стадіях.

Вхідні дані модуля організовано у вигляді двох структур: перша містить п'ять цілочисельних полів – кількості зовнішніх входів, виходів, запитів, внутрішніх логічних та зовнішніх інтерфейсних файлів, друга містить дробове значення коефіцієнта складності. Функції належності всіх лінгвістичних змінних задано таблицями параметрів, що дає змогу змінювати конфігурацію моделі без модифікації її алгоритмічного ядра. Перед обчисленням модуль перевіряє коректність вхідних даних: кількості компонентів мають бути невід'ємними цілими числами, а коефіцієнт складності – належати відрізку $[0; 1]$; у разі виходу значень за межі модуль повідомляє про помилку й не виконує обчислення.

Взаємодію блоків модуля організовано за принципом послідовного конвеєра, у якому вихід кожного блоку є входом наступного. Блок обчислення розміру передає до блоку фазифікації єдине число – функціональний розмір UFP, а коефіцієнт складності надходить до фазифікації безпосередньо із вхідних даних. Блок фазифікації формує два вектори ступенів належності – для розміру та для складності – і передає їх до блоку нечіткого виводу. Блок виводу обчислює ступені

					КР.КІ. 10658707.00.00.000.ПЗ	Арк.
						39
Змн.	Арк.	№ докум.	Підпис	Дата		

активації всіх правил бази, виконує імплікацію та агрегує часткові результати в єдину нечітку множину вартості, подану дискретними значеннями функції належності на універсальній множині. Нарешті, блок дефазифікації перетворює цю множину на одне числове значення вартості. Така конвеєрна організація забезпечує модульність: кожен блок можна перевірити й змінити незалежно від інших.

Програмну реалізацію модуля виконано у хмарному середовищі моделювання Simulations4All засобами вбудованої мови опису обчислень. Алгоритмічне ядро організовано у вигляді окремих функцій, що відповідають блокам структурної схеми, завдяки чому логіку нечіткого виводу відокремлено від конфігураційних даних. Параметри всіх функцій належності та базу правил винесено в окремі таблиці, тому зміна термів лінгвістичних змінних або вагових коефіцієнтів методу функціональних точок не потребує втручання в обчислювальний код. Свідома відмова від використання спеціалізованих бібліотек нечіткого виводу робить реалізацію самодостатньою та переносною: алгоритмічне ядро без структурних змін відтворюється в інших середовищах математичного моделювання, зокрема в MATLAB та GNU Octave.

Вихідні дані модуля подаються у двох взаємодоповнювальних формах. Основним результатом є числова оцінка вартості у тисячах гривень, придатна для безпосереднього використання в кошторисних розрахунках. Додатково результат інтерпретується в термінах лінгвістичної шкали вартості шляхом визначення терму, до якого здобуте значення належить із найбільшим ступенем. Така подвійна форма подання дає менеджерів проєкту як точкову оцінку для планування бюджету, так і якісну характеристику для швидкого порівняння варіантів. Поряд із числовим результатом модуль формує графічні залежності – функції належності змінних, поверхню нечіткого виводу та криву чутливості вартості до зміни коефіцієнта складності, – які використовуються для аналізу поведінки моделі в наступних підрозділах.

Формально розроблювану модель можна подати як відображення вхідної пари «розмір–складність» у значення вартості. Вхідними змінними є функціональний розмір продукту у функціональних точках, обчислений за

					КР.КІ. 10658707.00.00.000.ПЗ	Арк.
						40
Змн.	Арк.	№ докум.	Підпис	Дата		

методом IFPUG, і коефіцієнт складності проєкту в діапазоні $[0; 1]$; вихідною змінною є орієнтовна вартість розробки. Саме відображення реалізується через базу нечітких правил і процедуру нечіткого виводу.

На вхідні змінні накладено обмеження: розмір є невід'ємною величиною, а коефіцієнт складності належить відрізку $[0; 1]$; вихідна величина обмежена згори максимальним значенням універсальної множини вартості, визначеним з огляду на масштаб розглядуваних проєктів.

До функціональних вимог розроблюваного модуля віднесено: обчислення функціонального розміру програмного продукту за методом IFPUG із конфігурованими ваговими коефіцієнтами; задання коефіцієнта складності проєкту як другої вхідної змінної; реалізацію повного циклу нечіткого виводу за алгоритмом Мамдані з фазифікацією, активацією правил, агрегуванням і дефазифікацією; формування числової оцінки вартості з можливістю інтерпретації результату в термінах лінгвістичної шкали; візуалізацію функцій належності, поверхні виводу та залежності вартості від вхідних параметрів.

Нефункціональні вимоги охоплюють такі характеристики: час виконання одного циклу обчислення вартості не повинен перевищувати однієї секунди на типовому персональному комп'ютері; модель має бути реалізованою у безкоштовному кросплатформному середовищі; алгоритмічне ядро повинно припускати перенесення в інші середовища нечіткого моделювання без структурних змін; взаємодія з моделлю має забезпечувати інтуїтивно зрозуміле введення вхідних параметрів і наочне подання результатів.

3.2 Нечітка модель та база правил

Нечітку модель реалізовано у середовищі моделювання Simulations4All; повний текст програмного коду моделі наведено в Додатку Б. Модель має дві вхідні лінгвістичні змінні та одну вихідну. Вхідна змінна «розмір ПЗ» з областю

					КР.КІ. 10658707.00.00.000.ПЗ	Арк.
						41
Змн.	Арк.	№ докум.	Підпис	Дата		

визначення 0...600 функціональних точок описується трьома термами – «мала», «середня» та «велика». Вхідна змінна «складність» з областю визначення 0...1 описується термами «низька», «середня», «висока». Вихідна змінна «вартість» з областю визначення 0...3000 тис. грн описується п'ятьма термами – «дуже низька», «низька», «середня», «висока» та «дуже висока». Розширений набір термів вихідної змінної забезпечує плавнішу характеристику моделі та чітко відрізняє шкалу вартості від якісної шкали складності. Функції належності вхідних та вихідної змінної наведено на рисунку 3.1-3.3.

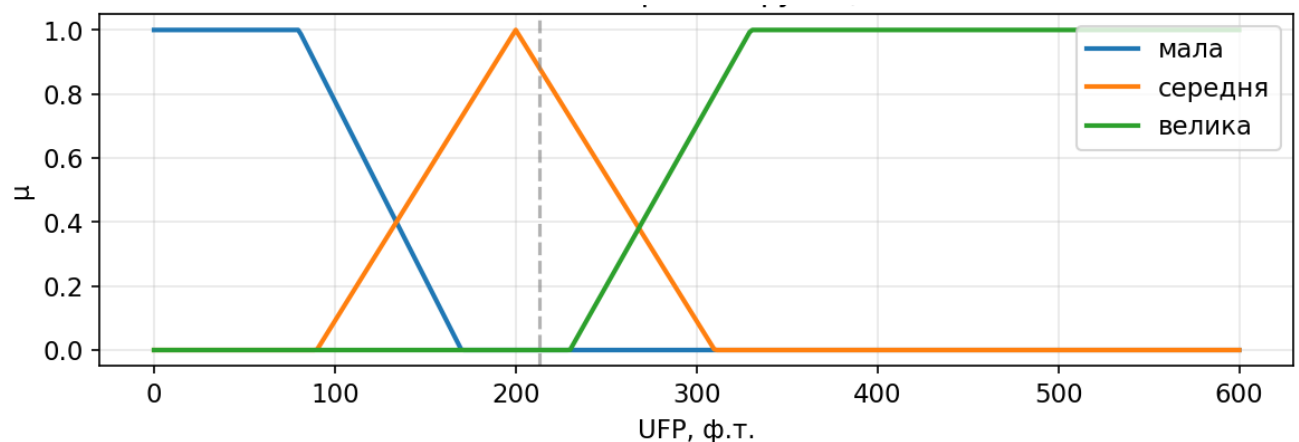


Рисунок 3.1 – Функції належності вхідних змінних “Розмір ПЗ”

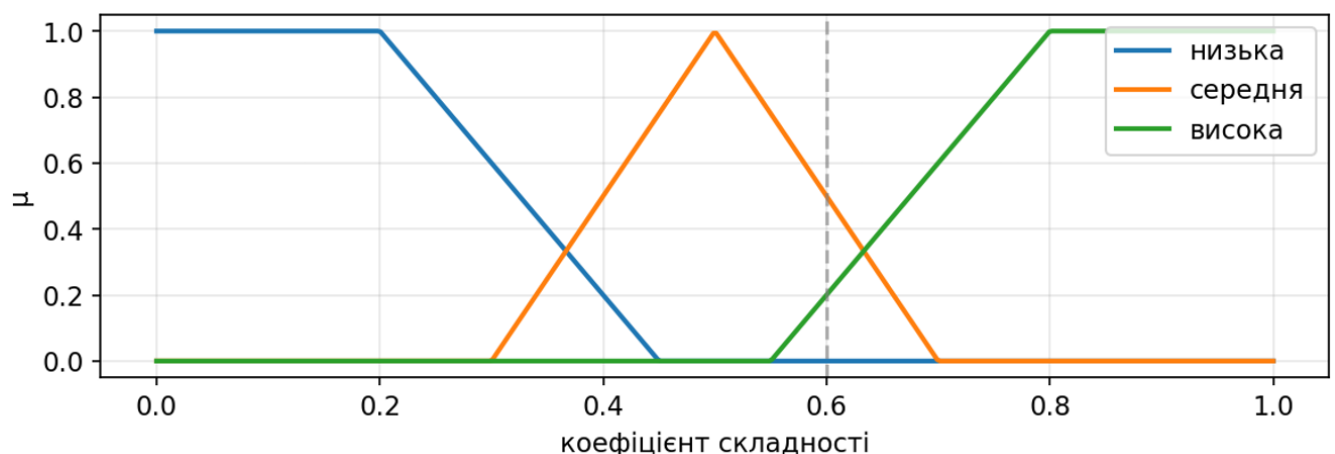


Рисунок 3.2 – Функції належності вхідних змінних “Складність проекту”

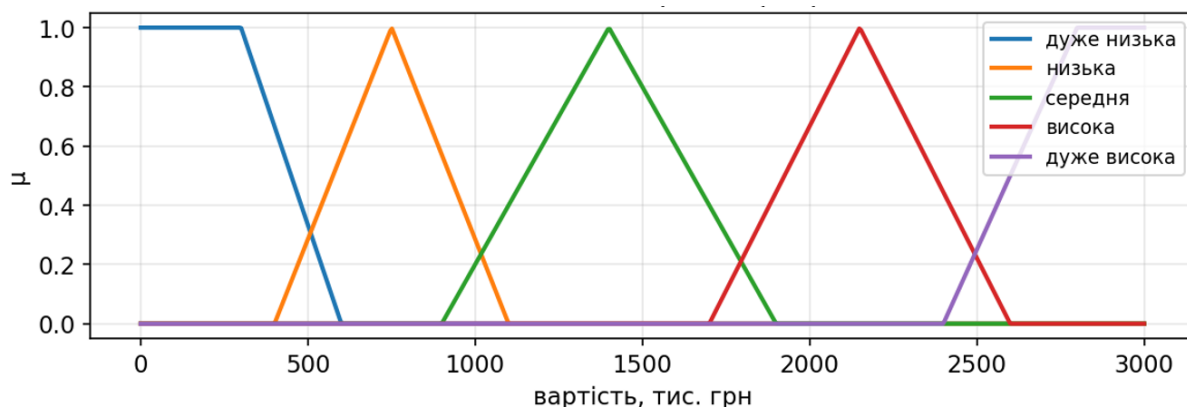


Рисунок 3.3 – Функції належності вихідної змінної “Вартість розробки”

Крайні терми кожної змінної задано трапецієподібними функціями належності, а проміжні – трикутними. Ступінь належності значення до трикутного терму обчислюється за формулою (3.2):

$$\mu(x) = \max(0, \min(\frac{x-a}{b-a}, \frac{c-x}{c-b})) \quad (3.2)$$

де a , b , c – абсциси відповідно лівої основи, вершини та правої основи трикутника. Сусідні терми навмисно перекриваються, що забезпечує плавність вихідної характеристики моделі.

Залежність вихідної змінної від вхідних задано базою правил нечіткого виводу у формі «якщо–то». Кожне правило ставить у відповідність комбінації станів вхідних змінних один терм вихідної змінної. База побудована так, що опрацьовує не лише повні, а й неповні вхідні дані; у матричному вигляді її подано в таблиці 3.2.

Таблиця 3.2 – База правил нечіткого виводу (вихідний терм вартості)

Розмір ПЗ	Складність	Вартість
малий	низька	дуже низька
малий	середня	низька
малий	висока	середня

Продовження таблиці 3.2

малий	відсутня	низька
середня	низька	низька
середня	середня	середня
середня	висока	висока
середня	відсутня	середня
висока	низька	середня
висока	середня	висока
висока	висока	дуже висока
висока	відсутня	висока
відсутня	низька	низька
відсутня	середня	середня
відсутня	висока	висока

У розгорнутому вигляді база складається з п'ятнадцяти продукційних правил. Дев'ять правил відповідають випадку, коли задано обидва входи (3 терми розміру $\times$ 3 терми складності). Ще шість правил передбачають неповні вхідні дані: три правила застосовуються, коли задано лише розмір (складність відсутня), і три – коли задано лише складність (розмір відсутній); у цих випадках оцінка формується за наявним входом. Випадок, коли відсутні обидва входи, виключено, оскільки система не може дати оцінку без жодного параметра, тому загальна кількість правил становить $4 \times 4 - 1 = 15$.

Приклад розрахунку правил: 1: ЯКЩО розмір = малий ТА складність = низька, ТО вартість = дуже низька; правило 2: ЯКЩО розмір = малий ТА складність = середня, ТО вартість = низька; правило 3: ЯКЩО розмір = мала ТА складність = висока, ТО вартість = середня; .

Структуру бази правил у матричному вигляді показано на рисунку 3.2.

База з 15 правил нечіткого виводу
ДН — дуже низька, Н — низька, С — середня, В — висока, ДВ — дуже висока; «—» — невизначений стан

Розмір ПЗ	мала	ДН	Н	С	Н
	середня	Н	С	В	С
	велика	С	В	ДВ	В
	відсутня	Н	С	В	—
		низька	середня	висока	відсутня
		Складність проекту			

Рисунок 3.2 – База з 15 правил нечіткого виводу

Нечіткий вивід виконується за алгоритмом Мамдані у чотири послідовні етапи. Розглянемо роботу моделі на тестовому проєкті з обчисленим вище розміром $UFP = 213$ функціональних точок та складністю $complexity = 0,6$.

На етапі фазифікації для заданих числових значень обчислюються ступені належності до всіх термів вхідних змінних. Для розміру 213 ф.т. ненульовою є належність лише до терму «середня»: $\mu(\text{середня}) = 0,88$, тоді як $\mu(\text{мала}) = \mu(\text{велика}) = 0$. Для складності 0,6 ненульовими є належності до термів «середня» ($\mu = 0,50$) та «висока» ($\mu = 0,20$).

На етапі активації правил для кожного правила обчислюється ступінь його виконання як мінімум ступенів належності передумов. Ненульовий ступінь мають лише два правила: правило 5 «ЯКЩО розмір = середня ТА складність = середня, ТО вартість = середня» зі ступенем $\alpha_5 = \min(0,88; 0,50) = 0,50$ та правило 6 «ЯКЩО розмір = середня ТА складність = висока, ТО вартість = висока» зі ступенем $\alpha_6 = \min(0,88; 0,20) = 0,20$.

На етапі імплікації функцію належності вихідного терму кожного активованого правила обмежують зверху рівнем ступеня виконання цього правила (операція мінімуму). На етапі агрегування одержані обмежені функції об'єднують в одну результуючу нечітку множину операцією максимуму. На етапі

дефазифікації результуючу нечітку множину перетворюють на одне числове значення методом центру ваги за формулою (3.3):

$$\text{Вартість} = \frac{\sum_k x_k \cdot \mu(x_k)}{\sum_k \mu(x_k)} \quad (3.3)$$

де x_k – точки дискретизації області визначення вихідної змінної;

$\mu(x_k)$ – значення агрегованої функції належності в цих точках.

Для тестового проєкту дефазифікація дає прогнозовану вартість 1619 тис. грн, що відповідає області між термами «середня» і «висока» вихідної змінної та узгоджується з ринковими оцінками вартості розробки ERP-модуля відповідного розміру.

3.3 Експериментальні дослідження розробленої моделі

Для перевірки адекватності побудованої моделі проведено серію обчислювальних експериментів. У першому експерименті модель застосовано до шести тестових проєктів різного масштабу – від простого проєкту-візитки до великої маркетплейс-платформи. Для кожного проєкту задано кількості функціональних елементів і експертний коефіцієнт складності, обчислено функціональний розмір UFP та прогнозовану вартість. Результати наведено в таблиці 3.3.

Таблиця 3.3 – Результати оцінювання вартості тестових проєктів

Проект	EI	EO	EQ	ILF	EIF	UFP	Складність	Вартість, тис. грн
Проект-візитка	3	4	2	1	0	50	0,20	233

Продовження таблиці 3.3

Лендинг з формами	6	5	3	2	1	88	0,30	258
Інтернет-магазин MVP	12	9	6	4	2	171	0,45	1400
ERP-модуль (тестовий)	15	10	8	5	3	213	0,60	1619
CRM корпоративна	22	18	12	8	5	341	0,75	2770
Платформа-маркетплейс	35	28	20	14	9	563	0,90	2783

Аналіз таблиці засвідчує, що модель формує монотонно зростаючу оцінку вартості зі збільшенням як розміру, так і складності проєкту – від 233 тис. грн для проєкту-візитки до 2783 тис. грн для маркетплейс-платформи. Перехід між суміжними класами проєктів супроводжується плавною, а не стрибкоподібною зміною вартості завдяки перекриттю термів функцій належності, що є характерною перевагою нечіткого підходу над детермінованими формулами.

Для дослідження поведінки моделі в усьому просторі вхідних параметрів побудовано поверхню нечіткого виводу – залежність прогнозованої вартості одночасно від розміру ПЗ і коефіцієнта складності (рисунок 3.3).

Поверхня має плавно-сходинковий характер: вартість монотонно зростає в обох напрямках, а перекриття термів забезпечує неперервність переходів. Відсутність розривів і немонотонних ділянок підтверджує коректність побудованої бази правил і функцій належності.

У другому експерименті досліджено чутливість моделі до коефіцієнта складності за фіксованого розміру ПЗ (UFP = 213 ф.т.). Складність змінювали в діапазоні [0; 1] з кроком 0,05; одержану залежність наведено на рисунку 3.4.

Поверхня нечіткого виводу: вартість(розмір, складність)

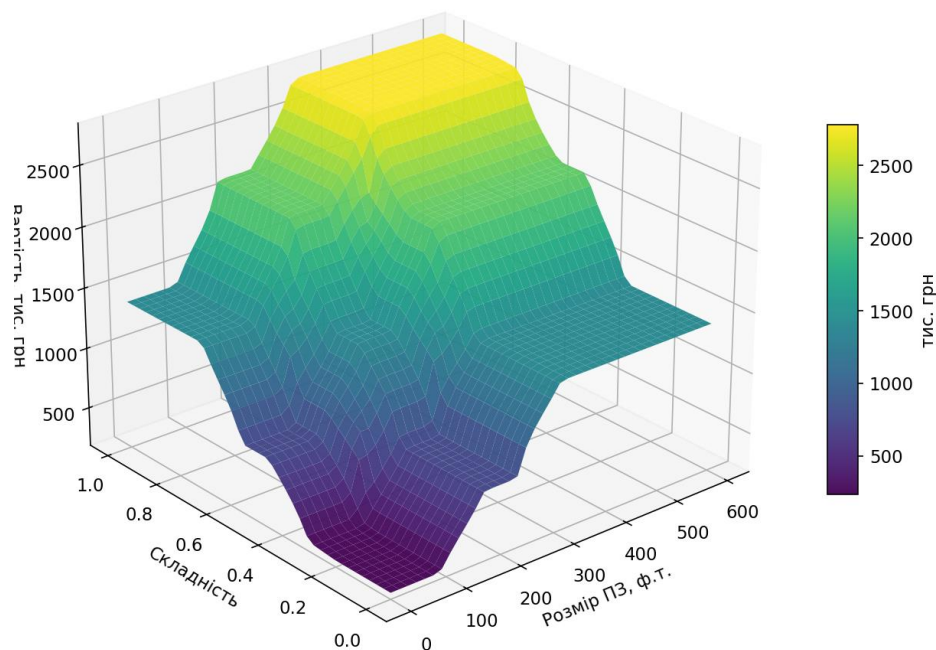


Рисунок 3.3 – Поверхня нечіткого виводу

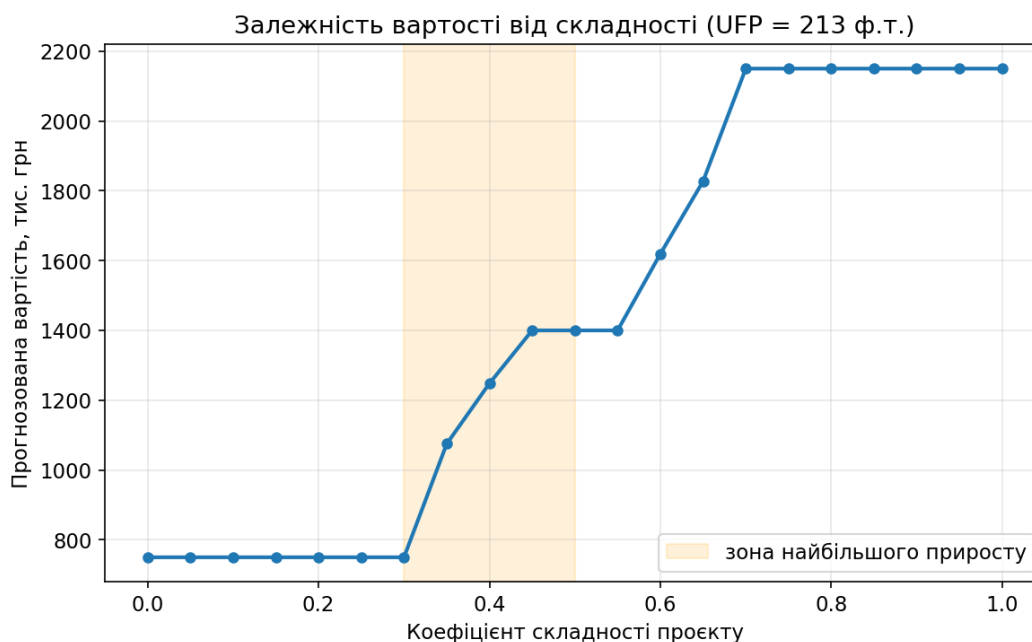


Рисунок 3.4 – Залежність прогнозованої вартості від коефіцієнта складності

Аналіз залежності дає змогу зробити такі висновки. По-перше, вартість монотонно зростає зі збільшенням складності – від 750 тис. грн за мінімальної складності до 2150 тис. грн за максимальної, тобто приблизно у 2,9 раза. По-друге,

найбільший приріст вартості зосереджено в діапазоні складності 0,3–0,5 (виділено на рисунку 3.4); саме в цьому діапазоні похибка експертної оцінки складності найсильніше впливає на прогнозовану вартість, тому менеджерів проекту слід приділяти особливу увагу обґрунтуванню складності проектів цього класу.

Проведені експерименти підтверджують, що розроблений модуль реалізує повний цикл нечіткого виводу – від обчислення функціонального розміру до дефазифікованої оцінки вартості – і формує адекватні, монотонні та плавні оцінки, придатні для попереднього планування бюджету IT-проекту на ранніх стадіях його життєвого циклу.

У третьому розділі розроблено та реалізовано нечіткий модуль оцінювання вартості IT-проектів за алгоритмом Мамдані у середовищі Simulations4All. Визначено структуру вхідних і вихідних даних на основі методу функціональних точок, побудовано функції належності двох вхідних і однієї вихідної лінгвістичних змінних та базу з п'ятнадцяти правил нечіткого виводу. Наведено покроковий числовий приклад роботи модуля та проведено обчислювальні експерименти: оцінювання шести тестових проектів, побудову поверхні виводу й аналіз чутливості. Одержані результати підтверджують працездатність і адекватність розробленої моделі.

					КР.КІ. 10658707.00.00.000.ПЗ	Арк.
						49
Змн.	Арк.	№ докум.	Підпис	Дата		

ВИСНОВКИ

У кваліфікаційній роботі розроблено та досліджено нечітку модель оцінювання вартості IT-проектів, що поєднує метод функціональних точок із нечітким виводом за алгоритмом Мамдані та реалізована у середовищі Simulations4All. За результатами роботи зроблено такі висновки:

1. Проаналізовано сучасні методи оцінювання вартості програмного забезпечення і виявлено їхнє спільне обмеження – потребу в точних числових даних та повернення єдиної точкової оцінки тоді, коли складність проекту описується якісно.

2. Досліджено метод функціональних точок IFPUG, як спосіб об'єктивного, незалежного від технології вимірювання розміру програмного забезпечення на основі п'яти типів функціональних компонентів (EI, EO, EQ, ILF, EIF), а також засади нечіткого виводу за алгоритмом Мамдані як засіб опрацювання якісної невизначеності складності.

3. Розроблено нечітку модель оцінювання вартості на основі двох вхідних лінгвістичних змінних – розміру програмного забезпечення у функціональних точках і коефіцієнта складності проекту – та однієї вихідної змінної – вартості розробки. Сформовано гібридний алгоритм, що поєднує обчислення функціонального розміру з нечітким виводом за базою з п'ятнадцяти правил «якщо–то» та дефазифікацією методом центру ваги.

4. Реалізовано модель у хмарному середовищі Simulations4All без використання комерційних бібліотек нечіткої логіки, що підтвердило можливість повноцінного дослідження загальнодоступними засобами. Адекватність моделі оцінено зіставленням її оцінок із ручними розрахунками на контрольних прикладах.

5. На основі даних обчислювальних експериментів підтверджено монотонність і плавність реакції моделі. Побудовано поверхню нечіткого виводу,

					КР.КІ. 10658707.00.00.000.ПЗ	Арк.
						50
Змн.	Арк.	№ докум.	Підпис	Дата		

яка адекватно відображає зростання вартості зі збільшенням розміру й складності. Аналіз чутливості показав зростання прогнозованої вартості від 750 до 2150 тис. грн (приблизно у 2,9 раза) за зміни складності від 0 до 1, з найбільшим приростом у критичному діапазоні складності 0,3–0,5.

					КР.КІ. 10658707.00.00.000.ПЗ	Арк.
						51
Змн.	Арк.	№ докум.	Підпис	Дата		

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. ISO/IEC 20926:2009. Software and systems engineering – Software measurement – IFPUG functional size measurement method. Geneva: ISO, 2009. 54 p.
2. Boehm B.W. Software Engineering Economics. Englewood Cliffs: Prentice Hall, 1981. 767 p.
3. Boehm B.W., Abts C., Brown A.W. et al. Software Cost Estimation with COCOMO II. Upper Saddle River: Prentice Hall, 2000. 502 p.
4. Ross T.J. Fuzzy Logic with Engineering Applications. 3rd ed. Chichester: Wiley, 2010. 606 p.
5. Mamdani E.H., Assilian S. An experiment in linguistic synthesis with a fuzzy logic controller. International Journal of Man-Machine Studies. 1975. Vol. 7, № 1. P. 1–13.
6. Jorgensen M., Shepperd M. A Systematic Review of Software Development Cost Estimation Studies. IEEE Transactions on Software Engineering. 2007. Vol. 33, № 1. P. 33–53.
7. Putnam L.H. A General Empirical Solution to the Macro Software Sizing and Estimating Problem. IEEE Transactions on Software Engineering. 1978. Vol. SE-4, № 4. P. 345–361.
8. IFPUG. Function Point Counting Practices Manual. Release 4.3.1. Princeton Junction: International Function Point Users Group, 2010. 192 p.
9. Pressman R.S., Maxim B.R. Software Engineering: A Practitioner's Approach. 9th ed. New York: McGraw-Hill, 2020. 768 p.
10. Sommerville I. Software Engineering. 10th ed. Boston: Pearson, 2016. 816 p.
11. Zadeh L.A. Fuzzy sets. Information and Control. 1965. Vol. 8, № 3. P. 338–353.
12. Бублик В.В., Дичка І.А. Основи програмної інженерії: підручник. Київ: Університет «Україна», 2004. 408 с.
13. Kitchenham B.A., Pfleeger S.L. Software Quality: The Elusive Target. IEEE Software. 1996. Vol. 13, № 1. P. 12–21.

					КР.КІ. 10658707.00.00.000.ПЗ	Арк.
						52
Змн.	Арк.	№ докум.	Підпис	Дата		

14. Musilek P., Pedrycz W., Succi G., Reformat M. Software Cost Estimation with Fuzzy Models. ACM SIGAPP Applied Computing Review. 2000. Vol. 8, № 2. P. 24–29.
15. Nguyen V., Steece B., Boehm B. A Constrained Regression Technique for COCOMO Calibration. Proc. of the Second ACM-IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM). 2008. P. 213–222.
16. Albrecht A.J. Measuring Application Development Productivity. Proc. of the Joint SHARE/GUIDE/IBM Application Development Symposium. Monterey: IBM, 1979. P. 83–92.
17. Standish Group International. CHAOS Report. Boston: The Standish Group, 2020. 56 p.
18. Albrecht A.J., Gaffney J.E. Software Function, Source Lines of Code, and Development Effort Prediction: A Software Science Validation. IEEE Transactions on Software Engineering. 1983. Vol. SE-9, № 6. P. 639–648.
19. Symons C.R. Function Point Analysis: Difficulties and Improvements. IEEE Transactions on Software Engineering. 1988. Vol. 14, № 1. P. 2–11.
20. Garmus D., Herron D. Function Point Analysis: Measurement Practices for Successful Software Projects. Boston: Addison-Wesley, 2001. 363 p.
21. Takagi T., Sugeno M. Fuzzy Identification of Systems and Its Applications to Modeling and Control. IEEE Transactions on Systems, Man, and Cybernetics. 1985. Vol. SMC-15, № 1. P. 116–132.
22. Mendel J.M. Fuzzy Logic Systems for Engineering: A Tutorial. Proceedings of the IEEE. 1995. Vol. 83, № 3. P. 345–377.
23. Klir G.J., Yuan B. Fuzzy Sets and Fuzzy Logic: Theory and Applications. Upper Saddle River: Prentice Hall, 1995. 574 p.
24. Zimmermann H.-J. Fuzzy Set Theory and Its Applications. 4th ed. Boston: Kluwer Academic Publishers, 2001. 514 p.
25. Yager R.R., Filev D.P. Essentials of Fuzzy Modeling and Control. New York: Wiley, 1994. 388 p.

					КР.КІ. 10658707.00.00.000.ПЗ	Арк.
						53
Змн.	Арк.	№ докум.	Підпис	Дата		

26. Idri A., Khoshgoftaar T.M., Abran A. Can Neural Networks Be Easily Interpreted in Software Cost Estimation? Proc. of the 11th IEEE International Conference on Fuzzy Systems (FUZZ-IEEE). Honolulu, 2002. Vol. 2. P. 1162–1167.

27. Ahmed M.A., Saliu M.O., AlGhamdi J. Adaptive Fuzzy Logic-Based Framework for Software Development Effort Prediction. Information and Software Technology. 2005. Vol. 47, № 1. P. 31–48.

28. Hodgkinson A.C., Garratt P.W. A Neuro-Fuzzy Cost Estimator. Proc. of the 3rd Conference on Software Engineering and Applications. Scottsdale, 1999. P. 401–406.

29. McConnell S. Software Estimation: Demystifying the Black Art. Redmond: Microsoft Press, 2006. 308 p.

30. Fenton N., Bieman J. Software Metrics: A Rigorous and Practical Approach. 3rd ed. Boca Raton: CRC Press, 2014. 617 p.

31. Jones C. Estimating Software Costs: Bringing Realism to Estimating. 2nd ed. New York: McGraw-Hill, 2007. 644 p.

32. Wiegers K., Beatty J. Software Requirements. 3rd ed. Redmond: Microsoft Press, 2013. 672 p.

33. Boehm B.W., Valerdi R. Achievements and Challenges in COCOMO-Based Software Resource Estimation. IEEE Software. 2008. Vol. 25, № 5. P. 74–83.

34. Albrecht A.J. Function Point Analysis. In: Marciniak J.J. (Ed.) Encyclopedia of Software Engineering. New York: Wiley, 1994. P. 518–524.

35. Heemstra F.J. Software Cost Estimation. Information and Software Technology. 1992. Vol. 34, № 10. P. 627–639.

36. Briand L.C., El Emam K., Bomarius F. COBRA: A Hybrid Method for Software Cost Estimation, Benchmarking, and Risk Assessment. Proc. of the 20th International Conference on Software Engineering (ICSE'98). Kyoto, 1998. P. 390–399.

37. Литвин В.В., Висоцька В.А., Шаховська Н.Б. Алгоритми та структури даних: навчальний посібник. Львів: «Магнолія 2006», 2020. 200 с.

38. ДСТУ ISO/IEC 25010:2025. Інженерія систем і програмних засобів. Вимоги до якості систем і програмних засобів та її оцінювання (SQuaRE). Модель

					КР.КІ. 10658707.00.00.000.ПЗ	Арк.
						54
Змн.	Арк.	№ докум.	Підпис	Дата		

якості продукту (ISO/IEC 25010:2023, IDT). Київ: ДП «УкрНДНЦ», 2025. Чинний від 2025-12-01.

39. Економіка проєктів в комп'ютерній інженерії : методичні вказівки до виконання практичних робіт для студентів першого (бакалаврського) рівня вищої освіти спеціальності 123 «Комп'ютерна інженерія» [Електронний ресурс] / уклад. Н. Я. Савка ; за ред. Л. О. Дубчак ; ЗУНУ. – Режим доступу: <https://wunu.edu.ua>. – Дата звернення: 12.06.2026.

40. Методичні вказівки до випускних кваліфікаційних робіт освітнього рівня «Бакалавр» спеціальності «Комп'ютерна інженерія» / О. М. Березький, Г. М. Мельник, Л. О. Дубчак [та ін.] ; під ред. О. М. Березького. – Тернопіль : ЗУНУ, 2024. – 52 с.

					КР.КІ. 10658707.00.00.000.ПЗ	Арк.
						55
Змн.	Арк.	№ докум.	Підпис	Дата		