Харківський національний університет імені В.Н. Каразіна
Міністерство освіти і науки України

Кваліфікаційна наукова
праця на правах рукопису

Панченко Артем Сергійович

УДК 004.05:004.03

ДИСЕРТАЦІЯ

“КОАЛГЕБРАЇЧНІ ЗАСОБИ СПЕЦИФІКАЦІЇ ТА АНАЛІЗУ СТАТИСТИЧНИХ ОБМЕЖЕНЬ ПОВЕДІНКИ РОЗПОДІЛЕНИХ СИСТЕМ”

Спеціальність 122 – "Комп’ютерні науки"

(Галузь знань 12 – "Інформаційні технології")

Подається на здобуття наукового ступеня
доктора філософії з комп’ютерних наук.
Дисертація містить результати власних досліджень. Використання
ідей, результатів і текстів інших авторів мають посилання на відповідне
джерело.

_____ А. С. Панченко

Науковий керівник Жолткевич Григорій Миколаєвич,
доктор технічних наук, професор.

Харків - 2024

АНОТАЦІЯ

Панченко А.С. Коалгебраїчні засоби специфікації та аналізу статистичних обмежень поведінки розподілених систем - Кваліфікаційна наукова праця на правах рукопису.

Дисертація на здобуття ступеня доктора філософії за спеціальністю 122 – Комп'ютерні науки (Галузь знань 12 – Інформаційні технології). – Харківський національний університет імені В. Н. Каразіна, Міністерства освіти і науки України, Харків, 2024

Дисертаційна робота присвячена вивченню можливостей удосконалення методів специфікації дискретних динамічних систем різного типу включно з розподіленими. Пропонується застосування універсальних коалгребр як основного інструменту для створення формальної математичної моделі поведінки систем з високим рівнем абстракції, що забезпечить можливість моделювання динаміки у системах без урахування їх несуттєвих особливостей та вивчення їх критичних станів з метою розробки стратегій уникання подібних станів та алгоритмів виходу з них.

Дисертаційне дослідження охоплює теоретичні підходи до аналізу та специфікації поведінки дискретних динамічних систем, зокрема розподілених, за допомогою моделювання з використанням універсальних коалгебр та теорії категорій. Запропоновано вивчення однієї з основних концепцій універсальної коалгебри - фінальних коалгебр для рандомних систем, що породжують вихідні данні. Такі моделі є найбільш репрезентативними для вивчення тому що несуть у собі усі основні властивості обраної для дослідження категорії систем.

Таким чином, описаний у дисертації підхід дає можливість створювати універсальні методи дослідження не окремої системи, як у стандартних підходах зі створенням артефактів специфікації вимог до програмного забезпечення, а цілого класу систем за допомогою високого рівня абстракції запропонованої математичної моделі. Результати, отримані під час дисер-

таційного дослідження, можуть бути використані інженерами та дослідниками, що займаються проектуванням, розробкою та аналізом розподілених та динамічних систем.

У вступі до дисертаційної роботи обґрунтовано актуальність теми дослідження, показано зв'язок роботи з науковими темами. Сформульовано мету дослідження, а саме: удосконалення уніфікованих методів специфікації та аналізу дискретних динамічних систем різного типу включно з розподіленими за рахунок використання універсальних коалгебр. Визначено об'єкт дослідження, а саме: процеси специфікації та анализу поведінки розподілених обчислювальних систем різної природи, як детермінованих так і рандомних. Визначено предмет дослідження: а саме колагебраїчні моделі дискретних динамічних систем та методи їх аналізу. Методи дослідження базуються на використання принципів теорії систем, що є загальною теоретичною базою дослідження; теорії категорій, що забезпечує універсальну формальну мову для дослідження систем різної природи; теорії універсальних коалгебр, що дозволяє моделювати поведінку дискретних динамічних систем різного типу, включаючи розподілені в рамках єдиного фреймворку; фінальної семантики, яка визначає сенс властивостей і відношень притаманих всім системам заданого типу; методу коіндукції, який дозволяє доводити визначати та доводити властивості фінальної системи.

У першому розділі здійснено аналітичний аналіз підходів до підвищення якості процесу специфікації обмежень розподілених систем за рахунок використання формальних методів моделювання та специфікації, зокрема з використанням універсальних коалгебр та теорії категорій. Слід зазначити, що ці методи спрямовані на семантичний аналіз поведінки моделі системи, що вивчається, який, в свою чергу, поділяється на нотаційний та денотаційний підходи. Дослідження актуального стану проблеми показало, що аналіз та специфікацію поведінки системи на етапі її проектування буде якіснішим з точки зору уникнення помилок, зумовлених "людським фактором якщо проводити за допомогою абстрактних математичних моделей на основі універсальних коалгебр із використанням монад для ізоляції

поведінки системи та теорії категорій, що зумовлює актуальність цього дисертаційного дослідження. У завершенні розділу була поставлена задача дисертаційного дослідження, а саме: розробка теоретичних засад та практичних методів до уніфікації методів специфікації поведінки дискретних динамічних систем з урахуванням їх різноманітності та високих вимог до безпеки (з точки зору завчасного виявлення критичних станів) та прогнозованості їх роботи.

У другому розділі дисертаційного дослідження була надана основна інформація щодо використання універсальної коалгебри з неформальною інтерпретацією абстрактних математичних моделей. Були описані основні теоритичні засади використання фінальної коалгебри як основного об'єкту вивчення для категорії систем для її аналізу. У розділі розглядалися варіанти виведення фінальної коалгебри для моделі системи. Слід зазначити, що така задача не є тривіальною, а сама фінальна коалгебра не завжди існує. Для того щоб оптимізувати процес виведення фінальної коалгебри було запропоновано спочатку перевірити факт її існування з використанням декартових квадратів. Було сформульовано та доведено достатню умову збереження ендофунктором у категорії множин слабких декартових квадратів, що дозволило отримати інструмент перевірки існування фінальної коалгебри для системи, що вивчається.

У третьому розділі дисертаційної роботи було наведено практичні підходи до аналізу динамічних систем за допомогою універсальних коалгебр, а саме: створено коалгебраїчну модель для дискретної детермінованої систем, що породжує вихідні данні, та визначена її фінальна коалгебра. Для урахування стохастичності процесів у складних динамічних системах, зокрема розподілених та кіберфізичних, була створена коалгебраїчна модель рандомної дискретної динамічної системи, що породжує вихідні данні. Реалізація введення стохастичності у вибір наступного стану системи була зроблена з використанням монади розподілу ймовірностей Джирі. Таким чином, створена модель детермінованої системи була розширена, а знання про неї були перевикористані. Для рандомної системи також було доведено

факт існування фінальної коалгебри, яка була виведена та проаналізована. Фінальна коалгебра для рандомних системи має форму дерева та названа у роботі нескінченним деревом без листя.

У четвертому розділі дисертаційної роботи були описані методи побудови імітаційних моделей представницької системи (яка завжди буде фінальною системою) для обраної категорій систем. Були представлені методології до визначення простору станів системи, що буде носієм моделі, створеною з використанням універсальних коалгебр. Був описаний коалгебраїчний підхід до динамічного аналізу (імітаційного моделювання) складних систем за допомогою універсальних коалгебр.

У дисертаційній роботі вирішена актуальна науково-прикладна задача аналізу та специфікації поведінки динамічних систем за допомогою використання універсальних коалгебр.

В результаті проведеного дослідження були отримані знання щодо підходів до моделювання рандомних систем та процесів за допомогою використання універсальних коалгебр, що дозволяють зробити висновки щодо того, що використання таких методів під час проектування систем є адекватним вибором та має значні переваги перед класичним підходом зі створенням артефактів специфікації вимог до системи через унеможливлення помилки, що може бути спричинена людським фактором. Окремо слід, що під час дисертаційного дослідження було виведено достатню умову існування фінальної коалгбери, що є корисним для усіх дослідників, що використовують універсальні коалгебри як інструмент моделювання

Наукова новизна отриманих результатів полягає у наступному:

1. **Вперше запропоновано** формалізацію рандомізації дискретної динамічної системи шляхом лівої композиції її ендофунктора з монадою Джірі.

2. **Вперше сформульовано** та доведено достатню умову для збереження ендофунктором категорії множин слабких декартових квадратів, що дозволяє встановити факт існування фінальної системи.

3. **Дістала подальшого розвитку** техніка обчислення фінальної системи певного типу шляхом використанням методу коіндукції у разі доведеного факту існування фінальної системи.

4. **Дістав подальшого розвитку** метод синтезу моделей для динамічного аналізу (імітаційного моделювання) складних систем з використанням техніки універсальних коалгебр.

Сукупність отриманих у дисертації нових наукових результатів, позитивна оцінка їхньої достовірності, наукової та практичної значущості дають змогу вважати сформульовану наукову задачу удосконалення уніфікованих методів специфікації та аналізу дискретних динамічних систем різного типу, включно з розподіленими за рахунок використання універсальних коалгебр – розв'язаною, а поставлену мету – досягнутою.

Ключові слова: *системи геолокації та трекінгу, розподілені та паралельні системи, аналіз вимог до системи, специфікація поведінки системи, концептуальна модель, моделі реального часу, імітаційне моделювання (симуляція), перехідні системи включно з автоматами, система підтримки та прийняття рішень, машинне навчання, алгебра, синтез програм, верифікація поведінки програмних систем, формальні методи верифікації, програмна логіка, логічна узгодженість, коректність моделі, повнота моделі забезпечення надійності та безперебійної роботи системи, обчислювальні системи, симуляція роботи системи, автомати, автоматизація процесів проектування, системи з підвищеними вимогами до надійності, універсальна коалгебра, фінальна коалгебра, (слабкий) декартовий квадрат*

ABSTRACT

Artem PANCHENKO Coalgebraic methods of specification and analysis of statistical limitations of distributed systems behavior – Qualification scholarly paper: a manuscriptt.

A dissertation submitted for the degree of Doctor of Philosophy in Information Technology: Speciality 122 – Computer science. V.N. Karazin Kharkiv National University, Ministry of Education and Science of Ukraine, Kharkiv, 2024.

The dissertation work is dedicated to studying the possibilities of improving methods for specifying discrete dynamic systems of various types, including distributed systems. The proposed approach involves the use of universal coalgebras as the main tool for creating a formal mathematical model of system behavior at a high level of abstraction. This allows for modeling system dynamics without considering insignificant details and studying their critical states to develop strategies for avoiding such states and algorithms for recovery.

The dissertation research encompasses theoretical approaches to the analysis and specification of discrete dynamic systems, particularly distributed ones, through modeling using universal coalgebras and category theory. It proposes studying one of the fundamental concepts of universal coalgebra - final coalgebras for random systems that generate output data. These models are the most representative for study because they encompass all the main properties of the selected category of systems.

Thus, the approach described in the dissertation provides the ability to create universal methods for studying not a single system, as in standard approaches involving the creation of software requirements specification artifacts, but an entire class of systems using the high level of abstraction of the proposed mathematical model. The results obtained during the dissertation research can be used by engineers and researchers involved in the design, development, and analysis of distributed and dynamic systems.

In the introduction, the dissertation work justifies the relevance of the research topic and shows the connection of the work with scientific themes. The research goal is formulated as the improvement of unified methods for specifying and analyzing discrete dynamic systems of various types, including distributed ones, through the use of universal coalgebras. The object of the study is defined as the processes of specifying and analyzing the behavior of distributed computing systems of various natures, both deterministic and random. The subject of the study is coalgebraic models of discrete dynamic systems and methods for their analysis. The research methods are based on the principles of systems theory, providing a general theoretical basis; category theory, offering a universal formal language for studying systems of various natures; universal coalgebras theory, allowing the modeling of the behavior of discrete dynamic systems, including distributed ones, within a single framework; final semantics, defining the meaning of properties and relations inherent to all systems of a given type; and the method of coinduction, which enables the determination and proof of properties of the final system.

In the first chapter, an analytical review of approaches to improving the quality of the specification process for distributed systems using formal modeling and specification methods, particularly universal coalgebras and category theory, is conducted. These methods focus on the semantic analysis of the system model behavior being studied, divided into notational and denotational approaches. The current state of the problem shows that the analysis and specification of system behavior during its design phase will be more accurate in terms of avoiding errors due to the "human factor"if conducted using abstract mathematical models based on universal coalgebras, employing monads to isolate system behavior and category theory, thus justifying the relevance of this dissertation research. The chapter concludes by setting the research task, namely the development of theoretical foundations and practical methods for unifying the specification methods of discrete dynamic systems' behavior, considering their diversity and high safety requirements (in terms of early detection of critical states) and the predictability of their operation.

In the second chapter, the dissertation provides basic information on the use of universal coalgebras with informal interpretations of abstract mathematical models. The main theoretical principles of using final coalgebras as the primary object of study for the system category are described. Various ways of deriving the final coalgebra for the system model are considered. It should be noted that this task is non-trivial, and the final coalgebra does not always exist. To optimize the process of deriving the final coalgebra, it is proposed to first check for its existence using Cartesian squares. A sufficient condition for the preservation of weak Cartesian squares by the endofunctor in the category of sets is formulated and proven, providing a tool for verifying the existence of the final coalgebra for the studied system.

In the third chapter, practical approaches to analyzing dynamic systems using universal coalgebras are presented, specifically the creation of a coalgebraic model for a discrete deterministic system that generates output data and the determination of its final coalgebra. To account for the stochasticity of processes in complex dynamic systems, particularly distributed and cyber-physical systems, a coalgebraic model of a random discrete dynamic system that generates output data was created. The stochasticity in selecting the next state of the system was implemented using the Giry monad of probability distributions. Thus, the deterministic system model was extended, and knowledge about it was reused. For the random system, the existence of the final coalgebra was also proven, derived, and analyzed. The final coalgebra for the random system takes the form of a tree and is referred to in the work as an infinite tree without leaves.

In the fourth chapter, methods for constructing simulation models of a representative system (which will always be the final system) for the chosen category of systems are described. Methodologies for determining the state space of the system, which will be the carrier of the model created using universal coalgebras, are presented. A coalgebraic approach to the dynamic analysis (simulation modeling) of complex systems using universal coalgebras is described.

The dissertation addresses a relevant scientific and practical problem of analyzing and specifying the behavior of dynamic systems using universal coalgebras.

As a result of the research, knowledge on modeling random systems and processes using universal coalgebras was obtained, allowing conclusions to be drawn about the adequacy and significant advantages of using such methods during system design over the classical approach involving the creation of system requirements specification artifacts, minimizing errors due to the human factor. Notably, during the dissertation research, a sufficient condition for the existence of the final coalgebra was derived, which is useful for all researchers using universal coalgebras as a modeling tool.

The scientific novelty of the obtained results is as follows:

1. **For the first time**, the randomization formalization of a discrete dynamic system through the left composition of its endofunctor with the Giry monad is proposed.

2. **For the first time**, a sufficient condition for the preservation of weak Cartesian squares by an endofunctor in the category of sets is formulated and proven, allowing the establishment of the existence of the final system.

3. The technique for calculating the final system of a certain type using the coinduction method, when the existence of the final system is proven, **has been further developed**.

4. The method for synthesizing models for dynamic analysis (simulation modeling) of complex systems using universal coalgebras **has been further developed**.

The set of new scientific results obtained in the dissertation, their positive evaluation regarding reliability, scientific, and practical significance, allows considering the formulated scientific task of improving unified methods for

specifying and analyzing discrete dynamic systems of various types, including distributed ones, using universal coalgebras as solved, and the set goal as achieved.

Keywords: *geolocation and tracking systems, distributed and parallel systems, analysis of system requirements, system behavior specification, conceptual model, real time models, simulation, transitional systems including automatic machines, support and decision-making system, machine learning, algebra, program synthesis, verification of the behavior of software systems, formal methods of verification, software logic, logical consistency, correctness of the model, completeness of the model ensuring reliability and smooth operation of the system, computer systems, simulation of system operation, machines, automation of design processes, systems with increased reliability requirements, universal coalgebra, final coalgebra, (weak) Pullback*

Список публікацій здобувача

Статті у наукових фахових виданнях, що входять до міжнародних наукометричних баз

1. Grygoriy Zholtkevych, Artem Panchenko. About One Possible Tool for Analysing Safeness of Discrete Dynamic Systems. 2023. 13th International Conference on Dependable Systems, Services and Technologies (DESSERT). pp. 1-7.
DOI:http://dx.doi.org/10.1109/DESSERT61349.2023.10416480
URL: https://ieeexplore.ieee.org/document/10416480
Ключові слова: discrete system, actual system state, forecasted system state, dynamical transformation, endofunctor, universal coalgebra, coalgebraic morphism, colimit, eventual coalgebra
(Особистий внесок: виведення частини моделей на основі універсальних коалгебр для розповсюджених динамічних систем а також написання частини тексту та його переклад
Особистий внесок Grygoriy Zholtkevych: формалізація фінальної коалгебри для дискретної детермінованої системи а також написання частини тексту та його переклад)

2. Grygoriy Zholtkevych, Artem Panchenko. An Approach to Construct Final Random System with Output. Communications in Computer and Information Science. 2022., Vol. 1698. Pp. 3 – 22.
DOI: https://doi.org/10.1007/978-3-031-20834-8_1
URL: https://link.springer.com/chapter/10.1007/978-3-031-20834-8_1
Ключові слова: coalgebra, coalgebraic morphism, final system, anamorphism
(Особистий внесок: розробка теоритичних засад існування фінальної коалгебри для обраного типу динамічних систем, написання частини тексту та переклад а також виступ на конференції
Особистий внесок Grygoriy Zholtkevych: формалізація фінальної коалгебри

для рандомних дискретних систем з вхідними даними а також написання частини тексту та переклад)

3. Grygoriy Zholtkevych, Artem Panchenko. Coalgebraic Understanding of Random Systems with Output. CEUR Workshop Proceedings. 2021. Vol 3031. Pp. 296 – 306.

Ключові слова: discrete system, system with output, random system, coalgebraic approach

URL: https://ceur-ws.org/Vol-3013/20210296.pdf

(Особистий внесок здобувача розробка підходів до впровадження стохастичності до детермінованої моделі за допомогою монади вірогіднісного розподілу Джирі, написання частини тексту та переклад а також виступ на конференції

Особистий внесок Zholtkevych: формалізація моделі рандомної дискретної системи з виходами з використанням універсальних коалгебр а також написання частини тексту та переклад)

4. Panchenko A., Prokhorchenko A., Panchenko S., Dekarchuk O., Gurin D., Medvediev I. Predicting the estimated time of cargo dispatch from a marshaling yard. Eastern-European Journal of Enterprise Technologies.2020. Vol. 4. Is. 3 (106). Pp. 6–1

DOI https://doi.org/10.15587/1729-4061.2020.209912

URL: https://journals.uran.ua/eejet/article/view/209912

Ключові слова: railroad, marshaling yard, cargo dispatch, expected departure time, machine learning

(Особистий внесок здобувача: розробка імітаційної моделі залізниці з оглядом на її природу, яку можна охарактеризувати як розподілену, написання частини тексту та переклад

Особистий внесок Prokhorchenko A: аналіз поточного стану проблеми, що досліджується, постановка задачі дослідження а також написання частини тексту та переклад

Особистий внесок Panchenko S: створення моделі на основі машинного навчання для прогнозування часу доставки вантажу а написання части-

ни тексту та переклад

Особистий внесок Dekarchuk O: дослідження вантажообігу на залізниці з метою покращення роботи імітаційної моделі а також написання частини тексту та переклад

Особистий внесок Gurin D: генерування тестових даних для навчання моделей машинного навчання а також написання частини тексту та переклад

Особистий внесок Medvediev I: крос валідація роботи моделі машинного навчання а також написання тексту та переклад)

Статті у наукових фахових виданнях України

5. Grygoriy Zholtkevych, Artem Panchenko. The technique of modeling Cyberphysical systems using Coalgebra. Bulletin of V.N. Karazin Kharkiv National University,. Series «Mathematical Modeling. Information Technology. Automated Control Systems»,. 2023. Vol. 58. Pp. 47-53.
DOI: https://doi.org/10.26565/2304-6201-2023-58-05
URL: https://periodicals.karazin.ua/mia/article/view/23500
Ключові слова: Cyber-Physical Systems, Coalgebra, Dynamic Systems Modeling, Category Theory, Final Coalgebra *(Особистий внесок здобувача: вивчення теоритичних засад використання декартових квадратів для доведення існування фінальної коалгебри для побудованих моделей динамічних систем, написання частини тексту та переклад*

Особистий внесок Grygoriy Zholtkevych:моделювання динамічних систем за допомогою універсальних коалгебр та визначення фінальних коалгебр для дискретних динамічних систем) Kharkiv National University,. Series «Mathematical Modeling. Information Technology. Automated Control Systems»,. 2023. Vol. 58. Pp. 47-53.
DOI: https://doi.org/10.26565/2304-6201-2023-58-05
URL: https://periodicals.karazin.ua/mia/article/view/23500
Ключові слова: Cyber-Physical Systems, Coalgebra, Dynamic Systems Modeling, Category Theory, Final Coalgebra *(Особистий внесок здобувача: вивче-*

ння теоритичних засад використання декартових квадратів для доведення існування фінальної коалгебри для побудованих моделей динамічних систем, написання частини тексту та переклад

ЗМІСТ

ВСТУП 19
РОЗДІЛ 1. АНАЛІЗ ПІДХОДІВ ДО ПІДВИЩЕННЯ ЯКОСТІ ПРОЦЕСУ СПЕЦИФІКАЦІЇ ОБМЕЖЕНЬ РОЗПОДІЛЕНИХ СИСТЕМ 32
1.1 Аналіз проблематики специфікації поведінки систем на етапі проєктування та її актуальність 32
1.2 Аналіз формальних методів специфікації поведінки систем. Обґрунтування доцільності їх використання 41
1.3 Аналіз підходів до специфікації поведінки систем з використанням універсальних коалгебр. Обґрунтування доцільності їх використання 54
1.4 Постановка задачі дисертаційного дослідження 58
Висновки до розділу 1 61
РОЗДІЛ 2. ВИКОРИСТАННЯ УНІВЕРСАЛЬНИХ КОАЛГЕБР ДЛЯ МОДЕЛЮВАННЯ ТА ВИВЧЕННЯ ДИСКРЕТНИХ ДИНАМІЧНИХ СИСТЕМ 63
2.1 Основні поняття теорії універсальних коалгебр 64
2.1.1 Фінальна коалгебра 68
2.1.2 Бісимуляція 70
2.1.3 Коіндукція 75
2.2 Важливість фінальної коалгебри для дослідження поведінки системи 77
2.3 Підходи до доведення існування фінальної коалгебри за допомогою декартових квадратів 78
2.3.1 Визначення слабкого декартового квадрату 79
2.3.2 Збереження ендофункторами декартових квадратів . 81
2.3.3 Збереження декартових квадратів підсистемами . . . 84
2.3.4 Достатня умова існування фінальної системи 86

Висновки до Розділу 2 86
РОЗДІЛ 3. ВИКОРИСТАННЯ УНІВЕРСАЛЬНИХ КОАЛГЕБР ДЛЯ МОДЕЛЮВАННЯ ТА ВИВЧЕННЯ ПОВЕДІНКИ ДИСКРЕТНИХ ДИНАМІЧНИХ СИСТЕМ 88
3.1 Моделювання детермінованої системи з вихідними даними за допомогою універсальної коалгебри 88
3.2 Підходи до впровадження стохастичності у детерміновану модель за допомогою монади Джирі 91
3.3 Визначення універсальної коалгебри для Рандомної системи з вихідними даними 93
3.4 Визначення фінальної коалгебри для рандомної систем . . . 95
3.4.1 Концепція помічених дерев без листя 96
3.4.2 Побудова фінальної системи 100
3.4.3 Побудова родини морфізмів 102
3.4.4 Фінальна система для рандомної дискретної системи з виходами 105
Висновки по Розділу 3 106
РОЗДІЛ 4. Практичне застосування універсальних коалгебр для специфікації та аналізу поведінки систем 107
4.1 Особливості складних програмних систем та програмних компонентів з перспективи їх специфікації 107
4.2 Метод синтезу імітаційних моделей складних динамічних систем з використанням універсальних коалгебр 111
4.3 Практичне застосування теорії коалгебр при дослідження складних розподілених систем 113
Висновки по Розділу 4 115
ВИСНОВОК 117
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 120
ДОДАТКИ 131
Додаток А. Список публікацій здобувача за темою дисертації . . 131

Додаток Б. Акти про практичне застосування отриманих результатів 134

ВСТУП

Актуальність теми. Специфікація динаміки системи є формальним описом очікуваної поведінки програмного або апаратного забезпечення [1]. Такий підхід до опису системи використовується для визначення і верифікації її функцій і властивостей в контексті розробки, тестування та верифікації програмного забезпечення. Можна виділити два основних підходи до проведення аналізу системи, що проектується, з метою її специфікації, а саме:

1. Неформальний. Цей підхід полягає у використанні звичайної природної мови для написання документації. Він включає створення текстових описів, діаграм, схем та інших засобів, які легко може зрозуміти команда розробки та тестування системи. Прикладом подібної специфікації є сценарії використання (Use Cases), що описують типові сценарії взаємодії користувача із системою. Серед переваг такого підходу можна виділити зрозумілість для широкого кола учасників проекту та простоту та швидкість виконання подібної специфікації. Недоліками подібного підходу є можливість неоднозначного тлумачення результатів специфікації та відсутність формальної верифікації.

2. Формальний. Використовує математичні моделі, логіку та формальні методи для специфікації поведінки системи. Цей підхід передбачає використання формальних мов та структур для однозначного і точного опису системи. Прикладами такого підходу є Логічна специфікація або Коалгебраїчна специфікація. Недоліками подібного підходу є складність у розумінні та застосуванні, а також висока вартість розробки та підтримки формальних специфікацій.

Отже, неформальні та формальні підходи до специфікації програмного забезпечення мають свої переваги та недоліки. Вибір підходу залежить від вимог проекту, складності системи та необхідності формальної верифікації.

Формальні методи в специфікації програмного забезпечення застосовуються в ряді випадків, коли важлива висока точність, однозначність і можливість формальної верифікації системи [2]. Основні випадки, коли слід застосовувати формальні методи, включають:

1. Критично важливі системи. Такі системи, де помилки можуть мати серйозні наслідки для безпеки, здоров'я або значних фінансових втрат. Прикладом подібних систем є авіоніка та медичні пристрої.

2. Складні системи. Системи з високою складністю архітектури, де важко забезпечити коректність за допомогою традиційних методів тестування. Прикладом подібних систем є розподілені та конкурентні системи.

3. Системи з високими вимогами до надійності та стійкості. Такі системи, де необхідно забезпечити високу надійність і стійкість до збоїв. Прикладом подібних систем є космічні апарати та інфраструктурні системи

Окремо слід зазначити, що розробка програмного забезпечення станом на сьогодні вимагає великих матеріальних затрат та їх оптимізація є постійною проблемою індустрії. Результати дослідження IBM Systems Sciences Institute [3] наводять наступні результати аналізу фінансових витрат на виправлення дефектів у ПЗ у залежності від етапу життєвого циклу, на якому вони були виявлені.

1. Етап специфікації системи: на цьому етапі помилки відносно легко та недорого виправити, оскільки зміни в основному стосуються документації та планів, а не безпосередньої реалізації програмного продукту;

2. Етап безпосередньої реалізації програмного продукту: якщо помилку виявлено на етапі кодування, її виправлення буде дорожчим, тому що вже написаний код доведеться змінювати і заново тестувати.

3. Етап тестування: на цьому етапі вартість виправлення помилки суттєво зростає, тому що необхідно не лише змінити код, а й провести повний цикл тестування для підтвердження виправлення.

4. Етап експлуатації: найдорожчі помилки - це ті, що виявляються після випуску продукту. Їх виправлення включає не лише технічні роботи, а й можливі витрати на підтримку клієнтів, оновлення вже встановлених систем, можливі репутаційні втрати і т.д.

Вважається, що якщо взяти за 1 умовну одиницю вартість виправлення помилки, що було виявлено на етапі проектування, то вартість виправлення такої помилки на етапі реалізації ПО буде становити 6.5 у.о., на етапі тестування - 15 у.о. та на етапі експлуатації - 100 у.о. Таким чином, можна зробити висновок щодо необхідності впровадження у процес розробки ПО таких практик, що дозволять виявляти та виправляти помилки на ранніх етапах життєвого циклу ПО.

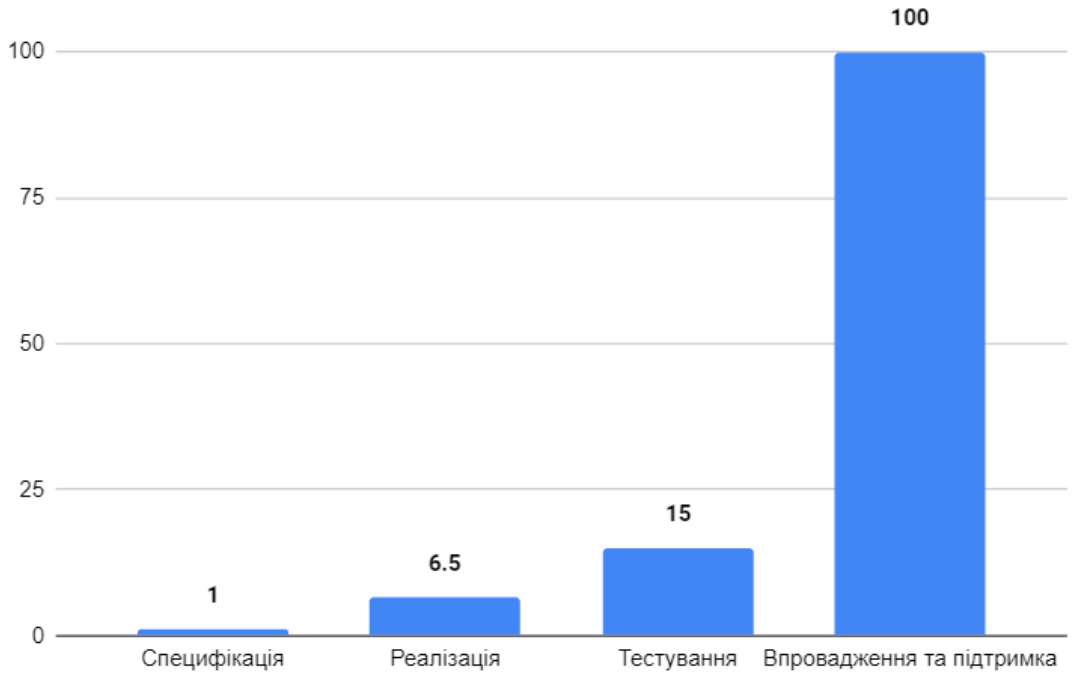


Рис. 1. Відносна вартість виправлення помилок

Таким чином, використання формальної верифікації систем крім підвищеної надійності рішення має наступні економічні переваги:

1. Зниження витрат на тестування і виправлення помилок. Традиційні методи тестування, такі як модульне тестування, інтеграційне тестування та системне тестування, хоча і є ефективними, мають свої обмеження. Формальна верифікація забезпечує систематичний підхід до виявлення помилок, що може значно зменшити витрати на тестування та виправлення помилок, особливо на пізніх стадіях розробки, де оптимізація витрат є критично важливою через високу собівартість внесення змін до програмного продукту.

2. Довгострокові економічні вигоди. Формальна верифікація може забезпечити довгострокові економічні вигоди за рахунок підвищення якості програмного забезпечення. Зменшення кількості помилок на етапі експлуатації знижує витрати на підтримку та обслуговування системи, а також покращує репутацію компанії, що розробляє програмне забезпечення.

Емпіричні дослідження економічної вигоди використання формальних підходів до специфікації програмних продуктів показують, що в галузі розробки авіоніки, подібні методи можуть знизити загальні витрати на розробку на 15-20 % за рахунок зменшення кількості помилок на стадії тестування та експлуатації. В галузі медичних пристроїв застосування формальної верифікації знижує витрати на сертифікацію та відповідність регуляторним вимогам на 30 %.

Серед усіх засобів формальної специфікації програмних продуктів слід виділити специфікацію з використанням універсальних коалгбер [4]. Коалгебраїчна специфікація є потужним формальним методом, який використовується для моделювання динамічних систем. Вона ґрунтується на понятті коалгебри і є природним доповненням до алгебраїчних методів, що часто використовуються для статичних аспектів програмного забезпечення. У цьому контексті коалгебраїчна специфікація має ряд переваг над іншими формальними методами, а саме

1. Природне моделювання динамічних систем. Однією з основних пере-

ваг коалгебраїчної специфікації є її здатність природно моделювати динамічні системи. Коалгебри особливо добре підходять для опису об'єктів і їхньої поведінки в часі, що дозволяє легко моделювати системи з безперервним або дискретним зміною станів. Це робить коалгебраїчну специфікацію ідеальною для систем, де важливо описати перехідні процеси та взаємодії між компонентами.

2. Підтримка об'єктно-орієнтованого програмування. Коалгебраїчні методи тісно пов'язані з об'єктно-орієнтованим програмуванням (ООП) [5]. Вони дозволяють моделювати об'єкти як коалгебри, де кожен об'єкт визначається його станами і переходами між ними. Це забезпечує природну інтеграцію з ООП-парадигмою, де поведінка об'єктів описується методами, а стани — властивостями. Таким чином, коалгебраїчна специфікація спрощує формалізацію та аналіз об'єктно-орієнтованих систем.

3. Сильна підтримка властивостей інваріантності. Коалгебраїчна специфікація забезпечує ефективні засоби для опису і перевірки властивостей інваріантності — властивостей, які залишаються незмінними при переходах між станами. Це особливо важливо для критично важливих систем, де необхідно гарантувати, що певні умови завжди виконуються. Використання коалгебраїчних методів дозволяє формально довести, що інваріанти дотримуються у всіх можливих станах системи.

4. Природна підтримка континуальних і дискретних систем. Коалгебраїчні методи добре підходять для моделювання як континуальних, так і дискретних систем. Це дозволяє використовувати їх для широкого спектра застосувань, від цифрових схем і програмного забезпечення до фізичних процесів і систем управління. Коалгебраїчна специфікація може описувати поведінку систем з нескінченними станами, що робить її універсальним інструментом для моделювання різномані-

тних систем.

5. Можливість інтеграції з іншими формальними методами. Коалгебраїчні специфікації можуть бути інтегровані з іншими формальними методами, такими як алгебраїчні специфікації та модельна перевірка (model checking) [6]. Це забезпечує додаткові можливості для формального аналізу і верифікації складних систем. Наприклад, алгебраїчні методи можуть бути використані для опису статичних аспектів системи, в той час як коалгебраїчні методи використовуються для динамічних аспектів, що забезпечує комплексний підхід до моделювання і верифікації.

Таким чином, коалгебраїчні специфікації можуть бути інтегровані з іншими формальними методами, такими як алгебраїчні специфікації та модельна перевірка (model checking). Це забезпечує додаткові можливості для формального аналізу і верифікації складних систем. Наприклад, алгебраїчні методи можуть бути використані для опису статичних аспектів системи, в той час як коалгебраїчні методи використовуються для динамічних аспектів, що забезпечує комплексний підхід до моделювання і верифікації.

Окремо слід зазначити, що подібні формальні методи, які базуються на використання теорії універсальних коалгебр мають сенс при роботі над специфікацією систем, які неможливо адекватно формалізувати за допомогою однієї моделі. До таких систем можна віднести кіберфізичні.

Кіберфізичні системи — це інтегровані системи, в яких відбувається тісна взаємодія між фізичними компонентами та кібернетичними (обчислювальними та комунікаційними) елементами [7]. В таких системах фізичні процеси контролюються комп'ютерними алгоритмами, які, у свою чергу, взаємодіють з мережею інтернет та іншими цифровими системами. Кіберфізичні системи складаються з трьох основних компонентів:

1. Фізичні компоненти: Це сенсори, актуатори, і різні фізичні пристрої, які взаємодіють із зовнішнім середовищем. Вони збирають дані про

фізичні процеси, які потім обробляються кібернетичними компонентами.

2. Кібернетичні компоненти: Це обчислювальні елементи, програмне забезпечення та алгоритми, які аналізують дані, приймають рішення та керують фізичними компонентами. Вони забезпечують обробку і зберігання даних, а також їх передачу через мережу.

3. Комунікаційні компоненти: Це мережеві інтерфейси та протоколи, які забезпечують передачу даних між фізичними і кібернетичними компонентами.

Одна з основних проблем специфікації кіберфізичних систем полягає в їх складності. Подібні системи включають в себе безліч різнорідних компонентів, які взаємодіють між собою в реальному часі [8]. Ці компоненти можуть бути механічними, електронними, програмними або навіть біологічними. Координація їх роботи вимагає розробки складних моделей і алгоритмів, які здатні врахувати всі можливі сценарії поведінки системи. Наприклад, в автономних транспортних системах необхідно врахувати поведінку транспортних засобів, дорожню інфраструктуру, пішоходів та інші фактори, що впливають на безпеку та ефективність руху.

Таким чином, подібні системи неможливо специфікувати використовуючи лише одну формальну модель. Це пов'язано з тим, що як було зазначено, елементи цієї системи занадто різні зі їх природою. Теорія універсальних коалгебр у свою чергу дає можливість використовувати композицію моделей да допомогою модульного підходу до моделювання.

З усього вищевикладеного можна зробити висновок, що формальні методи специфікації програмних продуктів та систем є підходом, що дозволяє забезпечити високу надійність системи та мінімізувати фінансові затрати на виправлення помилок у системі, у випадку, якщо вони виникнуть. Використання універсальних коалгебр для специфікації поведінки програмних продуктів та систем має низку переваг над іншими формальними методами, зокрема простоту у моделюванні динамічних дискретних систем, що в

свою чергу дає можливість зменшити затрати на створення специфікації для подібних типів систем. Отже, науково-прикладна задача розвинення методологій та підходів до специфікації поведінки динамічних дискретних систем, побудованих на основі універсальних коалгебр є актуальною.

Зв'язок роботи з науковими програмами, темами, планами. Дисертаційна робота виконана відповідно до:

1. Закону України № 2623-III "Про пріоритетні напрями розвитку науки і техніки"редакція від 20.02.2021 р., зокрема за напрямом "Інформаційні та комунікаційні технології";

2. Закону України № 3715-VI "Про пріоритетні напрями інноваційної діяльності в Україні"редакція від 05.12.2012 р., зокрема за напрямом "Розвиток сучасних інформаційних, комунікаційних технологій, робототехніки";

Тематика дисертаційної роботи пов'язана з дослідженнями

1. Участь у НДР «Integrated rail freight optimisation in Ukraine: Railway sleepers, rolling stock and logistics» (ДР № 0123U102700), у якості виконавця.

Наукові дослідження, викладені в дисертації, виконані згідно з напрямом наукової роботи кафедри Теоритичної і прикладної інформатики Харківського національного університету ім. В.Н.Каразіна.

Мета та задачі дослідження. Метою дисертаційної роботи є удосконалення уніфікованих методів специфікації та аналізу дискретних динамічних систем різного типу включно з розподіленими за рахунок використання теорії універсальних коалгебр.

Для досягнення зазначеної мети були поставлені та вирішені наступні задачі:

1. Аналіз сучасного стану методологій формальної специфікації програмних продуктів, в тому числі з використанням універсальних коалгебр.

2. Розробка коалгебрачних моделей для категорій дискретних динамічних систем, що породжують вихідні данні.

3. Дослідження властивостей ендофунктора дискретної динамічної системи з метою з'ясування факту наявності відповідної фінальної системи.

4. Використання монади Джирі для забезпечення рандомізації дискретних динамічних систем з виходами з метою формального визначення рандомних систем.

5. З'ясування існування фінальної системи для класу рандомних систем і її обчислення у разі існування.

6. Аналіз поведінки рандомної системи як композиції системи з виходами та монади Джірі.

7. Удосконалення методу синтезу моделей для динамічного аналізу складних систем шляхом використання техніки універсальних коалгебр.

Об'єктом дослідження є процеси специфікації та аналізу поведінки розподілених обчислювальних систем різної природи, як детермінованих так і рандомних.

Предметом дослідження є колагебраїчні моделі дискретних динамічних систем та методи їх аналізу.

Методи дослідження. Дослідження виконано із застосуванням наступних принципів та методів

1. Принципів теорії систем, що є загальною теоретичною базою дослідження.

2. Теорії категорій, що забезпечує універсальну формальну мову для дослідження систем різної природи.

3. Теорії універсальних коалгебр, що дозволяє моделювати поведінку дискретних динамічних систем різного типу включаючи розподілені в рамках єдиного фреймворку.

4. Фінальної семантики, яка визначає сенс властивостей і відношень притаманих всім системам заданого типу.

5. Методу коіндукції, який дозволяє доводити визначати та доводити властивості фінальної системи.

Наукова новизна отриманих результатів полягає в наступному

1. **Вперше запропоновано** формалізацію операції рандомізації дискретної динамічної системи шляхом лівої композиції її ендофунктора з монадою Джірі.

2. **Вперше сформульовано** та доведено достатню умову для збереження ендофунктором категорії множин слабких декартових квадратів, що дозволяє встановити факт існування фінальної системи.

3. **Дістала подальшого** розвитку техніка обчислення фінальної системи певного типу шляхом використанням методу коіндукції у разі доведеного факту існування фінальної системи.

4. **Дістав подальшого** розвитку метод синтезу моделей для динамічного аналізу (імітаційного моделювання) складних систем з використанням техніки універсальних коалгебр.

Особистий внесок здобувача

Дисертаційне дослідження виконано здобувачем самостійно, усі сформульовані в ньому положення та висновки з рекомендаціями обґрунтовані на основі особистих досліджень автора. Для аргументації окремих положень використані праці інших науковців, на які зроблені посилання. В індивідуальних наукових працях застосовано лише авторські ідеї та розробки.

Аспірант брав активну участь у наукових дискусіях, семінарах, підготовці наукових статей, опублікованих за темою дисертації, успішно доповідав результати досліджень на міжнародних конференціях.

У публікації [9] здобувач визначив моделі для формалізації роботи розповсюджених типів дискретних динамічних систем з використанням універсальних коалгебр.

У публікації [10] здобувач розробив теоретичні засади існування фінальної системи для обраного типу дискретних динамічних систем. Здобувач визначив ендофунктор, який було успішно використано для специфікації та аналізу загальних дискретних систем, які можуть бути використані для моделювання розподілених, вбудованих і кіберфізичних систем.

У публікації [11] здобувач розробив підходи до впровадження стохастичності до детермінованої моделі за допомогою монади вірогіднісного розподілу Джирі. Таким чином здобувачем було запропоновано процедуру рандомізації, яка використовує ендофунктор скінченних розподілів як інструмент для побудови випадкової системи на основі детермінованої системи.

У публікації [12] здобувач розробив імітаційну моделі залізниці з оглядом на її природу, яку можна охарактеризувати як розподілену. Це дозволило визначити макропараметри транспортного процесу, які найбільше впливають на час перебування вагонної відправки в системі сортування.

У публікації [13] здобувач навів теоретичні засади для використання декартових квадратів для доведення існування фінальної системи для побудованих моделей динамічних систем.

У всіх назначених публікаціях здобувач написав значну частину тексту та перекладав його.

Практичне значення отриманих результатів

Метод побудови імітаційної моделі складних динамічних систем з використанням універсальних коалгебр та враховуючи достатню умову збереження ендофунктором категорії множин слабких декартових квадратів, що дозволяє встановити факт існування фінальної коалгебри було викори-

стано для моделювання роботи сортувальної станції з метою моніторингу руху вагонопотоків для забезпечення прогнозного часу доставки вантажів на полігоні регіональної філії "Південна залізниця" АТ "Укрзалізниця" (акт впровадження від 12 грудня 2023р.), що дозволило абезпечити точність прогнозування ETD сортувальних станцій на розгалужених полігонах залізничної мережі на рівні, близькому до 90 % з похибкою 4 - 9 %. Практично було встановлено, що методи імітаційного моделювання з використанням універсальних коалгебр дають змогу зменшити потенційну кількість помилок при розробці систем моніторингу руху вагонопотоків до 15 %. Це може дозволити зменшити показник середнього часу простою вагона - транзит з переробкою до 10 % від існуючих показників.

Так само результати дисертаційного дослідження було впроваджено на виробництві ПАТ "Турбогаз". Використання методик коалгебраїчної специфікації на етапі проектування турбодетандерної техніки дало можливість підвищити надійність техніки, що виготовляється (акт впровадження від 27 грудня 2023р.)

Отримані в процесі дослідження теоретичні положення, моделі й методи впроваджено у навчальний процес УкрДУЗТ (акт від 14 грудня 2023р.) на факультеті "Управління процесами перевезень" кафедри "Управління експлуатаційною роботою" у дисциплінах "Інформаційні технології в управлінні міжнародними перевезеннями", "Сучасні інформаційні технології в управлінні залізничними підрозділами" та "Управління експлуатаційною роботою" при вивченні систем та підходів до проєктування систем моніторингу руху, зокрема при вивченні

1. принципів планування роботи станції використовувався метод побудови імітаційної моделі роботи сортувальної станції з використанням універсальних коалгебр

2. моніторингу руху вагонопотоків використовувався метод прогнозування очікуваного часу відправлення (ETD) вантажної відправки на сортувальній станції з використанням методу динамічного аналізу

(імітаційного моделювання) складних систем з використанням універсальних коалгебр

А також для підготовки до дипломування бакалаврів і магістрів освітньо-професійного рівня та магістрів освітньо-наукового рівня на факультеті “Управління процесами перевезень” за освітніми програмами “Організація перевезень і управління на транспорті”, “Організація міжнародних перевезень” у Українському державному університеті залізничного транспорту.

Апробація результатів дисертації. Основні теоретичні положення, висновки і пропозиції, які містяться в дисертації, обговорювалися та були затвердженні на засіданнях кафедри теоретичної та прикладної інформатики Харківського національного університету імені В.Н. Каразіна. Ключові положення дослідження оприлюднені у доповідях на науково-технічних конференціях всеукраїнського та міжнародного рівнів (2020–2024 роки).

1. 17th International Conference on ICT in Education, Research and Industrial Applications. Integration, Harmonization and Knowledge TransferAt: Kherson, Ukraine: Volume I: Main Conference, PhD Symposium, and Posters (Україна, м.Херсон, 2021р.)

2. 13th International Conference on Dependable Systems, Services and Technologies (DESSERT) (Греція, м.Афіни 2023р.)

Публікації. Основні теоретичні положення і висновки дисертації викладені у 5 наукових працях, з яких 1 стаття у наукових фахових виданнях України та ті, що входять до міжнародних наукометричних баз [9-12] та 3 тез наукових доповідей [9-11].

Структура та обсяг дисертації. Дисертаційна робота складається зі вступу, чотирьох розділів, висновків, списку використаних джерел і 2 додатків. Загальний обсяг дисертації становить 138 сторінки: у тому числі анотації на 10 сторінках, зміст на 3 сторінках, основний текст на 101 сторінках, список використаних джерел із 72 найменувань на 11 сторінках та два додатки на 8 сторінках. Робота містить 18 діаграм та 10 рисунків.

РОЗДІЛ 1. АНАЛІЗ ПІДХОДІВ ДО ПІДВИЩЕННЯ ЯКОСТІ ПРОЦЕСУ СПЕЦИФІКАЦІЇ ОБМЕЖЕНЬ РОЗПОДІЛЕНИХ СИСТЕМ

1.1 Аналіз проблематики специфікації поведінки систем на етапі проєктування та її актуальність

Специфікація поведінки системи є одним з ключових аспектів у розробці програмного забезпечення та систем, що дозволяє визначити як система повинна поводитися у відповідь на різні вхідні дані та взаємодії. Ця специфікація надає чітке уявлення про динамічні аспекти системи, що є критичним для забезпечення її коректності та надійності [14]. Специфікація поведінки системи описує динамічну поведінку системи у вигляді реакцій на події, входи і зміну станів. Це включає в себе створення моделі станів і переходів, розробку протоколів взаємодії та опис реакцій системи на події.

Модель станів і переходів системи є важливим інструментом для специфікації поведінки систем. Вони дозволяють описати, як система переходить між різними станами у відповідь на зовнішні та внутрішні події. Основними поняттями є Стан (відображає конкретний момент часу або конфігурацію системи), Подія(зовнішня або внутрішня дія, яка може змінити стан системи) та Перехід (процес зміни одного стану на інший, що відбувається у відповідь на подію). Є декілька методів для того щоб реалізувати модель станів та переходв, серед яких можна виділити діаграми станів UML (Unified Modeling Language) [15] та формальна модель з використанням скінчених автоматів.

Протоколи взаємодії описують правила та порядок обміну інформацією між різними компонентами системи або між системою та її зовнішнім середовищем. Протоколи взаємодії забезпечують чіткість і однозначність у комунікації, що критично важливо для розробки складних програмних

систем. З їх використанням визначають набір повідомлень, які можуть бути передані між компонентами системи, порядок їх передачі, а також умови та дії, що асоціюються з цими повідомленнями. Вони гарантують, що всі учасники взаємодії слідують встановленим правилам, забезпечуючи узгодженість і передбачуваність поведінки системи. Основними елементами протоколів взаємодії є Повідомлення (основні одиниці обміну інформацією між компонентами системи), Ролі (Визначають учасників протоколу та їхні функції), Умови (Логічні вирази, що визначають, коли і як можуть відбуватися певні взаємодії) та Дії (Операції, які виконуються у відповідь на певні повідомлення).

Значення специфікації та вигода її впровадження на етапі проектування системи та передпроектного аналізу полягає у наступному [16]

1. Формалізація поведінки: Моделі дозволяють формально описати динамічну поведінку системи, що сприяє точному розумінню та аналізу

2. Верифікація та валідація: Можливість формальної перевірки коректності системи та виявлення помилок на ранніх етапах розробки

3. Покращення комунікації: Забезпечують спільне розуміння між розробниками, замовниками та іншими учасниками проєкту

4. Спрощення тестування: Чіткі моделі полегшують розробку тестових сценаріїв і перевірку системи на відповідність специфікації

5. Гарантія сумісності: Протоколи забезпечують узгодженість і сумісність між різними компонентами системи

6. Полегшення інтеграції: Визначені правила і формати повідомлень спрощують інтеграцію з іншими системами

7. Забезпечення надійності та безпеки: Протоколи включають механізми для обробки помилок і захисту даних

8. Покращення продуктивності: Оптимальні протоколи можуть зменшити затримки і підвищити ефективність передачі даних

Таким чином, специфікація на етапі проєктування системи відіграє критично важливу роль, забезпечуючи структуроване та чітке визначення вимог, функціональності та обмежень системи [17]. Вона служить основою для всіх наступних етапів розробки, включаючи реалізацію, тестування та підтримку. Специфікація дозволяє узгодити очікування між замовниками, розробниками та іншими зацікавленими сторонами, зменшуючи ризик непорозумінь та помилок. Наявність такої документації залишається актуальною протягом всього життєвого циклу системи. Вона забезпечує основу для розширення системи, підтримки та обслуговування та навчання нових співробітників.

Однак, процес створення специфікації часто стикається з численними проблемами, які можуть вплинути на якість кінцевого продукту. Ці проблеми виникають через складність, неоднозначність, змінність вимог, а також технічні та організаційні бар'єри [18]. Серед ключових проблем можна виділити наступні:

1. Неоднозначність і неповнота вимог. Термінологія та формулювання вимог можуть бути інтерпретовані по-різному різними учасниками проекту. Це призводить до різних розумінь вимог, що впливає на узгодженість проекту. З іншого боку, вимоги можуть бути неповними або недостатньо детальними, що створює прогалини в розумінні функціональності системи. Це може призвести до помилок у розробці та інтеграції компонентів.

2. Змінність вимог. Вимоги до системи часто змінюються протягом життєвого циклу розробки через зміни в бізнес-процесах, зовнішніх умовах або нових технологічних можливостях. Це ускладнює підтримку актуальності специфікації та вимагає постійного оновлення документації.

3. Складність і масштабність систем. Сучасні програмні системи є висококонфігурованими та інтегрованими з різними іншими системами, що ускладнює розробку та підтримку специфікацій. Велика кількість компонентів і їх взаємодія створюють додаткові складнощі у визначенні та управлінні вимогами.

4. Комунікаційні бар'єри. Недостатня комунікація між зацікавленими сторонами проекту може призвести до неправильного розуміння вимог. Різні підходи до розробки, культурні та мовні відмінності також можуть впливати на якість специфікацій.

5. Формалізація специфікацій. Вибір між текстовими (неформальними) та формальними методами специфікації є критичним. Текстові методи часто є зрозумілими, але можуть бути неоднозначними, тоді як формальні методи забезпечують точність, але є складними для розуміння та використання широкою аудиторією.

6. Верифікація і валідація. Перевірка відповідності специфікації реальним вимогам користувачів та її правильності є складним завданням. Валідація включає переконання, що специфікація відповідає потребам замовника, а верифікація - що система відповідає специфікації.

7. Інтеграція з існуючими системами. Специфікація повинна враховувати сумісність з існуючими системами та інтерфейсами, що може бути складним через різні стандарти, протоколи та архітектурні підходи.

8. Документування нефункціональних вимог. Нефункціональні вимоги, такі як продуктивність, безпека, масштабованість, часто є важко визначуваними та вимірюваними, що створює проблеми для їх адекватної специфікації та тестування.

Окремо слід зазначити, що проблеми, пов'язані зі специфікацією поведінки систем та пошук методик до покращення процессів проектування, залишаються актуальними через зростаючу складність і масштабність

програмного забезпечення [19]. Неправильне або неповне визначення вимог може призвести до значних витрат на виправлення помилок на пізніх стадіях розробки, зниження якості системи та незадоволення користувачів.

Актуальність проблеми специфікації на сучасному етапі є надзвичайно важливою в основному через зростаючу складність програмних систем, швидкий розвиток технологій, постійно змінювані вимоги користувачів та підвищені очікування щодо якості програмного забезпечення. В умовах швидких змін на ринку та високої конкуренції, правильна та точна специфікація є ключовим фактором успішної розробки програмного забезпечення.

Сучасні програмні системи стають все більш складними та інтегрованими. Це створює додаткові виклики для розробників щодо визначення та документування вимог. Наприклад, системи Інтернету речей (IoT) [20], хмарні сервіси та розподілені системи вимагають врахування великої кількості взаємозв'язків та залежностей між компонентами. Крім того, технологічний прогрес призводить до постійного оновлення інструментів, платформ і методологій розробки програмного забезпечення. Вимоги до систем часто змінюються на пізніх етапах проекту, що ускладнює процес специфікації та реалізації.

Невизначеність та змінність бізнес-вимог також створюють виклики для специфікації, оскільки необхідно швидко адаптуватися до нових умов. Специфікація є основним засобом комунікації між різними зацікавленими сторонами, включаючи розробників, тестувальників, менеджерів проекту та замовників. Невідповідність розуміння вимог може призвести до значних помилок і витрат. У глобальних командах, де учасники можуть працювати в різних часових поясах та культурних контекстах, чітка та зрозуміла специфікація має критичне значення для узгодженості роботи [21].

Споживачі очікують високої якості, надійності та безпеки програмного забезпечення. Недоліки у специфікації можуть призвести до критичних помилок, які важко виправити на пізніх стадіях розробки [22]. Відповідність

нормативним вимогам і стандартам (наприклад, у медичних або фінансових системах) вимагає точних і детальних специфікацій для забезпечення відповідності і аудиту. Сучасні методології, такі як Agile [23] та DevOps [24], передбачають ітеративний підхід до розробки, що вимагає постійного оновлення та вдосконалення специфікацій на кожному етапі проекту. Специфікації мають бути гнучкими та адаптивними, щоб швидко реагувати на зворотний зв'язок та зміни вимог.

Впровадження штучного інтелекту та автоматизації в процеси розробки програмного забезпечення вимагає нових підходів до специфікації, оскільки алгоритми навчання і прийняття рішень можуть мати складні та непередбачувані поведінки [25]. Формальні методи специфікації, такі як коалгебраїчні моделі, стають важливими для забезпечення точності та верифікації таких систем.

Специфікація поведінки систем є ключовим елементом у процесі розробки програмного забезпечення, яка визначає його успіх та відповідність вимогам користувачів. Актуальність цієї проблеми постійно зростає у зв'язку з ускладненням систем, швидкими змінами в технологіях та бізнес-вимогах, підвищеними очікуваннями щодо якості, а також необхідністю ефективної комунікації між зацікавленими сторонами. Розробникам і менеджерам проектів необхідно враховувати ці виклики та активно використовувати сучасні методи і засоби для забезпечення якісної та гнучкої специфікації.

Існує кілька методів та підходів до створення специфікацій, які можна поділити на неформальні, напівформальні та формальні.

Неформальні методи специфікації поведінки систем забезпечують простоту та зрозумілість для широкого кола учасників проекту, що робить їх зручними для початкових етапів розробки або для команд з обмеженими технічними знаннями [26]. Неформальні методи специфікації мають свої переваги та обмеження. Вони забезпечують зрозумілість і простоту, що є важливим для початкових етапів проекту та для комунікації між різними зацікавленими сторонами. Однак, їх неоднозначність і можливість пропу-

ків роблять їх менш придатними для складних та критично важливих систем, де потрібна висока точність і однозначність. У таких випадках, неформальні методи можуть використовуватися у поєднанні з напівформальними та формальними методами для забезпечення повноти та точності специфікації. До неформальних методів можна віднести:

1. Текстові описи. Це найбільш поширений і зрозумілий спосіб специфікації, де вимоги описуються у вигляді тексту. Неформальні описи легко зрозумілі широкому колу учасників проекту, але можуть бути неоднозначними і неповними.

2. User Stories. Короткі оповідання від імені користувачів, які описують бажану функціональність системи. Цей підхід часто використовується в методологіях Agile і забезпечує зрозумілість вимог для всіх зацікавлених сторін [27].

3. Сценарії використання (Use Cases). Метод, що описує взаємодію між користувачем і системою, спрямований на досягнення певної мети. Сценарії використання деталізують різні шляхи, якими користувач може взаємодіяти з системою [28].

Напівформальні методи специфікації поведінки систем поєднують елементи як неформальних, так і формальних методів, забезпечуючи зрозумілість і структурованість, необхідні для ефективного проектування складних систем [29]. Вони використовують діаграми, моделі та інші візуальні засоби для опису системи, що робить їх придатними для використання у командах з різним рівнем технічної підготовки. Напівформальні методи специфікації забезпечують кращу структурованість та точність порівняно з неформальними методами, зберігаючи при цьому зрозумілість і простоту використання. Вони дозволяють створювати візуальні моделі, які легко читати і розуміти, що сприяє покращенню комунікації та співпраці між членами команди. Використання напівформальних методів є ефективним компромісом між точністю формальних методів та гнучкістю і простотою

неформальних методів, що робить їх особливо корисними для складних проектів, які потребують детального і структурованого підходу до специфікації поведінки систем. До напівформальних методів можна віднести:

1. UML (Unified Modeling Language). Використовується для візуалізації, специфікації, конструювання та документування компонентів програмних систем. UML включає діаграми класів, діаграми станів, діаграми діяльності та інші, що допомагають моделювати поведінку системи [30] [31].

2. DFD (Data Flow Diagrams). Використовується для представлення потоків даних в системі та обробки цих даних. DFD допомагає зрозуміти, як дані перетікають через систему і як вони змінюються [32].

Формальні методи специфікації поведінки систем є математично обґрунтованими підходами, що забезпечують точність і однозначність опису програмних систем [33]. Вони використовують формальні мови і моделі для визначення вимог і поведінки систем, дозволяючи здійснювати формальну верифікацію і доведення властивостей системи. Формальні методи специфікації мають кілька значних переваг. Вони забезпечують високу точність і однозначність опису, що знижує ймовірність помилок на стадії проектування. Формальні методи дозволяють здійснювати формальну верифікацію специфікацій, що сприяє виявленню і виправленню помилок на ранніх етапах розробки, зменшуючи таким чином витрати на їх виправлення у майбутньому. Крім того, формальні методи сприяють автоматизації процесу розробки, дозволяючи використовувати інструменти для автоматичної верифікації та генерації коду. Проте, використання формальних методів також має свої обмеження. Вони вимагають високого рівня математичної підготовки і спеціалізованих знань, що може бути бар'єром для їх широкого застосування. Крім того, процес формальної верифікації може бути трудомістким і затратним, особливо для великих і складних систем. Незважаючи на ці обмеження, формальні методи є потужним інструментом для забезпечення високої якості програмного забезпечення і знаходять широке

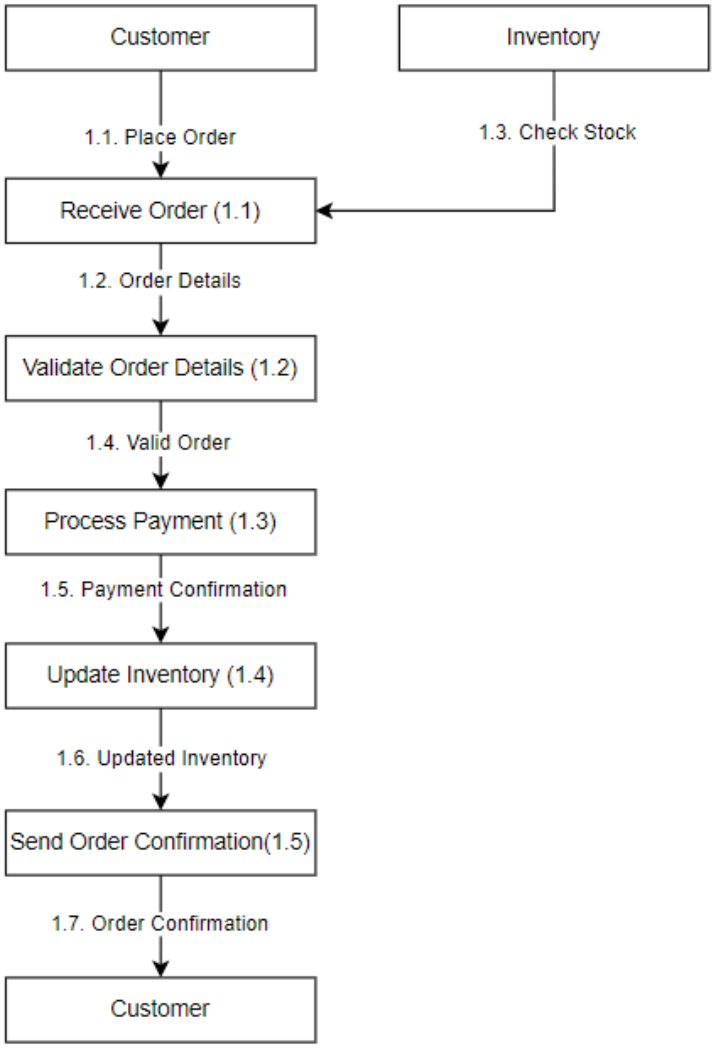


Рис. 1.1. Приклад використання Data Flow Diagrams

застосування в критично важливих системах, таких як авіоніка, медичні прилади та фінансові системи. До формальних методів можна віднести:

1. VDM (Vienna Development Method). Формальна методика специфікації, яка використовує математичні моделі для визначення поведінки програмних систем. VDM дозволяє здійснювати аналіз і верифікацію вимог [34].

2. Алгебраїчні специфікації. Використовуються для визначення поведінки абстрактних типів даних. Алгебраїчні специфікації забезпечують формальний підхід до визначення операцій над даними і їх властивостей [35].

3. Коалгебраїчні методи. Специфікація динамічних систем через коалгебри. Вони є зручними для опису систем, де поведінка визначається переходами між станами.

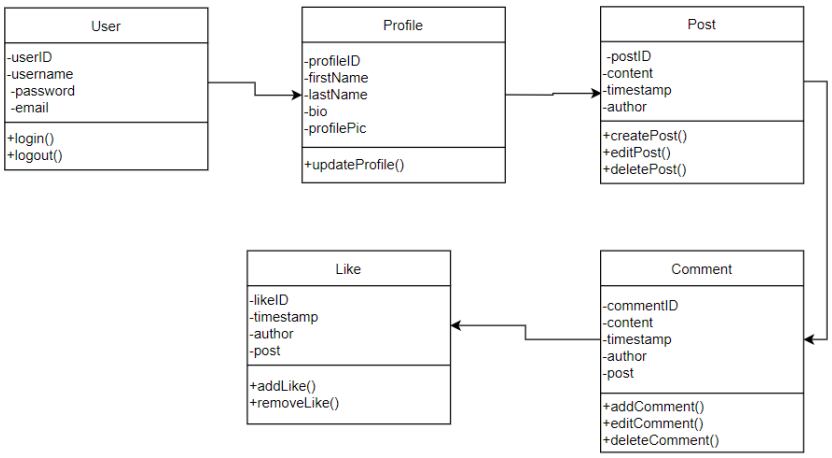


Рис. 1.2. Приклад використання UML

Вибір конкретного методу специфікації залежить від складності системи, вимог до точності і однозначності, а також від наявних ресурсів і рівня підготовки розробників. Неформальні методи підходять для початкових стадій проекту або для невеликих команд з обмеженими технічними знаннями. Напівформальні методи, такі як UML [36], є компромісом між зрозумілістю і структурованістю, тоді як формальні методи забезпечують високу точність і надійність, але вимагають значних зусиль на верифікацію і навчання.

1.2 Аналіз формальних методів специфікації поведінки систем. Обґрунтування доцільності їх використання

формальні методи специфікації, які відіграють ключову роль у забезпеченні точності, надійності та однозначності програмних систем. Вибір саме формальних методів має кілька суттєвих переваг, які зумовлюють їх актуальність і необхідність у сучасному процесі розробки складних та критично важливих систем. Вони базуються на використанні математичних

моделей і логіки, що забезпечує точне визначення вимог і поведінки систем. Це дозволяє уникнути неоднозначностей і протиріч, які часто виникають при використанні неформальних і напівформальних методів [37]. Завдяки математичній основі формальні методи дозволяють чітко формулювати специфікації, що знижує ймовірність помилок на етапі проектування.

Однією з основних переваг формальних методів є можливість формальної верифікації специфікацій. Це процес доведення правильності вимог і моделей системи з використанням математичних методів. Формальна верифікація дозволяє виявляти і виправляти помилки на ранніх стадіях розробки, що суттєво знижує витрати на їх усунення на пізніших етапах. Наприклад, використання методів формальної верифікації в авіоніці та медичних приладах дозволяє забезпечити високий рівень безпеки і надійності систем [38].

Формальні методи сприяють підвищенню загальної якості програмного забезпечення. Завдяки точним і однозначним специфікаціям розробники можуть створювати більш надійні і стійкі системи, які відповідають всім вимогам. Це особливо важливо для критично важливих систем, де помилки можуть призвести до серйозних наслідків. Використання формальних методів допомагає запобігти таким помилкам і забезпечити високий рівень якості програмного забезпечення. Формальні методи є особливо актуальними у галузях, де помилки можуть мати катастрофічні наслідки, таких як авіоніка, медичні прилади, ядерна енергетика та фінансові системи. В цих галузях вимоги до надійності і безпеки є надзвичайно високими, і формальні методи забезпечують необхідний рівень точності і надійності. Наприклад, авіаційні стандарти, такі як DO-178C [39], рекомендують використання формальних методів для забезпечення безпеки програмного забезпечення.

Таким чином, можна зробити висновок, що використання формальних методів специфікації поведінки систем є необхідним під час роботи зі складними системами, які мають підвищенні вимоги до надійності та безперебійності роботи, зокрема розподілені та кіберфізичні системи.

Формальні методи специфікації поведінки систем базуються на використанні математичних і логічних моделей для точного визначення властивостей та поведінки програмних систем. Кожен з методів має свої унікальні підходи та інструменти, які допомагають у формалізації вимог і їх подальшій верифікації [40]. Серед найпопулярніших формальних методів специфікації можна виділити наступні:

1. Мова формальної специфікації Z
2. Метод VDM (Vienna Development Method)
3. Алгебраїчні специфікації
4. Коалгебраїчні специфікації

Мова формальної специфікації Z (вимовляється як "зед") є одним з найвідоміших і найчастіше використовуваних формальних методів для опису і моделювання програмних систем [41]. Розроблена в Оксфордському університеті, ця мова використовується для створення точних і формально перевірених специфікацій, що забезпечує високу надійність і коректність програмних систем.

Мова Z базується на теорії множин і логіці першого порядку, що дозволяє створювати чіткі і формально обґрунтовані моделі систем. Вона використовує математичні нотації для опису станів системи, операцій над цими станами, а також їх властивостей. Основним елементом мови Z є схема (schema), яка використовується для групування пов'язаних між собою змінних і предикатів, що описують стан системи або поведінку окремих її частин.

Схема в мові Z має дві частини: деклараційну частину, що визначає змінні та їх типи, і предикатну частину, що задає обмеження або властивості цих змінних. Схеми дозволяють моделювати різні стани системи та описувати переходи між ними у вигляді операцій. Наприклад, схема для опису банківського рахунку може містити змінні для поточного балансу та обмеження, що забезпечують позитивний баланс після кожної операції.

Мова Z також включає засоби для моделювання складних структур даних за допомогою теорії множин. Це дозволяє створювати абстрактні моделі, що відображають структури даних і їх взаємодію. Крім того, логіка першого порядку забезпечує інструменти для формулювання інваріантів та передумов операцій, що дозволяє здійснювати формальну верифікацію властивостей системи.

Однією з ключових переваг мови Z є можливість формальної верифікації специфікацій. Це дозволяє доводити правильність вимог і моделі системи, що знижує ризики помилок на ранніх стадіях розробки. Формальна верифікація включає перевірку коректності інваріантів, передумов і постумов операцій, що гарантує відповідність специфікації реальним вимогам.

Мова Z є особливо корисною для розробки критично важливих систем, де помилки можуть мати серйозні наслідки. Наприклад, вона широко використовується в авіоніці, медичних приладах і фінансових системах. Використання формальної специфікації на ранніх етапах проектування допомагає забезпечити високу надійність і безпеку системи, знижуючи витрати на виправлення помилок на пізніших етапах розробки.

Одним з важливих аспектів використання мови Z є підтримка інструментальних засобів для автоматизації процесу формальної верифікації. Існують різні інструменти, що підтримують синтаксичний аналіз, перевірку коректності специфікацій і генерацію тестів. Це сприяє підвищенню ефективності процесу розробки та забезпечує додаткові гарантії правильності специфікацій.

Незважаючи на всі переваги, мова Z має певні обмеження. Вона вимагає від розробників високого рівня математичної підготовки і спеціалізованих знань, що може бути бар'єром для широкого використання. Крім того, процес формальної верифікації може бути трудомістким і затратним, особливо для великих і складних систем. Проте, ці недоліки компенсуються високою надійністю і точністю специфікацій, що робить мову Z незамінним інструментом для розробки критично важливих програмних систем.

Мова формальної специфікації Z використовується для точного і недво-

значного опису програмних систем. Нижче наведено приклад, який демонструє використання мови Z для моделювання банківського рахунку. Цей приклад включає опис станів банківського рахунку та операцій, які можуть виконуватися над цим рахунком.

Опис стану банківського рахунку

```
BANK_ACCOUNT
[ BALANCE ]
balance : BALANCE
balance ≥ 0
```

У цьому прикладі схема *BankAccount* описує стан банківського рахунку. Змінна *balance* представляє поточний баланс рахунку. Умова $balance \geq 0$ є інваріантом, який гарантує, що баланс рахунку не може бути від'ємним.

Опис операції депонування

```
DEPOSIT
Δ BANK_ACCOUNT
amount? : ℕ
amount? > 0
balance' = balance + amount?
```

Операція *Deposit* описує процес внесення коштів на банківський рахунок. Вона використовує схему $\Delta BankAccount$, яка означає, що стан рахунку змінюється. Змінна *amount?* представляє суму, яку користувач хоче внести, і має тип цілих чисел. Умова $amount? > 0$ гарантує, що внесена сума має бути додатньою. Після виконання операції новий баланс *balance'* обчислюється як сума попереднього балансу і внесеної суми.

Опис операції зняття коштів

```
WITHDRAW
Δ BANK_ACCOUNT
amount? : ℕ
amount? > 0 ∧ amount? ≤ balance
```

balance' = balance - amount?

Операція *Withdraw* описує процес зняття коштів з банківського рахунку. Вона також використовує схему $\Delta BankAccount$, що означає зміну стану рахунку. Змінна *amount?* представляє суму, яку користувач хоче зняти, і має тип цілих чисел. Умова $amount? > 0$ гарантує, що знята сума має бути додатньою, а умова $amount? \leq balance$ забезпечує, що знята сума не перевищує поточний баланс. Після виконання операції новий баланс *balance'* обчислюється як різниця між попереднім балансом і знятою сумою.

Повна специфікація банківського рахунку

BANK_ACCOUNT

[BALANCE]

balance : BALANCE

balance $\geq$ 0

DEPOSIT

Δ BANK_ACCOUNT

amount? : $\mathbb{N}$

amount? $>$ 0

balance' = balance + amount?

WITHDRAW

Δ BANK_ACCOUNT

amount? : $\mathbb{N}$

amount? $>$ 0 $\land$ amount? $\leq$ balance

balance' = balance - amount?

У цій повній специфікації об'єднано опис стану банківського рахунку та операцій депонування, зняття коштів і перевірки балансу. Використовуючи мову Z, можливо формально описати властивості банківського рахунку та забезпечити їх правильність і узгодженість. Наведений приклад показує,

як мова формальної специфікації Z може бути використана для створення точних і формально верифікованих моделей програмних систем, що сприяє підвищенню їх надійності і коректності.

Метод VDM (Vienna Development Method) [42] є одним з найстаріших і найбільш розповсюджених формальних методів для розробки програмного забезпечення. Він був розроблений у 1970-х роках у Віденському університеті та розвивався у рамках проекту IBM Vienna Laboratory. Основна мета VDM полягає у забезпеченні точного і формального опису програмних систем для підвищення їх надійності і коректності.

VDM базується на використанні математичних моделей для формалізації специфікацій програмного забезпечення. Цей метод дозволяє розробникам створювати специфікації, що можуть бути формально перевірені на відповідність вимогам, зменшуючи таким чином ймовірність помилок на ранніх етапах розробки. Основні концепції VDM включають абстрактні типи даних, інваріанти, передумови і постумови, що використовуються для опису і верифікації властивостей системи.

Абстрактні типи даних (АТД) є центральною концепцією VDM. Вони використовуються для моделювання структур даних системи на абстрактному рівні, що дозволяє відокремити логічні аспекти даних від їх конкретної реалізації. АТД визначаються через інваріанти та операції, що дозволяє формалізувати вимоги до даних і операцій над ними. Інваріанти є логічними умовами, які завжди повинні виконуватися для коректного стану даних. Наприклад, інваріант може вимагати, щоб баланс банківського рахунку завжди був невід'ємним.

Передумови і постумови є ще одним важливим аспектом VDM. Передумови визначають умови, які повинні бути виконані перед виконанням операції, тоді як постумови описують умови, які повинні виконуватися після завершення операції. Це дозволяє формалізувати вимоги до операцій і забезпечити їх коректне виконання. Наприклад, передумова для операції зняття коштів може вимагати, щоб сума зняття не перевищувала поточного балансу, тоді як постумова гарантує, що новий баланс буде зменшений на

відповідну суму.

VDM також включає формальні методи верифікації, які дозволяють перевірити відповідність специфікацій вимогам і їх коректність. Це досягається шляхом формального доведення, що інваріанти, передумови і постумови виконуються для всіх можливих станів системи. Формальна верифікація забезпечує високу надійність і коректність програмного забезпечення, що особливо важливо для критично важливих систем, таких як авіоніка, медичні прилади та фінансові системи.

Інструментальна підтримка є важливою частиною VDM. Існують різні інструменти, що підтримують синтаксичний аналіз, перевірку коректності специфікацій і генерацію тестів. Ці інструменти автоматизують процес формальної верифікації і підвищують ефективність розробки. Вони дозволяють розробникам швидко виявляти і виправляти помилки на ранніх етапах розробки, що знижує загальні витрати і час на розробку.

Метод VDM був застосований у багатьох реальних проектах і продемонстрував свою ефективність у забезпеченні високої надійності і коректності програмного забезпечення. Він є одним з основних методів у формальній розробці і продовжує використовуватися в сучасних проектах для створення надійних і безпечних систем.

Розглянемо невеликий приклад, який демонструє основні принципи VDM, використовуючи просту модель банківського рахунку. В цьому прикладі буде описано банківський рахунок з операціями внесення (депонування) та зняття коштів. Специфікація буде включати абстрактний тип даних для моделювання стану рахунку та операцій над ним.

Опис стану банківського рахунку Почнемо з визначення стану банківського рахунку, який включає поточний баланс.

```
types
    Balance = nat
    Account :: balance : Balance
    inv mk_Account(balance) == balance >= 0
```

У цьому прикладі ми визначили абстрактний тип даних `Account`, що містить змінну `balance` типу `Balance`, яка є невід'ємним цілим числом (натуральним числом). Інваріант гарантує, що баланс рахунку завжди буде невід'ємним.

Опис операції депонування

Далі визначимо операцію депонування, яка збільшує баланс рахунку на певну суму.

```
operations
    deposit(amount : nat) ext wr acc : Account
    pre amount > 0
    post acc' = mk_Account(acc.balance + amount)
```

Операція `deposit` приймає вхідний параметр `amount` типу `nat` (натуральне число). Вона модифікує стан рахунку (`wr` означає write) і має передумову `pre`, що гарантує, що сума внеску є додатньою. Постумова `post` описує новий стан рахунку після виконання операції: новий баланс дорівнює сумі попереднього балансу та внесеної суми.

Опис операції зняття коштів

Тепер визначимо операцію зняття коштів, яка зменшує баланс рахунку на певну суму, якщо поточний баланс достатній.

```
operations
    withdraw(amount : nat) ext wr acc : Account
    pre amount > 0 and amount <= acc.balance
    post acc' = mk_Account(acc.balance - amount)
```

Операція `withdraw` приймає вхідний параметр `amount` типу `nat`. Вона також модифікує стан рахунку і має передумови: сума зняття повинна бути додатньою і не перевищувати поточний баланс рахунку. Постумова описує новий стан рахунку після виконання операції: новий баланс дорівнює різниці між попереднім балансом та знятою сумою.

Повна специфікація банківського рахунку у VDM

```
types
    Balance = nat
    Account :: balance : Balance
    inv mk_Account(balance) == balance >= 0

operations
    deposit(amount : nat) ext wr acc : Account
    pre amount > 0
    post acc' = mk_Account(acc.balance + amount)

    withdraw(amount : nat) ext wr acc : Account
    pre amount > 0 and amount <= acc.balance
    post acc' = mk_Account(acc.balance - amount)
```

Ця специфікація описує стан банківського рахунку та дві основні операції над ним: депонування та зняття коштів. Використання інваріантів, передумов і постумов гарантує коректність операцій і дозволяє формально перевіряти специфікацію.

Таким чином показано, що метод VDM забезпечує формальні засоби для опису та верифікації властивостей програмних систем. Використовуючи VDM, розробники можуть створювати специфікації, що можуть бути формально перевірені, зменшуючи ймовірність помилок і підвищуючи надійність програмного забезпечення. У нашому прикладі ми побачили, як метод VDM може бути використаний для моделювання банківського рахунку, формально описуючи його стан і операції над ним.

Алгебраїчні специфікації є формальним методом, що використовується для опису та аналізу програмних систем. Вони базуються на математичних поняттях алгебри і дозволяють формально визначати абстрактні типи даних та операції над ними через систему рівнянь і логічних формул. Цей метод використовує алгебраїчні структури для моделювання поведінки програмних компонентів, забезпечуючи точний і формальний опис їх

властивостей.

Алгебраїчна специфікація визначає абстрактні типи даних за допомогою множин (сортів) і операцій. Сорти представляють різні категорії даних, наприклад, цілі числа, рядки або користувацькі типи даних, такі як списки або множини. Операції визначають функції або процедури, які можна виконувати над цими даними, і включають визначення їх сигнатур (аргументів і результатів).

Основні компоненти алгебраїчної специфікації включають сигнатуру, аксіоми і моделі. Сигнатура описує сорти і операції, тоді як аксіоми визначають властивості операцій у вигляді рівнянь або логічних формул. Моделі є конкретними реалізаціями специфікації, які задовольняють всі аксіоми.

Сигнатура містить визначення всіх сортів і операцій, що використовуються у специфікації. Наприклад, для абстрактного типу даних "черга"можна визначити сорти "Queue"і "Element а також операції "enqueue "dequeue "front"і "isEmpty". Ці операції матимуть свої сигнатури, що визначають типи їх аргументів і тип результату.

Аксіоми визначають властивості операцій у вигляді рівнянь, що гарантують їх коректність. Наприклад, для операції "enqueue"у черзі можна визначити аксіому, що операція додає елемент у кінець черги, а операція "dequeue"видаляє елемент з початку черги. Аксіоми формулюються таким чином, щоб забезпечити логічну узгодженість і правильність всіх операцій над абстрактним типом даних.

Моделі є конкретними інтерпретаціями специфікації, які задовольняють всі аксіоми. Наприклад, модель черги може бути реалізована за допомогою списків або масивів, де операції "enqueue"і "dequeue"реалізуються відповідними вставками і видаленнями елементів. Це дозволяє перевіряти, чи відповідає реалізація формальному опису, забезпечуючи правильність і надійність програмного забезпечення.

Алгебраїчні специфікації мають кілька важливих переваг. Вони забезпечують точний і недвозначний опис системи, знижуючи ймовірність помилок на ранніх етапах розробки. Завдяки модульності, алгебраїчні специфі-

кації дозволяють описувати систему як набір незалежних модулів, кожен з яких можна розробляти і перевіряти окремо. Аксіоми і рівняння використовуються для формальної верифікації коректності реалізації, що підвищує надійність системи. Абстракція відокремлює логічну структуру даних від їх конкретної реалізації, що полегшує зміну реалізації без зміни специфікації.

Алгебраїчні специфікації широко використовуються в різних галузях програмного забезпечення, включаючи системи баз даних, компілятори, операційні системи та інші критично важливі системи. Вони особливо корисні в контексті систем з високими вимогами до надійності і безпеки, де формальна верифікація є обов'язковою [43].

Наведемо приклад використання алгебраїчної специфікації для банківського рахунку з основними операціями депонування, зняття коштів та перевірки балансу.

Сигнатура

Спочатку визначимо сигнатуру, яка описує типи і операції

```
sorts Account, Amount, Balance
operations
  create: -> Account
  // створює новий банківський рахунок
  deposit: Account x Amount -> Account
  // депонує певну суму на рахунок
  withdraw: Account x Amount -> Account
  // знімає певну суму з рахунку
  get_balance: Account -> Balance
  //  повертає поточний баланс рахунку
  is_overdrawn: Account -> Bool
  // перевіряє, чи є рахунок перевищеним
```

Аксіоми

Тепер визначимо аксіоми, які описують поведінку цих операцій:

```
axioms
  for all a: Account, amt: Amount, b: Balance
    get_balance(create()) = 0
    // новостворений рахунок має нульовий баланс
    get_balance(deposit(a, amt)) = get_balance(a) + amt
    // після депонування суми amt, баланс рахунку
    //збільшується на цю суму
    get_balance(withdraw(a, amt)) = get_balance(a) - amt
    // після зняття суми amt, баланс рахунку
    // зменшується на цю суму
    is_overdrawn(create()) = false
    // новостворений рахунок не є перевищеним
    is_overdrawn(a) = (get_balance(a) < 0)
    // рахунок є перевищеним, якщо його баланс менший за нуль
```

Ця алгебраїчна специфікація може бути використана для формальної перевірки властивостей банківського рахунку і його операцій. Вона допомагає забезпечити коректність реалізації цих операцій і їх відповідність заданим вимогам.

Таким чином, можна зробити висновок, що формальні методи забезпечують математичну точність в описі поведінки програмних систем [44]. Вони дозволяють формально верифікувати властивості системи, доводячи їх правильність відносно заданих специфікацій. Це особливо важливо в критично важливих областях, таких як авіація, медицина, банківська справа, де помилки можуть призвести до катастрофічних наслідків. Формальні методи дозволяють виявляти помилки на ранніх етапах життєвого циклу програмного забезпечення. Оскільки специфікація є формальною, можливі недоліки у вимогах або проектуванні можна виявити ще до початку кодування. Це значно знижує витрати на виправлення помилок, оскільки, як відомо, чим раніше виявлено помилку, тим дешевше її виправити.

1.3 Аналіз підходів до специфікації поведінки систем з використанням універсальних коалгебр. Обґрунтування доцільності їх використання

Окремо серед формальних методів специфікації є сенс виділити коалгебраїчний метод специфікації, що є потужним формальним інструментом для моделювання та аналізу динамічних систем, які можна описати в термінах станів і переходів між ними . На відміну від традиційних алгебраїчних методів, які добре підходять для специфікації статичних структур даних, коалгебраїчні методи орієнтовані на опис поведінки систем, що змінюються з часом. Цей підхід особливо корисний для моделювання нескінченних структур та систем, які взаємодіють з оточенням.

Коалгебраїчні специфікації базуються на концепції коалгебр, які є дуальними до алгебр. У алгебраїчних специфікаціях основна увага приділяється операціям і їхнім аксіомам, що визначають, як побудувати нові елементи структури з існуючих. Коалгебраїчні специфікації, навпаки, фокусуються на спостережуваних властивостях системи і переходах між станами, що дозволяє описувати їхню поведінку в термінах спостережень та реакцій на зовнішні дії [45].

Основні поняття коалгебраїчної специфікації включають:

1. Станові простори (State Spaces): Станові простори описують множину всіх можливих станів системи. В коалгебраїчному контексті, стани не є статичними; вони визначаються через спостереження та переходи.

2. Спостереження (Observations): Спостереження визначають, які аспекти стану системи можуть бути виміряні або спостережені зовнішнім спостерігачем. Це можуть бути вихідні значення або внутрішні властивості, які можна побачити ззовні.

3. Переходи (Transitions): Переходи описують, як система змінюється з одного стану в інший у відповідь на зовнішні дії або внутрішні по-

дії. Коалгебраїчна специфікація моделює ці переходи за допомогою функцій переходів, що визначають новий стан системи на основі поточного стану та дії.

Коалгебраїчна специфікація системи формально визначається як коалгебра для функторів, що задають структуру станів та їхніх переходів. Нехай F — функтор, який задає типи спостережень і переходів для системи. Тоді коалгебра для F — це пара (S, γ), де S — множина станів, а $\gamma : S \to F(S)$ — функція, що визначає переходи та спостереження для кожного стану. Ця функція γ часто називається коалгебраїчним оператором.

Наприклад, розглянемо специфікацію нескінченного потоку значень. В коалгебраїчному підході, потік можна визначити як пару (спостереження, перехід), де спостереження видає перший елемент потоку, а перехід повертає залишок потоку. Формально, це можна описати як коалгебру для функтора $F(X) = \mathbb{A} \times X$, де $\mathbb{A}$ — тип елементів потоку.

Коалгебраїчні методи мають низку переваг:

1. Природний опис динаміки: Коалгебраїчні специфікації природно описують динамічні властивості систем, такі як послідовності дій та реакцій на події, що робить їх придатними для моделювання реактивних систем.

2. Моделювання нескінченних структур: Коалгебраїчний підхід легко справляється з моделюванням нескінченних структур, таких як потоки даних, що робить його корисним для систем, які повинні працювати безперервно або тривалий час.

3. Відповідність між моделюванням та реалізацією: Коалгебраїчні специфікації забезпечують чітку відповідність між формальною моделлю і реалізацією, що спрощує перевірку коректності реалізації щодо специфікації.

4. Автоматизація верифікації: Існують інструменти і методи для автоматичної верифікації коалгебраїчних специфікацій, що дозволяє фор-

мально доводити властивості системи або знаходити контрприклади для перевірки коректності специфікації.

Коалгебраїчні методи стали важливим інструментом у теоретичній інформатиці та інженерії програмного забезпечення завдяки їхній здатності описувати та аналізувати динамічні аспекти систем формально та математично строго. Вони знаходять застосування в широкому спектрі областей, включаючи верифікацію програмного забезпечення, аналіз систем управління, моделювання інтерактивних систем і багато іншого [46].

Для ілюстрації коалгебраїчної специфікації, розглянемо простий приклад моделювання банківського рахунку. Коалгебраїчний підхід дозволяє нам описати поведінку банківського рахунку у відповідь на різні дії, такі як депозити та зняття коштів, за допомогою станів і переходів.

Розглянемо коалгебраїчну специфікацію банківського рахунку. Нехай State — це невід'ємне ціле число, що представляє баланс рахунку, а Action — це дії, які можуть виконуватися над рахунком: deposit(x) і withdraw(y).

Функція переходу $\gamma : \text{State} \to (\mathbb{N} \times \text{State})$ визначається таким чином:

1. Для дії deposit(x):

$$\gamma(\text{balance}) = (\text{new\_balance}, \text{new\_balance})$$

де $\text{new\_balance} = \text{balance} + x$.

2. Для дії withdraw(y):

$$\gamma(\text{balance}) = \begin{cases} (\text{new\_balance}, \text{new\_balance}) & \text{if } y \leq \text{balance} \\ (\text{balance}, \text{balance}) & \text{if } y > \text{balance} \end{cases}$$

де $\text{new\_balance} = \text{balance} - y$.

Ця специфікація описує, як змінюється стан банківського рахунку при виконанні дій депозита і зняття коштів. Для дії депозиту баланс збільшується на відповідну суму. Для дії зняття коштів перевіряється, чи достатньо коштів на рахунку; якщо так, то баланс зменшується, інакше залишається незмінним. Такий підхід дозволяє чітко визначити поведінку системи

у відповідь на зовнішні дії та гарантувати її коректність через формальні методи верифікації.

Коалгебраїчні специфікації мають кілька значних переваг при моделюванні дискретних динамічних систем. Ці системи змінюють свій стан у дискретні моменти часу і включають реактивні системи, програмне забезпечення з подієвою логікою, а також апаратні компоненти. Нижче наведені основні переваги коалгебраїчних специфікацій для таких систем.

Коалгебраїчні специфікації дозволяють природно моделювати динаміку дискретних систем, оскільки вони зосереджуються на переходах між станами та спостереженнях. Дискретні динамічні системи визначаються набором дискретних станів і подіями, що викликають зміни стану. Коалгебраїчний підхід відображає цю структуру, описуючи систему у вигляді функції переходів, яка визначає, як система реагує на кожну подію, переходячи з одного стану до іншого. Це дозволяє легко зрозуміти та аналізувати поведінку системи.

Однією з ключових переваг коалгебраїчних специфікацій є їхня здатність працювати з нескінченними або дуже великими дискретними структурами. У дискретних динамічних системах часто виникає необхідність моделювання нескінченних послідовностей станів або подій. Коалгебраїчний підхід дозволяє ефективно описувати такі системи за допомогою нескінченних коалгебр, що дозволяє формально визначити та аналізувати їхню поведінку.

Коалгебраїчні специфікації підтримують формальну верифікацію дискретних динамічних систем. Завдяки математичній строгості коалгебр, можна використовувати формальні методи доведення для перевірки властивостей системи. Це включає перевірку коректності переходів між станами, відповідність специфікації та виявлення можливих помилок або несподіваних поведінок.

Формальна верифікація важлива для дискретних динамічних систем, особливо в контексті критичних систем, де помилки можуть призвести до серйозних наслідків. Коалгебраїчні специфікації забезпечують чітку відпо-

відність між моделлю та її реалізацією, що дозволяє легко перевіряти, чи відповідає реалізація початковій специфікації. Це спрощує процес розробки та знижує ймовірність помилок, забезпечуючи високу надійність системи [47].

Коалгебраїчні методи дозволяють автоматизувати аналіз та верифікацію специфікацій дискретних динамічних систем. Існує ряд інструментів та методів, які підтримують автоматичну генерацію перевірок властивостей, симуляцію та аналіз поведінки систем. Це зменшує трудомісткість та помилки при ручній перевірці, забезпечуючи ефективний та надійний підхід до верифікації.

Важливою перевагою коалгебраїчних специфікацій є їхній високий рівень абстракції. Вони дозволяють описувати поведінку дискретних динамічних систем на високому рівні, зосереджуючись на спостереженнях і переходах. Це робить моделі зрозумілими та легкими для аналізу, сприяє модульності та повторному використанню специфікацій.

Формальні методи коалгебраїчних специфікацій також забезпечують узгодженість і повноту опису дискретних динамічних систем. Всі можливі стани і переходи системи чітко визначені, що виключає суперечності та неповні описи. Це забезпечує надійність та точність специфікацій, що є критично важливим для розробки та підтримки складних систем.

Коалгебраїчні методи знаходять застосування у широкому спектрі областей, включаючи верифікацію програмного забезпечення, аналіз систем управління, моделювання інтерактивних систем і багато іншого [48]. Це робить їх універсальним інструментом для специфікації дискретних динамічних систем, які потребують формального та математично строгого підходу.

1.4 Постановка задачі дисертаційного дослідження

Специфікація поведінки системи є описом змін її станів в процесі функціонування. Цей документ формально визначає бажану поведінку про-

грамного або апаратного забезпечення і використовується для визначення та верифікації функцій і властивостей системи. Таке визначення є критичним для розробки, тестування та верифікації програмного забезпечення. Можна виділити два основних підходи до проведення аналізу системи, що проектується, з метою її специфікації, а саме - формальні та неформальні. Обидва підходи до специфікації програмного забезпечення мають свої переваги та недоліки. Вибір підходу залежить від вимог проекту, складності системи та необхідності формальної верифікації. Формальні методи в специфікації програмного забезпечення застосовуються в тих випадках, коли важлива висока точність, несуперечливість і можливість формальної верифікації системи. Окремо слід зазначити, що використання формальних методів специфікації програмних систем здешевлює процес розробки продуктів та зменшує потенційні витрати на усунення помилок. Серед усіх засобів формальної специфікації програмних продуктів слід виділити специфікацію з використанням універсальних коалгбер - алгебраїчних об'єктів дуальних універсальним алгебрам. Коалгебраїчна специфікація є потужним формальним методом, який використовується для моделювання динамічних систем.

Окремо слід зазначити, що коалгебраїчні специфікації можуть бути інтегровані з іншими формальними методами, такими як алгебраїчні специфікації та модельна перевірка (model checking). Це забезпечує додаткові можливості для формального аналізу і верифікації саме складних систем, які як правило не можуть бути описані в рамках однієї моделі. Наприклад, алгебраїчні методи можуть бути використані для опису статичних аспектів системи, в той час як коалгебраїчні методи використовуються для динамічних аспектів, що забезпечує комплексний підхід до моделювання і верифікації. Формальна специфікація динаміки системи особливо необхідна для розподілених систем, оскільки вони мають додаткову невизначеність спричинену недетермінованістю процесів обміну інформацією між компонентами системи.

З викладеного вище можна зробити висновок, що формальні методи

специфікації програмних продуктів та систем є підходом, що при правильному його використанні дозволяє забезпечити високу надійність системи та мінімізувати фінансові витрати на виправлення проєктних помилок, у випадку, якщо вони виникнуть. Використання універсальних коалгебр для специфікації поведінки програмних продуктів та систем має низку переваг над іншими формальними методами, зокрема, виразність моделювання дискретних динамічних систем, що в свою чергу дає можливість зменшити витрати на створення специфікації для подібних типів систем. Отже, науково-прикладна задача удосконалення моделей та методів специфікації поведінки дискретних динамічних систем, побудованих на основі універсальних коалгебр є актуальною.

Отже, метою дисертаційної роботи є удосконалення уніфікованих методів специфікації та аналізу дискретних динамічних систем різного типу включно з розподіленими за рахунок використання універсальних коалгебр. Основні завдання дисертаційного дослідження:

1. Аналіз сучасного стану методологій формальної специфікації програмних продуктів, в тому числі з використанням універсальних коалгебр.

2. Розробка коалгебрачних моделей для категорій дискретних динамічних систем, що породжують вихідні данні.

3. Дослідження властивостей ендофунктора дискретної динамічної системи з метою з'ясування факту наявності відповідної фінальної системи.

4. Використання монади Джирі для забезпечення рандомізації дискретних динамічних систем з виходами з метою формального визначення рандомних систем.

5. З'ясування існування фінальної системи для класу рандомних систем і її обчислення у разі існування.

6. Аналіз поведінки рандомної системи як композиції системи з виходами та монади Джірі.

7. Удосконалення методу синтезу моделей для динамічного аналізу складних систем шляхом використання техніки універсальних коалгебр.

Висновки до Розділу 1

В першому розділі розглядаються особливості процесу аналізу поведінки та специфікації систем з використанням різних методів. Наводиться стислий огляд публікацій за темою дисертації, опис методів специфікації програмного забезпечення та систем. Розглядаються переваги та недоліки формальних, напівформальних та неформальних методів специфікації. Описані і наведені приклади застосування усіх основних методів розробки специфікацій.

На основі результатів проведеного аналізу формулюється задача дисертаційного дослідження як удосконалення уніфікованих методів специфікації та аналізу дискретних динамічних систем різного типу включно з розподіленими за рахунок використання універсальних коалгебр, а саме

1. Аналіз сучасного стану методологій формальної специфікації програмних продуктів, в тому числі з використанням універсальних коалгебр.

2. Розробка коалгебрачних моделей для категорій дискретних динамічних систем, що породжують вихідні данні.

3. Дослідження властивостей ендофунктора дискретної динамічної системи з метою з'ясування факту наявності відповідної фінальної системи.

4. Використання монади Джирі для забезпечення рандомізації дискретних динамічних систем з виходами з метою формального визначення

рандомних систем.

5. З'ясування існування фінальної системи для класу рандомних систем і її обчислення у разі існування.

6. Аналіз поведінки рандомної системи як композиції системи з виходами та монади Джірі.

7. Удосконалення методу синтезу моделей для динамічного аналізу складних систем шляхом використання техніки універсальних коалгебр.

Список джерел, які використано у даному розділі, наведено у повному списку інформаційних джерел під номерами: [14-48].

РОЗДІЛ 2. ВИКОРИСТАННЯ УНІВЕРСАЛЬНИХ КОАЛГЕБР ДЛЯ МОДЕЛЮВАННЯ ТА ВИВЧЕННЯ ДИНАМІЧНИХ СИСТЕМ

Дискретна система – це система, в якій зміни стану відбуваються в дискретні моменти часу [49]. Стан системи визначається сукупністю всіх внутрішніх змінних, які описують її поточний конфігурацію. Кожен стан може мати відповідний набір характеристик, що змінюються в залежності від внутрішніх або зовнішніх впливів.

При дослідженні таких систем, основними об'єктами є переходи від поточних станів системи до станів, досяжних безпосередньо з поточних. Вони описують можливі зміни стану системи у відповідь на події або умови. Такий перехід можна описати як функцію, яка приймає поточний стан і подію (або вхідний сигнал) і повертає наступний стан.

Формально, перехідна функція може бути задана як

$$\delta : S \times \Sigma \to S$$

де S – множина станів, а Σ – множина подій. Для конкретного стану s і події σ,

$$\delta(s, \sigma) = s'$$

визначає новий стан s', до якого переходить система.

Окремо слід зазначити, що глибина знань про поточний та майбутні стани системи не є однаковими [50]. Зазвичай ми можемо визначити поточний стан системи доволі точно, у той час, як стан, доступний безпосередньо з поточного визначити однозначно не завжди можливо. Для цього існує багато причин, які можна пов'язати як з тим, що система може мати вади та через них не досягати наступного стану або через високий рівень розподіленості системи. Більш детально причини неоднозначності переходів між станами у системі будуть описані у Розділі 3.

2.1 Основні поняття теорії універсальних коалгебр

У цьому розділі буде наведено основні концепції теорії універсальних коалгебр, які використовуються для формальної специфікації поведінки дискретних динамічних систем. Також буде наведено неформальне пояснення коалгебрїчних концептів, що дозволить показати їх природність в сенсі використання саме для специфікації [51].

Припустимо, S є множиною можливих точних специфікацій системи, які називають станами (іноді чистими станами) системи. Тоді множина специфікацій станів, які досягаються безпосередньо з якогось стану з S, може бути специфікована за дпомогою не самого S, а скоріше за допомогою множини, що є застосквання до S деякою функцією від S. Позначимо цю функцію $\boldsymbol{F}$. Тоді можна розглядати клас систем, визначений цією функцією, як клас функцій $a : S \to \boldsymbol{F}S$. Кожна функція зазначеного класу виражає обмеження для можливих прямих наступників поточного стану.

Звичайно, можна розглядати клас систем, визначений функцією тотожності $\boldsymbol{F}$, тобто $\boldsymbol{F}S = S$ для всіх множин S. У цьому випадку ми маємо повністю детерміновані системи з повною інформацією про їхні стани [52]. Але така ситуація далека від реальною та має занадто високий рівень абстракції у першу чергу через те, що така детермінованість практично неможлива.

Основна концепція цієї загальної математичної основи виникла в теорії категорій і тепер відома як універсальна коалгебра. Далі в розділі будуть наведені основні визначення Теорії категорій та Теорії універсальних коалгебр, що будуть використанні у дослідженні.

Теорія категорій є важливою галуззю математики, яка надає універсальні інструменти для абстрактного та структурного аналізу математичних об'єктів [53]. Вона широко використовується в багатьох дисциплінах, включаючи алгебру, геометрію, теоретичну інформатику та теорію типів. У розділі будуть стисло описані основні концепції теорії категорій.

Означення 2.1 *Категорія $\mathcal{C}$ складається з множини об'єктів $Ob(\mathcal{C})$,*

множини морфізмів $Hom(\mathcal{C})$, та двох операцій: композиції морфізмів та відображення, що відображає кожен об'єкт в його тотожний морфізм.

Іншими словами категорію можна визначити наступним чином:

1. Для кожних двох об'єктів A і B існує множина $Hom(A, B)$, елементи якої називаються морфізмами з A в B.

2. Для кожних трьох об'єктів A, B і C існує операція композиції морфізмів $Hom(A, B) \times Hom(B, C) \to Hom(A, C)$, яка асоціативна.

3. Для кожного об'єкта A існує тотожний морфізм $id_A \in Hom(A, A)$, який є одиницею відносно композиції.

Означення 2.2 *Функтор*

$$F : \mathcal{C} \to \mathcal{D}$$

між двома категоріями $\mathcal{C}$ і $\mathcal{D}$ є відображенням, яке асоціює кожному об'єкту A категорії $\mathcal{C}$ об'єкт $F(A)$ категорії $\mathcal{D}$, і кожному морфізму

$$f : A \to B$$

морфізм $F(f) : F(A) \to F(B)$ так, що:

1. *$F(id_A) = id_{F(A)}$ для кожного об'єкта A.*

2. *$F(g \circ f) = F(g) \circ F(f)$ для кожних двох морфізмів $f : A \to B$ і $g : B \to C$.*

Натуральна перетворення $\eta : F \Rightarrow G$ між двома функторами $F, G : \mathcal{C} \to \mathcal{D}$ є сім'єю морфізмів $\eta_A : F(A) \to G(A)$, визначених для кожного об'єкта A категорії $\mathcal{C}$, таких що для кожного морфізму $f : A \to B$ в $\mathcal{C}$ наступна діаграма комутує:

$$\begin{array}{ccc} F(A) & \xrightarrow{F(f)} & F(B) \\ \eta_A \downarrow & & \downarrow \eta_B \\ G(A) & \xrightarrow{G(f)} & G(B) \end{array}$$

Означення 2.3 *Морфізм $f : A \to B$ в категорії $\mathcal{C}$ називається мономорфізмом, якщо для будь-яких двох морфізмів $g_1, g_2 : C \to A$, з $f \circ g_1 = f \circ g_2$ випливає, що $g_1 = g_2$. Іншими словами, f є ін'єктивним відносно композиції зліва.*

Морфізм $f : A \to B$ в категорії $\mathcal{C}$ називається ізоморфізмом, якщо існує морфізм $g : B \to A$ такий, що $g \circ f = id_A$ і $f \circ g = id_B$. Ізоморфізми можна розглядати як бієкції між об'єктами категорії.

Дві категорії $\mathcal{C}$ і $\mathcal{D}$ називаються еквівалентними, якщо існують два функтори $F : \mathcal{C} \to \mathcal{D}$ і $G : \mathcal{D} \to \mathcal{C}$, такі що $G \circ F$ є натурально ізоморфним тотожному функтору $id_{\mathcal{C}}$, а $F \circ G$ є натурально ізоморфним тотожному функтору $id_{\mathcal{D}}$. Це означає, що F і G встановлюють взаємно однозначну відповідність між об'єктами та морфізмами категорій $\mathcal{C}$ і $\mathcal{D}$.

Дамо більш суворе визначення коалгебри та основних пов'язаних з нею понять. Для цього розглянемо деяку категорію $\mathbb{C}$ і її ендофунктор $\boldsymbol{F} : \mathbb{C} \to \mathbb{C}$.

Означення 2.4 *Стрілка c з категорії $\mathbb{C}$ називається $\boldsymbol{F}$-коалгеброю, якщо* $\operatorname{cod} c = \boldsymbol{F}(\operatorname{dom} c)$. *Для коалгебри c категорію $\mathbb{C}$ прийнято називати основною.*

Означення 2.5 *Для $\boldsymbol{F}$-коалгебри c об'єкт* $\operatorname{dom} c$ *з категорії $\mathbb{C}$ називається носієм коалгебри і позначається $\underline{c}$.*

Таким чином, дискретна система означається як система, що змінює свій стан під дією функції переходу, при чому стан з якого було здійснено перехід знаходиться у тій же множині станів що і стан у який було здійснено перехід. Завдяки використанню ендофункторів, коалгебри дозволяють абстрактно описувати переходи між станами системи без необхідності занурення у деталі реалізації. Це означає, що коалгебра може моделювати будь-яку дискретну систему, фокусуючись лише на її поведінці, а не на конкретних механізмах реалізації переходів.

Означення 2.6 *Крім того, для* $\boldsymbol{F}$*-коалгебр* b *і* c $\boldsymbol{F}$*-морфізм з* b *в* c *є стрілкою* $\underline{b} \xrightarrow{f} \underline{c}$ *в основній категорії* $\mathbb{C}$ *такою, що*

$$\textit{діаграма} \quad \begin{array}{ccc} \underline{b} & \xrightarrow{\;f\;} & \underline{c} \\ {\scriptstyle b}\downarrow & & \downarrow{\scriptstyle c} \\ \boldsymbol{F}\,\underline{b} & \xrightarrow{\;\boldsymbol{F}f\;} & \boldsymbol{F}\,\underline{c} \end{array} \quad \textit{комутативна, тобто} \quad cf = (\boldsymbol{F}f)b. \tag{2.1}$$

Таким чином, використання теорії універсальних коалгебр, зокрема властивостей ендофункторів дає можливість узагальнювати знання про систему як об'єкта відповідної категорії систем [54]. Використання подібних діаграм дає нам можливість визначити співвідношення між усіма станами та функціями переходів попарно для усіх систем з категорії систем, що досліджується.

Наступна теорема є цілком очевидною, але дуже важливою для аналізу поведінки та специфікації класу систем, які мають спільні властивості.

Теорема 2.1 *Нехай* $\mathbb{C}$ *— категорія, а* $\boldsymbol{F}$ *— її ендофунктор, тоді сукупності* $\boldsymbol{F}$*-коалгебр і* $\boldsymbol{F}$*-морфізмів утворюють категорію, яку називають* $\mathbb{C}_{\boldsymbol{F}}$*.*
Крім того, правила

$$\mathrm{U}c = \underline{c} \qquad \textit{де } c \textit{ } \boldsymbol{F}\textit{-коалгебра} \tag{2.2а}$$

$$\mathrm{U}(b \xrightarrow{f} c) = \underline{b} \xrightarrow{f} \underline{c} \qquad \textit{де } b \textit{ і } c \textit{ } \boldsymbol{F}\textit{-коалгебри} \tag{2.2б}$$

визначте забуваючий функтор $\mathrm{U} : \mathbb{C}_{\boldsymbol{F}} \to \mathbb{C}$*, який, очевидно, є строгим.*

Використання цієї теореми дає можливість стверджувати, що результати специфікації, що проводиться з використання теорії універсальних коалгебр дійсно можна застосовувати не для однієї конкретної системи, а для усієї категорії однотипних систем. Ця універсальність дозволяє використовувати єдину теоретичну основу для аналізу широкого спектру систем.

Згідно з Яном Руттеном [55], $\boldsymbol{F}$-коалгебри називаються $\boldsymbol{F}$-системами у випадку, коли $\boldsymbol{F}$ є ендофунктором категорії **Set** множин і функцій. Далі

у дослідженні буде використана саме таке обмежене розуміння, що значно спростить використання математичних моделей.

Такий підхід виправдовується тим, що коли універсальні коалгебри використовуються як інструмент специфікації та аналізу дискретних динамічних систем, пов'язаних з кібернитичними та кіберфізичними комплексамикомплексами, визначенні ендофунктори будуть працювати саме у категорії **Set**, що є адекватним вибором для їх моделювання.

Серед основних понять теорії універсальних коалгебр, які використовуються для дослідження категорій систем можна виділити наступні [56]:

1. поняття фінальної системи,
2. поняття бісимуляції,
3. принцип коіндукції.

Далі буде наведено означення цих понять з загальнокоалгебраїчної точки зору з умовою, що $\boldsymbol{F} : \mathbb{C} \to \mathbb{C}$ — ендофунктор категорії $\mathbb{C}$. Доцільність та справедливість цієї умови була наведена вище.

2.1.1 Фінальна коалгебра

Означення 2.7 *$\boldsymbol{F}$-коалгебра $\boldsymbol{\nu F}$ називається фінальною, якщо існує рівно один $\boldsymbol{F}$-морфізм з будь-якої $\boldsymbol{F}$-коалгебри в $\boldsymbol{\nu F}$, тобто якщо $\boldsymbol{\nu F}$ — фінальний об'єкт категорії $\mathbb{C}_{\boldsymbol{F}}$.*

Для $\boldsymbol{F}$-коалгебри c єдиний $\boldsymbol{F}$-морфізм з c у чназивається анаморфізмом і позначається як $[\![\boldsymbol{c}]\!]$, або $[\![\boldsymbol{c}]\!]_{\boldsymbol{F}}$, якщо потрібне явне посилання на ендофунктор.

Важливість поняття фінальної коалгебри демонструє Лема Ламбека [57], яка встановлює фундаментальну властивість $\boldsymbol{\nu F}$.

Теорема 2.2 (Лема Ламбека) *Стрілка* $\mathrm{U}(\boldsymbol{\nu F})$ *є ізоморфізмом* $\underline{\boldsymbol{\nu F}} \cong \boldsymbol{F}\,\underline{\boldsymbol{\nu F}}$ *в* $\mathbb{C}$.

Доведення. Розглянемо наступну схему

$$\begin{array}{ccccc}
\underline{\boldsymbol{\nu F}} & \xrightarrow{\boldsymbol{\nu F}} & \boldsymbol{F}\,\underline{\boldsymbol{\nu F}} & \xrightarrow{[\![\boldsymbol{F}(\boldsymbol{\nu F})]\!]} & \underline{\boldsymbol{\nu F}} \\
{\scriptstyle \boldsymbol{\nu F}}\downarrow & & \downarrow{\scriptstyle \boldsymbol{F}(\boldsymbol{\nu F})} & & \downarrow{\scriptstyle \boldsymbol{\nu F}} \\
\boldsymbol{F}\,\underline{\boldsymbol{\nu F}} & \xrightarrow{\boldsymbol{F}(\boldsymbol{\nu F})} & \boldsymbol{F}(\boldsymbol{F}\,\underline{\boldsymbol{\nu F}}) & \xrightarrow{\boldsymbol{F}[\![\boldsymbol{F}(\boldsymbol{\nu F})]\!]} & \boldsymbol{F}\,\underline{\boldsymbol{\nu F}}
\end{array}$$

Очевидно, вона комутативна. Враховуючи фінальність $\boldsymbol{\nu F}$, маємо

$$[\![\boldsymbol{F}(\boldsymbol{\nu F})]\!]\boldsymbol{\nu F} = \mathrm{id}_{\underline{\boldsymbol{\nu F}}}\ .$$

Далі,

$$\boldsymbol{\nu F}[\![\boldsymbol{F}(\boldsymbol{\nu F})]\!] = (\boldsymbol{F}[\![\boldsymbol{F}(\boldsymbol{\nu F})]\!])(\boldsymbol{F}\boldsymbol{\nu F}) =$$
$$\boldsymbol{F}([\![\boldsymbol{F}(\boldsymbol{\nu F})]\!]\boldsymbol{\nu F}) = \boldsymbol{F}(\mathrm{id}_{\underline{\boldsymbol{\nu F}}}) = \mathrm{id}_{\boldsymbol{F}\,\underline{\boldsymbol{\nu F}}}\ .$$

Таким чином, $\boldsymbol{\nu F}^{-1} = [\![\boldsymbol{F}(\boldsymbol{\nu F})]\!]$ ■

Іншими словами, лема Ламбека стверджує, що для будь-якого функтора $\boldsymbol{F}$ на категорії $\mathbb{C}$, фінальна $\boldsymbol{\nu F}$-коалгебра (якщо вона існує) є нерухомою точкою функтора $\boldsymbol{F}$. Тобто, якщо така система є фінальною $\boldsymbol{F}$-коалгеброю, то існує ізоморфізм $\mathrm{U}(\boldsymbol{\nu F})$.

З наведеної леми можна зробити наступні висновки:

1. По-перше, вона встановлює, що структура фінальної коалгебри повністю визначається її образом під дією функтора $\boldsymbol{F}$. Це означає, що ми можемо розглядати фінальну коалгебру як об'єкт, який є стійким (незмінним) під дією цього функтора.

2. По-друге, лема підкреслює, що для будь-якої $\boldsymbol{F}$-коалгебри (C, γ), існує єдиний гомоморфізм до фінальної коалгебри $\boldsymbol{\nu F}$. Це забезпечує однозначність способу, яким інші коалгебри відображаються у фінальну коалгебру.

3. По-третє, лема Ламбека є фундаментальною для теоретичного обґрунтування використання коалгебр у моделюванні та аналізі дискретних систем. Вона забезпечує інструменти для формального аналізу властивостей систем і перевірки їхньої поведінки.

4. Фінальна коалгебра є нерухомою точкою ендофунктора $\boldsymbol{F}$.

2.1.2 Бісимуляція

Наступною важливою концепцією є бісимуляція. Бісимуляція – це математичне поняття, що використовується у теорії автоматів, теорії обчислень та формальній верифікації для визначення еквівалентності між станами дискретних динамічних систем, таких як автомати або перехідні системи [58]. Дві системи (або два стани систем) вважаються бісиміляpними, якщо вони можуть імітувати поведінку одна одної.

Формально бісимуляцію можна визначити як відношенням R між станами двох систем переходів (наприклад, автоматів), яке задовольняє такі умови [59]:

Нехай $(S, \to)$ та $(T, \Rightarrow)$ – дві системи переходів, де S і T – множини станів, а $\to$ і $\Rightarrow$ – перехідні відношення. Відношення $R \subseteq S \times T$ є бісимуляцією, якщо для будь-яких станів $s \in S$ і $t \in T$, які перебувають у відношенні $s\ R\ t$, виконуються такі умови:

1. Якщо $s \to s'$, то існує такий стан $t' \in T$, що $t \Rightarrow t'$ і $s'\ R\ t'$.

2. Якщо $t \Rightarrow t'$, то існує такий стан $s' \in S$, що $s \to s'$ і $s'\ R\ t'$.

У випадку $\boldsymbol{F}$-систем бісимуляція є різновидом відношення між носіями двох систем, що дозволяє створити відповідну структуру $\boldsymbol{F}$-системи для цього відношення. Бісимуляції використовуються як інструмент для порівняння та аналізу поведінки систем. Універсальні коалгебри забезпечують зручну математичну основу для моделювання станів і переходів у динамічних системах, а бісимуляції дозволяють формально визначити еквівалентність між різними системами чи станами в рамках цієї моделі.

Отже, нехай s і t є $\boldsymbol{F}$-системами, тоді відношення $R \subset \underline{s} \times \underline{t}$ є бісимуляцією, якщо існує $r : R \to \boldsymbol{F}R$ що забезпечує комутативність наступної діаграми

$$\begin{array}{ccccc} \underline{s} & \xleftarrow{\pi_s} & R & \xrightarrow{\pi_t} & \underline{t} \\ {\scriptstyle s}\downarrow & & \vdots\,{\scriptstyle r} & & \downarrow{\scriptstyle t} \\ \boldsymbol{F}\,\underline{s} & \xleftarrow{\boldsymbol{F}\pi_s} & \boldsymbol{F}R & \xrightarrow{\boldsymbol{F}\pi_t} & \boldsymbol{F}\,\underline{t} \end{array} \tag{2.3}$$

де π_s і π_t — обмеження на R натуральних проекцій від $\underline{s} \times \underline{t}$ на $\underline{s}$ і $\underline{t}$ відповідно.

Беручи до уваги, що інтервал у будь-якій категорії є діаграмою форми

$$s \xleftarrow{\pi_s} r \xrightarrow{\pi_t} t$$

, можна сказати, що діаграма 2.3 є інтервалом у категорії $\mathbf{Sys}_{\boldsymbol{F}}$.

Також за конструкцією інтервал

$$\underline{s} \xleftarrow{\pi_s} R \xrightarrow{\pi_t} \underline{t}$$

у категорії **Set** це моноінтервал, тобто для будь-якої множини X і функцій $f, g : X \to R$,

комутативність діаграми $\begin{array}{ccccc} & & X & & \\ & & {\scriptstyle f}\downarrow\downarrow{\scriptstyle g} & & \\ \underline{s} & \xleftarrow{\pi_s} & R & \xrightarrow{\pi_t} & \underline{t} \end{array}$ забезпечує $f = g$.

Таким чином, можна узагальнити визначення бісимуляції для систем на випадок коалгебр у довільній основній категорії наступним чином.

Означення 2.8 *Нехай $\mathbb{C}$ — категорія, $\boldsymbol{F}$ - її ендофунктор, а b і c — $\boldsymbol{F}$-коалгебри, тоді моноінтервал*

$$\underline{b} \xleftarrow{\pi_b} R \xrightarrow{\pi_c} \underline{c}$$

є бісимуляцією між b і c, якщо існує $\boldsymbol{F}$-коалгебра r така, що $\underline{r} = R$ і для якої π_b і π_c є $\boldsymbol{F}$-морфізмами з r в b і c відповідно.

Іншими словами, бісимуляція - це моноінтервал

$$\underline{b} \xleftarrow{\pi_b} R \xrightarrow{\pi_c} \underline{c}$$

у $\mathbb{C}$ який піднімається до інтервалу

$$b \xleftarrow{\pi_b} r \xrightarrow{\pi_c} c$$

в $\mathbb{C}_{\boldsymbol{F}}$ таким чином, що $\underline{r} = R$.

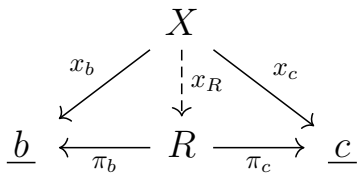
Означення 2.9 *Припустимо*

$$\underline{b} \xleftarrow{\pi_b} R \xrightarrow{\pi_c} \underline{c}$$

є бісимуляцією $\boldsymbol{F}$ *- коалгебр* b *та* c *тоді можна стверджувати, що* $X \xrightarrow{x_b} \underline{b}$ *та* $X \xrightarrow{x_c} \underline{c}$ *бісимулюються інтервалом*

$$\underline{b} \xleftarrow{\pi_b} R \xrightarrow{\pi_c} \underline{c}$$

якщо існує $X \xrightarrow{x_R} R$*, такий, що*

діаграма  *є комутативною.*

Для категорії $\mathbb{C}$ і ендофунктора $\boldsymbol{F} : \mathbb{C} \to \mathbb{C}$ припустимо, що існує фінальна $\boldsymbol{F}$-коалгебра.

Теорема 2.3 *Для* $\boldsymbol{F}$*-коалгебр* b *та* c *та бісимуляції*

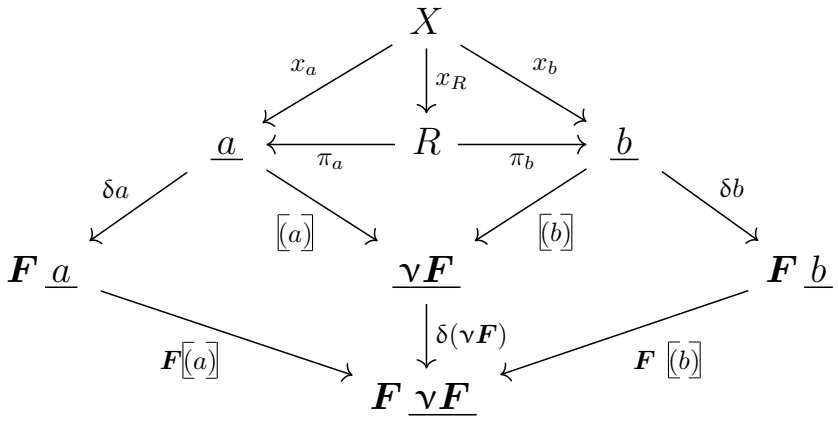
$$\underline{b} \xleftarrow{\pi_b} R \xrightarrow{\pi_c} \underline{c}$$

, якщо $X \xrightarrow{x_b} \underline{b}$ *та* $X \xrightarrow{x_c} \underline{c}$ *бісимулюються інтервалом*

$$\underline{b} \xleftarrow{\pi_b} R \xrightarrow{\pi_c} \underline{c}$$

тоді $[\![b]\!]x_b = [\![c]\!]x_c$.

Доведення. Спочатку покажемо що наступна діаграма є комутативною



Отже, маємо

$$\Big(\big(\boldsymbol{F}[\![a]\!]\big)\delta a\Big)x_a = \Big(\big(\boldsymbol{F}[\![b]\!]\big)\delta b\Big)x_b$$

Використовуючи комутативність обох нижніх квадратів, можна переписати рівняння наступним чином

$$\delta(\boldsymbol{\nu F})[\![a]\!]x_a = \delta(\boldsymbol{\nu F})[\![b]\!]x_b$$

Тепер, завдяки лемі Ламбека, можна зробити висновок $[\![a]\!]x_a = [\![b]\!]x_b$. ■

Додатково припускаючи, що $\mathbb{C}$ містить усі декартові квадрати, а $\boldsymbol{F}$ їх зберігає, існування бісимуляції може бути гарантовано.

Теорема 2.4 *Для $\boldsymbol{F}$-коалгебр a та b, інтервал*

$$\underline{a} \xleftarrow{\pi_a} R \xrightarrow{\pi_b} \underline{b}$$

в $\mathbb{C}$ такий що

$$\begin{array}{ccc} R & \xrightarrow{\pi_b} & \underline{b} \\ {\scriptstyle \pi_a}\downarrow & & \downarrow{\scriptstyle [\![b]\!]} \\ \underline{a} & \xrightarrow{[\![a]\!]} & \underline{\boldsymbol{\nu F}} \end{array} \qquad \textit{це діаграма є декартовим квадратом} \tag{2.4}$$

тоді можна стверджувати, що

$$a \xleftarrow{\pi_a} r \xrightarrow{\pi_b} b$$

є моноінтервалом в $\mathbb{C}_{\boldsymbol{F}}$ *тобто зазначений інтервал є бісимуляцією. Ця бісимуляція задовольняє умові: для об'єкта* X *з* $\mathbb{C}$ *та морфізмів* $X \xrightarrow{x_a} \underline{a}$, $X \xrightarrow{x_b} \underline{b}$, *умова* $[\![a]\!]x_a = [\![b]\!]x_b$ *заебзпучує, що*

$$\underline{a} \xleftarrow{\pi_a} R \xrightarrow{\pi_b} \underline{b}$$

бісімулює x_a *та* x_b.

Доведення. Зауважимо, що

$$\underline{a} \xleftarrow{\pi_a} R \xrightarrow{\pi_b} \underline{b}$$

це моноінтервал згідно з (2.4).

$$\begin{array}{ccccc}
R & \xrightarrow{\pi_b} & \underline{b} & & \\
\downarrow{\scriptstyle \pi_a} & \overset{\delta}{\searrow} & & \searrow{\scriptstyle \delta b} & \\
\underline{a} & & \boldsymbol{F}R & \xrightarrow{\boldsymbol{F}\pi_b} & \boldsymbol{F}\,\underline{b} \\
& \searrow{\scriptstyle \delta a} & \downarrow{\scriptstyle \boldsymbol{F}\pi_a} & & \downarrow{\scriptstyle \boldsymbol{F}[\![b]\!]} \\
& & \boldsymbol{F}\,\underline{a} & \xrightarrow{\boldsymbol{F}[\![a]\!]} & \boldsymbol{F}\,\underline{\nu\boldsymbol{F}}
\end{array} \tag{2.5}$$

Перевіримо рівняння

$$\big(\boldsymbol{F}[\![a]\!]\big)(\delta a)\pi_a = \big(\boldsymbol{F}[\![b]\!]\big)(\delta a)\pi_b. \tag{2.6}$$

Дійсно,

$$(\delta\nu\boldsymbol{F})[\![a]\!] = \big(\boldsymbol{F}[\![a]\!]\big)\delta a$$

та

$$(\delta\nu\boldsymbol{F})[\![b]\!] = \big(\boldsymbol{F}[\![b]\!]\big)\delta a$$

тому що $[\![a]\!]$ та $[\![b]\!]$ є $\boldsymbol{F}$-морфізми. Отже, після заміни ми приходимо до необхідності перевірки

$$\delta(\nu\boldsymbol{F})[\![a]\!]\pi_a = \delta(\nu\boldsymbol{F})[\![b]\!]\pi_b$$

або, беручи до уваги лему Ламбека,

$$[\![a]\!]\pi_a = [\![b]\!]\pi_b.$$

Але останнє рівняння вірне завдяки (2.4). Враховуючи припущення, що $\boldsymbol{F}$ зберігає декартові квадрати, південно-східний квадрат (2.5) також є діаграмою декартового квадрату. Підсумовуючи міркування, можна зробити висновок, що існує лише одна стрілка δ, яка робить (2.5) комутативним. Якщо r визначено як $\boldsymbol{F}$-коалгебру таку, що $\underline{r} = R$ і $\delta r = \delta$, то r є ліфтом із визначення бісимуляції. Умова, яку стверджує теорема, безпосередньо випливає з (2.4). ■

Отже, бісимуляція в універсальних коалгебрах є ключовим інструментом для:

1. Порівняння систем: Дозволяє формально визначати, чи дві системи є еквівалентними за їх поведінкою.

2. Зменшення складності моделей: Допомагає виявляти і замінювати частини системи їхніми еквівалентами, що спрощує аналіз і верифікацію.

3. Аналізу і верифікації: Застосовується у формальній верифікації систем, що потребує точного аналізу їх поведінки.

2.1.3 Коіндукція

Коіндукція є ключовою концепцією в теорії універсальних коалгебр, яка надає формалізм для визначення та аналізу нескінченних структур і динамічних систем. Цей підхід базується на дуальності до індукції та забезпечує методи для роботи з нескінченними об'єктами, такими як потоки даних, автоматні структури та інші динамічні системи [60]. Коіндукція дозволяє формально доводити властивості та еквівалентність таких систем.

Як було зазначено вище, універсальна коалгебра складається з множини станів та функції переходів, яка визначає динаміку системи. Коалгебра визначається як пара (C, γ), де C — множина станів, а $\gamma : C \to FC$ — функція переходів, що описує динаміку системи через функтор F. Коіндукція базується на ідеї, що два об'єкти є еквівалентними, якщо вони не можуть

бути розрізнені на основі спостережуваних властивостей [61]. Це відношення еквівалентності визначається через бісимуляцію, яка є відношенням між двома коалгебрами, що зберігає структуру їх переходів.

Основна ідея коіндуктивних доведень полягає в тому, щоб показати, що певне відношення є бісимуляцією. Якщо відношення є бісимуляцією, то всі стани, пов'язані цим відношенням, коіндуктивно еквівалентні, тобто мають однакові поведінкові властивості. Коіндукція дозволяє доводити властивості систем, показуючи, що певне відношення зберігає структуру переходів. Це є особливо корисним для аналізу систем з нескінченною або рекурсивною природою.

Коіндукція широко застосовується у формальній верифікації програмного забезпечення, теорії автоматів, аналізі процесів і систем. Вона дозволяє формально визначати та доводити властивості систем, забезпечуючи точність і строгість аналізу. Коіндукція також сприяє модульності, дозволяючи розділити аналіз системи на менші частини, які можна аналізувати окремо.

Розглянемо приклад нескінченного потоку натуральних чисел. Нехай $Stream$ — коалгебра, яка визначає кожен стан потоку як пару з першого елемента і залишку потоку. Функція переходів $\gamma : Stream \to \mathbb{N} \times Stream$ може бути визначена як:

$$\gamma(s) = (head(s), tail(s))$$

де $head(s)$ — перший елемент потоку, а $tail(s)$ — залишок потоку. Два потоки є бісимілярними, якщо їхні перші елементи рівні і залишки потоків також бісимілярні. Для доведення еквівалентності двох потоків s_1 і s_2 коіндуктивно, необхідно показати, що:

$$head(s_1) = head(s_2)$$

$$tail(s_1) \sim tail(s_2)$$

де $\sim$ позначає відношення бісимуляції. Якщо ці умови виконуються, то потоки s_1 і s_2 є коіндуктивно еквівалентними.

Коіндукція надає декілька переваг у формальному аналізі систем [62]. По-перше, вона природно підходить для аналізу нескінченних структур, дозволяючи працювати з об'єктами, що мають нескінченну або рекурсивну природу. По-друге, коіндукція забезпечує точні і формальні методи для доведення властивостей систем, що особливо корисно у верифікації програмного забезпечення. По-третє, коіндукція сприяє модульності, дозволяючи розділити аналіз системи на менші частини, що спрощує загальний процес верифікації.

Таким чином, коіндукція дозволяє формально визначати і доводити властивості динамічних систем, забезпечуючи точність і строгість аналізу. Вона дозволяє працювати з нескінченними або рекурсивними об'єктами, забезпечуючи коректність їх специфікації та верифікації. Коіндукція також сприяє модульності, дозволяючи розділити аналіз системи на менші частини, що спрощує загальний процес верифікації. Це робить коіндукцію потужним інструментом для формального аналізу і верифікації динамічних систем.

2.2 Важливість фінальної коалгебри для дослідження поведінки системи

Фінальна коалгебра є фундаментальним поняттям у теорії коалгебр, яке відіграє критичну роль у специфікації та аналізі дискретних динамічних систем. Коалгебри, як дуальні до алгебр конструкції, забезпечують потужний математичний апарат для моделювання динамічних систем, що змінюються з часом. Фінальна коалгебра, як нерухома точка функтора, надає канонічну модель для коалгебричної специфікації, що дозволяє формально і ефективно досліджувати властивості таких систем.

Фінальна коалгебра є ключовою для специфікації та аналізу дискретних динамічних систем, оскільки вона надає канонічний спосіб визначення по-

ведінкової еквівалентності станів системи. Універсальні коалгебри дозволяють описувати динамічні системи в абстрактний спосіб, зосереджуючи увагу на поведінці системи, а не на її внутрішній структурі [63]. Це особливо важливо для моделювання складних систем, де важливо враховувати їхню поведінку у всіх можливих станах.

Фінальна коалгебра надає природний спосіб визначення бісимуляції, оскільки вона забезпечує єдину канонічну форму для всіх коалгебр. Це дозволяє формально доводити еквівалентність різних реалізацій однієї і тієї ж системи, що є критичним для її верифікації.

Однією з важливих переваг використання фінальної коалгебри є модульність. Це дозволяє розділити складну систему на менші частини, кожну з яких можна аналізувати окремо [64]. Такий підхід значно спрощує процес специфікації і верифікації, оскільки кожну підсистему можна розглядати незалежно, а потім інтегрувати результати аналізу для отримання загальної картини. Це особливо корисно при роботі зі складними програмними системами, де важливо забезпечити коректність і надійність кожного компонента.

Таким чином, фінальна коалгебра відіграє важливу роль у специфікації дискретних динамічних систем з використанням універсальних коалгебр. Вона забезпечує канонічний спосіб визначення поведінкової еквівалентності, що є критичним для формального аналізу і верифікації систем. Використання фінальної коалгебри дозволяє формально доводити еквівалентність і коректність систем, забезпечуючи точність і надійність аналізу. Це робить коалгебри потужним інструментом для моделювання, специфікації і верифікації складних динамічних систем.

2.3 Підходи до доведення існування фінальної коалгебри за допомогою декартових квадратів

Однак слід зазначити, що процес виведення фінальної системи не завжди є тривіальним та потребує використання коіндуктивного підходу. Його

використання у свою чергу можливо тільки якщо ми впевнені, що фінальна система існує (що не завжди вірно). У цьому розділі буде запропонована достатня умова існування фінальної системи, використання якої значно спрощує аналіз поведінки та специфікацію дискретних динамічних систем.

При дослідженні довільного ендофунктора відповідна категорія систем може мати доволі бідний набір властивостей, якого недостатньо для повноцінного аналізу системи. бідний властивостей. Підходи, що будуть розглянуті у цьому розділі, забезпечать збагачення цього набору властивостей.

Ці підходи, пов'язані з вивченням діаграм що називаються коінтервалами та мають наступний вигляд

$$\circ \longrightarrow \circ \longleftarrow \circ$$

2.3.1 Визначення слабкого декартового квадрату

Декартовий квадрат — це математичне поняття, яке використовується для побудови нового простору з двох заданих множин [65]. Формально, якщо A і B — дві множини, то їхній декартовий квадрат або декартовий добуток позначається як $A \times B$ і визначається як множина всіх упорядкованих пар (a, b), де $a \in A$ і $b \in B$.

У загальному випадку, для будь-яких двох множин A і B:

$$A \times B = \{(a, b) \mid a \in A \text{ і } b \in B\}$$

Це означає, що декартовий квадрат включає всі можливі комбінації елементів з A і B у вигляді пар. Наприклад, якщо $A = \{1, 2\}$ і $B = \{x, y\}$, то їхній декартовий квадрат буде:

$$A \times B = \{(1, x), (1, y), (2, x), (2, y)\}$$

Більш формально декартів квадрат визначається наступним чином

Означення 2.10 *Для коінтервалу*

$$X \xrightarrow{f} Z \xleftarrow{g} Y$$

розглянемо наступні множину та стрілки

$$P = \{(x, y) \in X \times Y \mid fx = gy\}$$
$$q_X = \lambda\,(x, y)\,.\,x : P \to X$$
$$q_Y = \lambda\,(x, y)\,.\,y : P \to Y.$$

Тоді інтервал

$$X \xleftarrow{q_X} P \xrightarrow{q_Y} Y$$

називають декартовим квадратом коінтервалу

$$X \xrightarrow{f} Z \xleftarrow{g} Y$$

.

Наступні спостереження дадуть можливість формально визначити поняття слабкого декартового квадрату. Розглянемо наступну діаграму

$$\begin{array}{ccc} P & \xrightarrow{q_Y} & Y \\ {\scriptstyle q_X}\downarrow & & \downarrow{\scriptstyle g} \\ X & \xrightarrow{f} & Z \end{array},$$

що складається з вихідного коінтервалу

$$X \xrightarrow{f} Z \xleftarrow{g} Y$$

та його декартова квадрату

$$X \xleftarrow{q_X} P \xrightarrow{q_Y} Y$$

, яка, очевидно, є комутативною.

Для цієї діаграми виконується наступна універсальна властивість: для будь-яких U, u, v таких, що зовнішній квадрат у (2.7) комутує

$$\begin{array}{ccccc} U & & \xrightarrow{v} & & Y \\ & \searrow^{h} & & & \| \\ {\scriptstyle u}\downarrow & & P & \xrightarrow{q_Y} & Y \\ & & {\scriptstyle q_X}\downarrow & & {\scriptstyle g}\downarrow \\ X & = & X & \xrightarrow{f} & Z \end{array} \tag{2.7}$$

існує *рівно один* h, що робить (2.7) комутативним.

Вимога єдиності h в універсальній властивості (2.7) є досить сильною з математичної точки зору. "Ослаблення"цієї умови призводить до концепції слабкого декартового квадрату.

Означення 2.11 *Інтервал*

$$X \xleftarrow{q_X} Q \xrightarrow{q_Y} Y$$

є слабким декартовим квадратом для коінтервалу

$$X \xrightarrow{f} Z \xleftarrow{g} Y$$

якщо діаграма

$$\begin{array}{ccc} Q & \xrightarrow{q_Y} & Y \\ {\scriptstyle q_X}\downarrow & & \downarrow{\scriptstyle g} \\ X & \xrightarrow{f} & Z \end{array}$$

комутативна та для будь-яких U, $u : U \to X$, $v : U \to Y$ *таких, що зовнішній квадрат в* (2.8) *є комутативним*

$$\begin{array}{ccccc} U & & \xrightarrow{\quad v \quad} & & Y \\ & \searrow^{h} & & & \| \\ {\scriptstyle u}\downarrow & & Q & \xrightarrow{q_Y} & Y \\ & & \downarrow{\scriptstyle q_X} & & {\scriptstyle g}\downarrow \\ X & = & X & \xrightarrow{f} & Z \end{array} \tag{2.8}$$

існує ***принаймні один*** h, *що робить* (2.8) *комутативним.*

2.3.2 Збереження ендофункторами декартових квадратів

У цьому розділі будуть показані властивості ендофунктора зберігати (слабкі) декартові квадрати та надані відповідні визначення, необхідні для виведення достатньої умови існування фінальної системи. для відповідного класу систем. Тому ми даємо відповідні визначення та обговорюємо тут їх співвідношення.

Означення 2.12 *Нехай* $\boldsymbol{F}$ *- ендофунктор категорії* **Set**, *можна стверджувати, що* $\boldsymbol{F}$ *зберігає декартові квадрати, якщо діаграма*

$$\boldsymbol{F}X \xleftarrow{\boldsymbol{F}q_X} \boldsymbol{F}P \xrightarrow{\boldsymbol{F}q_Y} \boldsymbol{F}Y$$

є декартовим квадратом коінтервалу

$$\boldsymbol{F}X \xrightarrow{\boldsymbol{F}f} \boldsymbol{F}Z \xleftarrow{\boldsymbol{F}g} \boldsymbol{F}Y$$

у той час коли діаграма

$$X \xleftarrow{q_X} P \xrightarrow{q_Y} Y$$

є декартовим квадратом коінтервалу

$$X \xrightarrow{f} Z \xleftarrow{g} Y$$

.

Наступні прості факти демонструють зв'язок між введеними поняттями.

Твердження 2.1 *Нехай* $\boldsymbol{F}$ *це ендофунктор в категорії* **Set** *тоді*

1. *якщо* $\boldsymbol{F}$ *зберігає декартові квадрати тоді він зберігає і слабкі декартові квадрати;*

2. *якщо* $\boldsymbol{F}$ *зберігає слабкі декартові квадрати тоді він перетворює декартові квадрати у слабкі декартові квадрати;*

3. *якщо* $\boldsymbol{F}$ *перетворює декартів квадрат у слабкий декартів квадрат, тоді він зберігає слабкі декартові квадрати.*

Приведена далі Лема є необхідною для подальшого виведення достатньої умови існування фінальної системи.

Лема 2.1 *Нехай* $\boldsymbol{F}$ *— ендофунктор категорії* **Set** *такий, що для декартового квадрату*

$$X \xleftarrow{q_X} P \xrightarrow{q_Y} Y$$

для коінтевалу

$$X \xrightarrow{f} Z \xleftarrow{g} Y$$

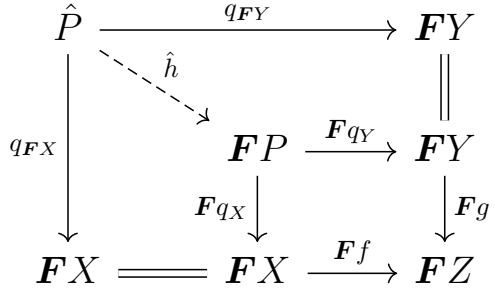
та декартового квадрату

$$\boldsymbol{F}X \xleftarrow{q_{\boldsymbol{F}X}} \hat{P} \xrightarrow{q_{\boldsymbol{F}Y}} \boldsymbol{F}Y$$

для коінтевалу

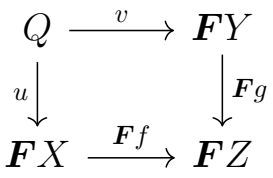
$$\boldsymbol{F}X \xrightarrow{\boldsymbol{F}f} \boldsymbol{F}Z \xleftarrow{\boldsymbol{F}g} \boldsymbol{F}Y$$

існує $\hat{h} : \hat{P} \to \boldsymbol{F}P$ *, що забезпечує комутативність діаграми*

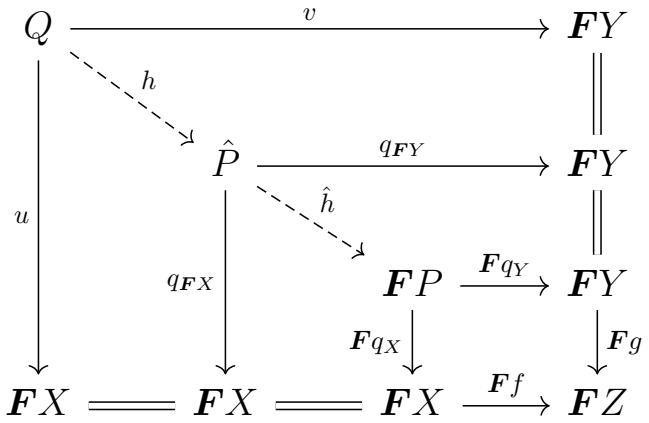


тоді $\boldsymbol{F}$ *зберігає слабкі декартові квадрати.*

Доведення. Припустимо діаграма



комутативна. Тепер можна будувати наступну діаграму



де h — функція з визначення декартового квадрату. Очевидно, що ця діаграма є комутативною, і тому $\hat{h} \circ h : Q \to \boldsymbol{F}P$ є функцією з визначення слабкого декартового квадрату. ■

Таким чином ми довели достатню умову збереження ендофунктором слабких декартових квадратів. Наступне очевидне твердження розширяє цю умову на композицію ендофункторів наступним чином.

Твердження 2.2 *Нехай* $\boldsymbol{F}$ *і* $\boldsymbol{G}$ *є ендофункторами категорії* **Set** *і обидва ці ендфунктори зберігають (слабкі) декартові квадрати, тоді ендфунктор* $\boldsymbol{G} \circ \boldsymbol{F}$ *також зберігає (слабкі) декартові квадрати .*

2.3.3 Збереження декартових квадратів підсистемами

Підсистема в теорії універсальних коалгебр розглядається як коалгебра, яка є частиною або компонентом більшої коалгебри. З точки зору коалгебр, підсистема може бути визначена як коалгебра (S, γ_S), яка вбудована у більшу коалгебру (C, γ_C) через ін'єктивний морфізм коалгебр. Це вбудовування забезпечує зв'язок між поведінкою підсистеми і поведінкою всієї системи.

Універсальні коалгебри забезпечують абстрактний і загальний підхід до моделювання динамічних систем, дозволяючи розглядати підсистеми як окремі коалгебри, що взаємодіють з іншими компонентами системи через морфізми. Це дозволяє формально аналізувати і доводити властивості підсистем незалежно від решти системи, забезпечуючи модульність і масштабованість специфікацій.

У цьому розділі буде наведено розширення умов зберігання ендофункторами (слабких) декартових квадратів на підсистеми. Усі твердження будуть робитися з фіксацією ендофунктора $\boldsymbol{F}$ категорії **Set**.

Для будь-якої $\boldsymbol{F}$-системи (X, δ) і підмножини $V \subset X$ можна розглянути

діаграму

$$\begin{array}{ccc} V & \xhookrightarrow{i} & X \\ & & \downarrow{\scriptstyle \delta} \\ \boldsymbol{F}V & \xrightarrow{\boldsymbol{F}i} & \boldsymbol{F}X \end{array}$$

Якщо цю діаграму можна доповнити деякою $\delta' : V \to \boldsymbol{F}V$ до комутативної діаграми

$$\begin{array}{ccc} V & \xhookrightarrow{i} & X \\ {\scriptstyle \delta'}\vdots & & \downarrow{\scriptstyle \delta} \\ \boldsymbol{F}V & \xrightarrow{\boldsymbol{F}i} & \boldsymbol{F}X \end{array} \tag{2.9}$$

тоді пара (B, δ') є $\boldsymbol{F}$-системою, а i є $\boldsymbol{F}$-мономорфізмом цієї системи в систему (X, δ). Слід зазначити, що у разі існування такого δ' він визначається діаграмою (2.9) однозначно через ін'єктивність i, а отже, і $\boldsymbol{F}i$.

Слід зазначити, що не кожна підмножина простору станів системи є простором станів деяких її підсистем. Наступне твердження це пояснює.

Твердження 2.3 *Для простої детермінованої системи $(X, \delta : X \to X)$ підмножина $V \subset X$ може мати структуру простої детермінованої системи тоді і тільки тоді, коли $f(V) \subset V$.*

Структура сімейства всіх підсистем $\boldsymbol{F}$-системи невідома, але цю структуру можна описати, якщо $\boldsymbol{F}$ зберігає слабкі декартові квадрати.

Теорема 2.5 *Нехай $\boldsymbol{F}$ — ендофунктор категорії* **Set**, *який зберігає слабкі декартові квадрати, а (X, δ) — $\boldsymbol{F}$-система, тоді сімейством підсистем (X, δ) є повна решітка, операцією зустрічі для якої є перетин.*

Доведення цієї теореми можна знайти в [66]

Далі в цьому розділі застосовується властивість ендофункторів зберігати слабкі декартові квадрати.

Припустимо, що (X, δ) є $\boldsymbol{F}$-системою і $G \subset X$, тоді можна розглянути наступну сімейство підсистем системи (X, δ)

$$\{(V, \delta_V) \text{ є підсистемою } (X, \delta) \mid G \subset X\}$$

і взяти її найменшого члена, позначивши його як $\langle G \rangle$.

Означення 2.13 *Підсистема* $\langle G \rangle$ *називається підсистемою, породженою* G.
Якщо $H \subset X$ *такий, що* $\langle H \rangle = (X, \delta)$, *тоді члени* H *називаються генераторами* (X, δ).

2.3.4 Достатня умова існування фінальної системи

Як було зазначено раніше, доведений факт існування фінальної системи для конкретного ендофунктора дає можливість одразу використовувати коіндукцію для її виведення, на відміну від ситуації, коли цей факт не доведений. У цьому розділі буде використана теорема 10.4 з [55] та доведені раніше властивості ендофункторів зберігати слабкі декартові квадрати для виведення достатньої умові існування фінальної системи.

Для формулювання цієї теореми необхідно наступне визначення.

Означення 2.14 *Ендофунктор* $\boldsymbol{F}$ *в* **Set** *називають обмеженим, якщо існує множина* V *така, що для кожної* $\boldsymbol{F}$*-системи* (X, δ) *і кожного* $x \in X$, *існує ін'єктивна функція з носія підсистеми* $\langle x \rangle$ *у множину* V.

Теорема 2.6 (Достатня умова існування фінальної системи) *Для кожного обмеженого ендофунктора* $\boldsymbol{F}$ *в* **Set**, *який зберігає сдабкі декартові квадрати, фінальна* $\boldsymbol{F}$*-система існує.*

Доведення див. в [55, Теорема 10.4].

Ця теорема і доведена вище Лема 2.1 дають достатньо потужний інструмент доведення факту існування фінальної системи для різних практично важливих ендофункторів.

Висновки до Розділу 2

У розділі описані основні концепції теорії універсальних коалгебр як інструменту для специфікації дискретних динамічних систем, а саме: фінальну систему, бісимуляцію та коіндукцію. За допомогою неформального

пояснення наведених визначень було обгрунтовано використання саме коалгебраїчного підходу до вирішення задач специфікації та аналізу поведінки складних систем.

Окремо описано концепцію фінальної системи, як ключової концепції у теорії універсальних коалгебр, яку можна застосовувати як основу побудови моделей специфікації систем. Це обгрунтовується тим, що фінальні системи інкапсулюють (без надлишковості) у собі усі загальні властивості відповідної категорії систем. Також існує механізм становлення відповідності між усіма системами категорії систем та фінальної системи.

Було наведено достатню умову існування фінальної системи. Доведення базується на властивості деяких ендофункторів зберігати декартові квадрати. Представлені усі необхідні твердження для формулювання цієї умови та доведені відповідні теореми. Цей результат отримано вперше.

Основні положення цього розділу викладені у публікаціях автора [9-11].

РОЗДІЛ 3. ВИКОРИСТАННЯ УНІВЕРСАЛЬНИХ КОАЛГЕБР ДЛЯ МОДЕЛЮВАННЯ ТА ВИВЧЕННЯ ПОВЕДІНКИ ДИСКРЕТНИХ ДИНАМІЧНИХ СИСТЕМ

Дискретні динамічні системи використовуються для моделювання та аналізу систем, де зміни відбуваються в дискретні моменти часу. Це можуть бути як природні явища, так і технологічні процеси. Основною властивістю таких систем є те, що час у них змінюється дискретно, тобто у вигляді дискретних кроків, і стан системи змінюється відповідно до деякого закону переходу.

Формально, дискретні динамічні системи, як зазначалося вище, можна описати як пару (X, f), де X — множина можливих станів системи, а $f : X \to X$ — функція переходу, яка визначає, як стан системи змінюється з поточного моменту часу до наступного. Ця функція часто називається оператором або правилом еволюції системи.

У цьому розділі буде описано узагальнення стандартної моделі динамічної системи за рахунок зміни моделі множини наступних станів, які можуть описуватися в інших ніж поточні стани термінах – множини можливих станів, розподіли ймовірностей можливих станів, функції з множини керувань у множину станів тощо.

3.1 Моделювання детермінованої системи з вихідними даними за допомогою універсальної коалгебри

Перш за все розглянемо системи, що при зміні стану генерують повідомлення, які можуть бути інтерпретовані як вихідна інформація про стани. У рамках цього дослідження такі системи будуть називатися дискретними динамічними системами з виходами.

У таких системах згенерований вихід і новий стан однозначно залежать від поточного стану системи через її властивість детермінованості. При дослідженні поведінки та специфікації дискретних динамічних систем з виходами зі застосуванням теорії універсальних коалгбер зазвичай не аналізується їх керуюча функція. Це пов'язано з тим, що інструментарій універсальної коалгебри дає можливість аналізувати поведінку системи у цілому, абстрагуючись від її специфічних властивостей, що у свою чергу забезпечує механізм розширення знань про поведінку конкретної систему на увесь клас подібних систем з використання бісимуляції. Детально цей підхід було описано у Розділі 2.

Отже, дискретну детерміновану систему з виходами можна визначити як систему, що змінює свій стан крок за кроком і надсилає повідомлення системному середовищу на кожному кроці. Для створення формальної моделі подібних систем з використанням універсальних коалгебр необхідно зафіксувати деякий набір повідомлень $\mathbf{M}$ та використати наступний ендофунктор $\boldsymbol{\mathcal{S}}_{\mathbf{M}} : \mathbf{Set} \to \mathbf{Set}$

$$\boldsymbol{\mathcal{S}}_{\mathbf{M}} X = \mathbf{M} \times X \qquad \text{для будь-якої множини } X \tag{3.1}$$

$$\boldsymbol{\mathcal{S}}_{\mathbf{M}}(X \xrightarrow{f} Y) = \mathrm{id}_{\mathbf{M}} \times f \qquad \text{для будь-яких множин} X, Y \tag{3.2}$$

Зазначемо, що (3.1) і (3.2) визначають ендофунктор.

Таким чином, формально, детермінована система з виходом — це відображення з множини станів системи у пару стан системи - повідомлення

$$a : X \to \mathbf{M} \times X,$$

яке можна однозначно розкласти на безпосередньо потік повідомлень, при чому кожному стану системи буде поставлено у відповідність повідомлення

$$a^{\mathrm{out}} : X \to \mathbf{M}$$

та потік переходів між станами системи, який можна формулювати як від-

ображення з множини станів системи у ту ж саму множину станів системи

$$a^{\mathrm{tr}} : X \to X,$$

так, що для кожного переходу ми можемо отримати декомпозицію

$$a = \langle a^{\mathrm{out}}, a^{\mathrm{tr}} \rangle$$

.

Окремо слід зазначити, що в категорії детермінованих систем з виходом є фінальна система.

Цю систему можна формально описати наступним чином

$$\underline{\boldsymbol{\nu}\boldsymbol{\mathcal{S}}_{\mathbf{M}}} = \mathbf{M}^{\omega} \tag{3.3}$$

$$\boldsymbol{\nu}\boldsymbol{\mathcal{S}}_{\mathbf{M}} = \lambda \langle m_0, m_1, m_2, \ldots \rangle \,.\, (m_0, \langle m_1, m_2, \ldots \rangle) : \mathbf{M}^{\omega} \to \mathbf{M} \times \mathbf{M}^{\omega} \tag{3.4}$$

і для будь-якої $\boldsymbol{\mathcal{S}}_{\mathbf{M}}$-системи $\boldsymbol{a}$, $x \in \underline{\boldsymbol{a}}$ наступна формула визначає відповідний анаморфізм

$$[\![\boldsymbol{a}]\!] x = \Big\langle \boldsymbol{a}^{\mathrm{out}}\big(\big(\boldsymbol{a}^{\mathrm{tr}}\big)^{k} x\big) \mid k \in \mathbb{N} \Big\rangle.$$

Беручи до уваги специфіку категорії **Set** і те, що $\boldsymbol{\mathcal{S}}_{\mathbf{M}}$ зберігає декартові квадрати, можна стверджувати, що бісимуляція систем із виходом $\boldsymbol{a}$ і $\boldsymbol{b}$ визначається відношенням

$$R \subset \underline{\boldsymbol{a}} \times \underline{\boldsymbol{b}}$$

таким, що $(x, y) \in R$ забезпечує

$$\boldsymbol{a}^{\mathrm{out}} x = \boldsymbol{b}^{\mathrm{out}} y$$

, а також

$$(\boldsymbol{a}^{\mathrm{tr}} x, \boldsymbol{b}^{\mathrm{tr}} y) \in R$$

.

3.2 Підходи до впровадження стохастичності у детерміновану модель за допомогою монади Джирі

Умова повної детермінованості для системи є доволі сильною. На практиці вона майже ніколи не виконується, коли мова йде про складні системи. Це пов'язано з багатьма факторами, серед яких можна виділити наступні

1. Асинхронна робота системи. До подібного класу систем можна віднести системи, в якій компоненти працюють незалежно один від одного без глобального синхронізуючого сигналу. Такі системи використовують локальні сигнали для управління взаємодією між компонентами, що дозволяє їм функціонувати паралельно та незалежно. Таким чином система у принципі не збирає у єдиному місці інформацію про свій поточний стан.

2. Високий ступінь розподіленості системи. До таких систем можна віднести системи, які працюють синхронно, але мають велику кількість елементів. Таким чином, можливий випадок, коли для того щоб зібрати повну інформацію про поточний стан системи необхідно більше часу ніж для того щоб система змінила свій стан.

У наступному розділі будуть окремо наведені детальні приклади виникнення подібних ситуацій, е використання детермінованої моделі неможливі. Для того щоб мати змогу їх моделювати необхідно наділити дискретну систему властивістю стохастичності або рандомності. Таким чином така модель буде зберігати усі основні властивості, які були описані вище для детермінованої системи, але буде змого оброблювати стохастичні події, які можуть бути визвані нечіткістю визначення поточного стану системи.

Звичайним способом переходу від детермінованої моделі до випадкової є застосування процедури рандомізації з використанням монади Джирі. Монада Джирі використовує структуру монад для роботи з розподілами

ймовірностей. Вона надає способи для композиції ймовірнісних обчислень, що дозволяє створювати складні розподіли на основі простіших.

Формально для можна сформулювати роботу монади Джирі для моделей, створених з використанням універсальних коалгебр за допомогою ендофунктора скінченного розподілу, визначеного [67] наступним чином

$$\boldsymbol{\mathcal{D}}X = \Big\{ p : X \xrightarrow{0,1}\!\!| \; p(x) \neq 0 \text{ тільки для скінченних } x \in X \text{ та } \sum_{x\in X} p(x) = 1 \Big\} \quad \text{для будь-якої множини } X \tag{3.5}$$

$$\boldsymbol{\mathcal{D}}(X \xrightarrow{f} Y) = \lambda p \in \boldsymbol{\mathcal{D}}X \,.\, \lambda y \in Y \,.\, \sum_{x\in X} [fx = y] \cdot p(x) \quad \text{for any sets } X, Y. \tag{3.6}$$

Тут ми використовуємо дужки Айверсона $[P]$ (див. [68]), які для пропозиції P дорівнюють 1, якщо виконується P, і 0 інакше.

Змістовно $\boldsymbol{\mathcal{D}}f$ можна інтерпретувати як ймовірність отримання конкретного $y \in Y$ при застосуванні відображення f без будь-якої інформації про вихідний $x \in X$.

Зауваження 3.1 (3.5) *і* (3.6) *визначають ендофунктор категорії* **Set**.

Доведення. Для будь-якої множини X, маємо

$$(\boldsymbol{\mathcal{D}}\,\mathrm{id}_X)p = \lambda x \in X \,.\, p(x) = p$$

тобто $\boldsymbol{\mathcal{D}}\,\mathrm{id}_X = \mathrm{id}_{\boldsymbol{\mathcal{D}}X}$. Далі для $X \xrightarrow{f} Y \xrightarrow{g} Z$,

$$\begin{aligned}
\big((\boldsymbol{\mathcal{D}}g)(\boldsymbol{\mathcal{D}}f)\big)p = (\boldsymbol{\mathcal{D}}g)\big((\boldsymbol{\mathcal{D}}f)p\big) &= (\boldsymbol{\mathcal{D}}g)\left(\lambda y \in Y \,.\, \sum_{x\in X|fx=y} p(x)\right) \\
&= \lambda z \in Z \,.\, \sum_{y\in Y|g(y)=z} \left(\lambda y \in Y \,.\, \sum_{x\in X|fx=y} p(x)\right)(y) \\
&= \lambda z \in Z \,.\, \sum_{y\in Y|g(y)=z} \left(\sum_{x\in X|fx=y} p(x)\right) = \lambda z \in Z \,.\, \sum_{x\in X|g(f(x))=z} p(x) \\
&= \big(\boldsymbol{\mathcal{D}}(gf)\big)p.
\end{aligned}$$

Таким чином, $\boldsymbol{\mathcal{D}}$ є ендофунктором категорії **Set**. ■

Наведене визначення ендофунктора дозволяє моделювати рандомні системи на основі коалгебраїчного підходу. Цей спосіб є узагальненням техніки моделювання для детермінованих систем з виходом.

3.3 Визначення універсальної коалгебри для Рандомної системи з вихідними даними

Системи цього типу поводяться неоднозначно як щодо вибору виходу, так і щодо зміни стану, на відміну від наведених вище. Але, слід зазначити що у рамках поточного дослідження ця неоднозначність обмежена кінцевим набором пар станів та виходів. Крім того, припускається, що реалізація кожної з дійсних пар залежить від деякого попередньо визначеного розподілу ймовірностей.

Рандомну систему з виходими можна формалізувати як коалгебраїчну систему, визначену ендфунктором $\boldsymbol{\mathcal{R}}_{\mathbf{M}} = \boldsymbol{\mathcal{D}} \circ \boldsymbol{\mathcal{S}}_{\mathbf{M}} : \mathbf{Set} \to \mathbf{Set}$, тобто.

$$\boldsymbol{\mathcal{R}}_{\mathbf{M}} X = \boldsymbol{\mathcal{D}}(\mathbf{M} \times X) \qquad \text{для будь-якої множини } X \tag{3.7}$$

$$\boldsymbol{\mathcal{R}}_{\mathbf{M}}(X \xrightarrow{f} Y) = \boldsymbol{\mathcal{D}}(\mathrm{id}\,\mathbf{M} \times f) \qquad \text{для будь-якої множини } X, Y \tag{3.8}$$

Означення 3.1 *Нехай X є набором множин станів системи, тоді відображення $\boldsymbol{\xi} : X \to \boldsymbol{\mathcal{R}}_{\mathbf{M}} X$ нижче називається випадковою системою з виходом. У цьому випадку множина X називається носієм $\boldsymbol{\xi}$ і позначається як $\underline{\boldsymbol{\xi}}$.*

Зауваження 3.2 *Для рандомної системи з результатом $\boldsymbol{\xi} : X \to \boldsymbol{\mathcal{R}}_{\mathbf{M}} X$ ми будемо використовувати наступне позначення для більшої виразності*

$$\Pr\left(x \xrightarrow[\boldsymbol{\xi}]{m} x'\right) = (\boldsymbol{\xi} x)(m, x') \quad \textit{де } x, x' \in X, \ \textit{і } m \in \mathbf{M}.$$

Приклад 3.1 (Як виникає рандомність?) *Припустимо, що двовимірна точка решітки є повною специфікацією поточного стану систе-*

ми.Система переходить з поточного стану в сусідній стан, розташований на північ, захід, південь або схід від поточного, за такт дискретного часу. При проходженні система надсилає сповіщення про напрямок відповідного переходу (північ, захід, південь чи схід).

Отже, коли відомо початкове розташування системи та послідовність її повідомлень, можна визначити поточне розташування системи. Цей метод визначення розташування системи іноді використовується замість безпосереднього вимірювання розташування системи для керування системою. На жаль, системне сповіщення може бути випадково спотворене на каналі зв'язку. У цьому випадку блок управління отримує дані, що призводить до помилки в обчисленні поточного розташування системи. Зверніть увагу, що в цьому та наступних прикладах ми не обмежуємо джерела «випадковості» ненадійним зв'язком. Просто сам приклад пов'язаний саме з виникненням некоректного переходу.

Більш формально, простір станів системи $\mathbb{Z}^2$*, набір* M *повідомлень* $\{\mathbf{n}, \mathbf{w}, \mathbf{s}, \mathbf{e}\}$*, де* $\mathbf{n}$*-північ,* $\mathbf{w}$*-захід,* $\mathbf{s}$*-південь,* $\mathbf{e}$*-схід.*

Припустимо, що E *є матрицею* 4×4*. Його рядки та стовпці індексуються елементами множини* M*, а елемент* $E_{m'm''}$ *є ймовірністю отримати* m''*, якщо реальний напрямок проходження* m'*.*

Припустимо також, що $\boldsymbol{a} : \mathbb{Z}^2 \to \boldsymbol{\mathcal{S}}_{\mathsf{M}}\mathbb{Z}^2$ *є детермінованою системою з виходом, і визначте*

$$\Pr\left(x \xrightarrow[\boldsymbol{\xi}]{m'} x'\right) = E_{\boldsymbol{a}^{\mathrm{out}}x\ m'} \cdot [\boldsymbol{a}^{\mathrm{tr}}x = x'].$$

Тут квадратні дужки (звані дужками Айверсона), застосовані до вкладеного логічного виразу, дають 1*, якщо вираз оцінюється як істина, і 0 в іншому випадку.*

Зверніть увагу, що $\boldsymbol{\xi}$ *моделює ситуацію детермінованої системи з спотвореним виходом.*

Приклад 3.2 (Більш загальний випадок) *Нехай* M *— деякий набір повідомлень,* $\boldsymbol{a}$ *—* $\boldsymbol{\mathcal{S}}_{\mathsf{M}}$*-система, а* $E : \mathsf{M}\times\mathsf{M} \xrightarrow{0,1}$ *так, що* $\sum\limits_{m'\in\mathsf{M}} E(m, m') = 1$

для кожного $m \in \mathbf{M}$. *Потім*

$$\Pr\left(x \xrightarrow[\boldsymbol{\xi}]{m} x'\right) = E(\boldsymbol{a}^{\mathrm{out}}x, m) \cdot [\boldsymbol{a}^{\mathrm{tr}}x = x'] \qquad \text{де } m \in \mathbf{M} \text{ і } x, x' \in \underline{\boldsymbol{a}}$$

визначає випадкову систему з результатом $\boldsymbol{\xi}$ *наступним чином*

$$(\boldsymbol{\xi}x)(m, x') = E(\boldsymbol{a}^{\mathrm{out}}x, m) \cdot [\boldsymbol{a}^{\mathrm{tr}}x = x'].$$

Дійсно, набір пар $(x', m) \in \mathbf{M} \times \underline{\boldsymbol{a}}$ *складається щонайбільше з такої ж кількості елементів, як* $\mathbf{M}$ *і*

$$\sum_{m \in \mathbf{M}} \sum_{x' \in \underline{\boldsymbol{a}}} (\boldsymbol{\xi}x)(m, x') = \sum_{m \in \mathbf{M}} \sum_{x' \in \underline{\boldsymbol{a}}} E(\boldsymbol{a}^{\mathrm{out}}x, m) \cdot [\boldsymbol{a}^{\mathrm{tr}}x = x'] = 1.$$

Означення 3.2 *Для систем із виходом* $\boldsymbol{\xi}$ *і* $\boldsymbol{\eta}$ *відображення* $f : \underline{\boldsymbol{\xi}} \to \underline{\boldsymbol{\eta}}$ *є* $\boldsymbol{\mathcal{R}}_{\mathbf{M}}$*-морфізм з* $\boldsymbol{\xi}$ *в* $\boldsymbol{\eta}$ *якщо*

$$\Pr\left(fx \xrightarrow[\boldsymbol{\eta}]{m} y\right) = \sum_{x' \in \underline{\boldsymbol{\xi}}} [fx' = y] \cdot \Pr\left(x \xrightarrow[\boldsymbol{\xi}]{m} x'\right) \tag{3.9}$$

де $m \in \mathbf{M}$ *та* $y \in \underline{\boldsymbol{\eta}}$.

Зауважте, що перевірка (3.9) визначає той самий клас відображень, що й клас $\boldsymbol{\mathcal{R}}_{\mathbf{M}}$-морфізмів, є легкою вправою.

Зауваження 3.3 *Якщо* $\boldsymbol{\mathcal{R}}_{\mathbf{M}}$*-морфізм* $f : \boldsymbol{\xi} \to \boldsymbol{\eta}$ *ін'єктивний, то* (3.9) *еквівалентний наступному закону збереження*

$$\Pr\left(fx \xrightarrow[\boldsymbol{\eta}]{m} fx'\right) = \Pr\left(x \xrightarrow[\boldsymbol{\xi}]{m} x'\right)$$

where $x, x' \in \underline{\boldsymbol{\xi}}$ *and* $m \in \mathbf{M}$.

3.4 Визначення фінальної коалгебри для рандомної систем

У цьому розділі ми будуємо важливі приклади випадкової системи з виходом і сімейства $\boldsymbol{\mathcal{R}}_{\mathbf{M}}$-морфізмів з будь-якої $\boldsymbol{\mathcal{R}}_{\mathbf{M}}$-системи в сконструйовану. Наша гіпотеза полягає в тому, що сконструйована випадкова система з виходом $\boldsymbol{\mathcal{R}}_{\mathbf{M}}$ є кінцевою $\boldsymbol{\mathcal{R}}_{\mathbf{M}}$-системою, а сконструйована сім'я $\boldsymbol{\mathcal{R}}_{\mathbf{M}}$-морфізмів є сімейством відповідних анаморфізмів.

3.4.1 Концепція помічених дерев без листя

Загальною практикою для моделювання індетермінованих динамічних систем є використання деревних структур [69]. Така практика зумовлена багатьма причинами.

У [66] автори пояснюють, чому коалгебраїчний підхід є плідним для визначення семантичних моделей для мов програмування та яка роль кінцевих об'єктів у відповідних категоріях коалгебр.

Неформально кажучи, кінцева коалгебра є ключовим інструментом для розуміння обсягу загальних властивостей, що стосуються всіх систем, пов'язаних з деяким ендофунктором. Цей інструмент дозволяє дивитися на динамічну систему, так би мовити, з нескінченно віддаленої точки часу.

Природно, що для індетермінованих систем необхідно розглянути всі можливі способи поведінки системи в майбутньому. Отже, якщо кількість варіантів локального вибору для кожного стану системи є кінцевою, кінцево розгалужене дерево є природним інструментом для моделювання поведінки системи.

Перш за все, розглянемо множину $\overline{\mathsf{T}}_{\mathbf{M}}$, що складається з часткових відображень виду $t : \mathbb{N}^+ \dashrightarrow \mathbf{M} \times [0,1]$ такий, що

1. для деяких $n \in \mathbb{N}$, $\langle n \rangle \in \operatorname{dom} t$;
2. для будь-яких $\boldsymbol{n} \in \mathbb{N}^*$,
   (а) $\{n \in \mathbb{N} \mid \langle \boldsymbol{n}, n \rangle \in \operatorname{dom} t\} = k)$ для деяких $k \in \mathbb{N}$;
   (б) для деяких $n \in \mathbb{N}$, $\langle \boldsymbol{n}, n \rangle \in \operatorname{dom} t$ будь-коли $\boldsymbol{n} \in \operatorname{dom} t$;
   (в) $\boldsymbol{n} \in \operatorname{dom} t$ whenever $\langle \boldsymbol{n}, n \rangle \in \operatorname{dom} t$ для деяких $n \in \mathbb{N}$;
   (г) $\sum_{n=0}^{\infty} \left[\langle \boldsymbol{n}, n \rangle \in \operatorname{dom} t\right] \cdot \operatorname{pr}_{[0,1]}(t\langle \boldsymbol{n}, n \rangle) = 1$ де $\operatorname{pr}_{[0,1]} : \mathbf{M} \times [0,1] \xrightarrow{0,1} \epsilon$ проєкцією з $\mathbf{M} \times [0,1]$ в $[0,1]$.

Пояснимо, чому $t \in \overline{\mathsf{T}}_{\mathbf{M}}$ визначає деяке непорожнє дерево без листків, позначене елементами $\mathbf{M} \times [0,1]$. Дерево (N_t, E_t), що відповідає $t \in \overline{\mathsf{T}}_{\mathbf{M}}$, будується наступним чином

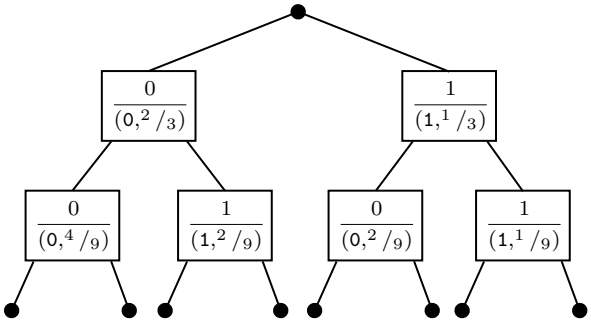


Рис. 3.1. Дерево, що відповідає прикладу 3.3

- набір вузлів N_t дерева $\{\langle\rangle\} \cup \operatorname{dom} t$;
- набір ребер E_t дерева

$$\{(\boldsymbol{n}', \boldsymbol{n}'') \in N_t \times N_t \mid \boldsymbol{n}'' = \langle \boldsymbol{n}', n\rangle \text{ для деяких } n \in \mathbb{N}\};$$

- позначка вузла $\langle \boldsymbol{n}, n\rangle$ is $t\langle \boldsymbol{n}, n\rangle$.

Легко перевірити, що за вимогами 1, 2.a–2.d побудований граф є деревом, кожен вузол якого, за винятком кореня дерева, позначений елементами з $\overline{\mathsf{T}}_{\mathsf{M}}$.

Приклад 3.3 *Let* $\mathsf{M} = \{\mathtt{0}, \mathtt{1}\}$ *та*

$$t\langle n\rangle = \begin{cases} (\mathtt{0}, {}^{2}/_{3}) & \text{if } n = 0 \\ (\mathtt{1}, {}^{1}/_{3}) & \text{if } n = 1 \\ \text{не визначено} & \text{if } n > 1 \end{cases}$$

$$t\langle \boldsymbol{n}, n\rangle = \begin{cases} \left(\mathtt{0}, {}^{2}/_{3} \cdot \operatorname{pr}_{[0,1]}(t\langle \boldsymbol{n}\rangle)\right) & \text{if } t\langle n\rangle \text{ визначено та } n = 0 \\ \left(\mathtt{1}, {}^{1}/_{3} \cdot \operatorname{pr}_{[0,1]}(t\langle \boldsymbol{n}\rangle)\right) & \text{if } t\langle n\rangle \text{ визначено та } n = 1 \\ \text{не визначено} & \text{в іншому випадку} \end{cases}$$

де $\boldsymbol{n} \in \mathbb{N}^*$ *і* $n \in \mathbb{N}$ *то відповідне дерево показано на Рис. 3.1.*

Приклад 3.1 показує, що деякі симетрії можуть бути властивими елементам $\overline{\mathsf{T}}_{\mathsf{M}}$. Дійсно, якщо ми поміняємо місцями піддерево, що відповідає

вузлу 0, на піддерево, що відповідає вузлу 1, тобто застосувамо транспозицію $(0,1)$ до ідентифікаторів вузлів, які утворюють перший рівень дерева, то перетворений і оригінальний дерева мають однакову геометрію. У цьому випадку у нас немає підстав розрізняти ці два дерева.
Рис. 3.2 представляє ілюстрацію цього зауваження.

Зауваження 3.4 *Під час нашого дослідження ми виявили, що послідовність міток не ідентифікує вузол однозначно, тому ми були змушені запровадити додаткову нумерацію. Більш детально цей метод описано нижче.*

У математиці канонічним інструментом для ідентифікації елементів множини є відношення еквівалентності на цій множині. Таким чином, ми приходимо до наступного коіндуктивного визначення.

Означення 3.3 *Відношення подібності на* $\overline{\mathbf{T}}_{\mathbf{M}}$ *є найменшою еквівалентністю* $\simeq_S$ *такою, що для будь-якого* $t', t'' \in \overline{\mathbf{T}}_{\ objM}$, $t' = t''$ *або*

1. $\{n \in \mathbb{N} \mid \langle n \rangle \in \operatorname{dom} t'\} = [k) = \{n \in \mathbb{N} \mid \langle n \rangle \in \operatorname{dom} t''\}$ *and*
2. *існує така перестановка* $\sigma \in S_k$, *що* t'_n *і* $t''_{\sigma(n)}$ *еквівалентні, де* S_k *— симетрична група на* k *символів,* $0 \leq n < k$, $t'_n = \lambda \boldsymbol{n} \,.\, t'\langle n, \boldsymbol{n} \rangle$ *і* $t''_n = \lambda \boldsymbol{n} \,.\, t''\langle n, \boldsymbol{n} \rangle$, *які, очевидно, є елементами* $\overline{\mathbf{T}}_{\mathbf{M}}$.

Незважаючи на свою елегантність, Визначення 3.3 не дає корисного інструменту для встановлення еквівалентності $t', t'' \in \overline{\mathbf{T}}_{\mathbf{M}}$. Тому наведемо більш громіздке, але більш ефективне визначення.

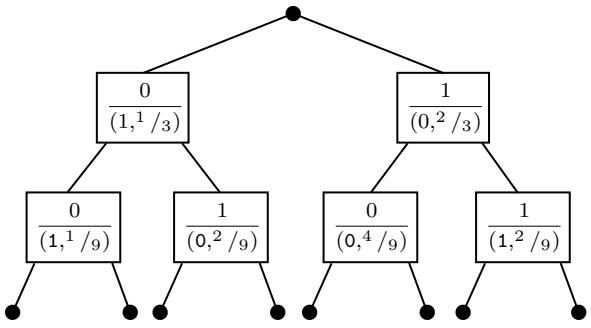


Рис. 3.2. Інше дерево для прикладу 3.3

Перш за все, визначимо повну решітку $\mathcal{E}$ еквівалентностей на $\overline{\mathsf{T}}_{\mathsf{M}}$ і наступний оператор $F : \mathcal{E} \to \mathcal{E}$:

$$
\begin{array}{ll}
\forall & E \in \mathcal{E} \\
\text{та} & t', t'' \in \overline{\mathsf{T}}_{\mathsf{M}},\ t' \simeq_{F(E)} t'' \text{ якщо} \\
& t' \simeq_E t'' \text{ або для деяких } k \in \mathbb{N}, \\
& \quad \{n \in \mathbb{N} \mid \langle n \rangle \in \operatorname{dom} t'\} = [k) = \{n \in \mathbb{N} \mid \langle n \rangle \in \operatorname{dom} t''\} \text{ та} \\
& \quad \text{існує } \sigma \in S_k \text{ такі, що } t'_n \simeq_E t''_n \text{ щоразу } 0 \leq n < k.
\end{array}
$$

Щоб перевірити, що F є правильним оператором на $\overline{\mathsf{T}}_{\mathsf{M}}$, потрібно показати, що $F(E)$ є еквівалентністю. Але це цілком очевидно завдяки конструкції.

Зверніть увагу, що відношення подібності є найменшою фіксованою точкою оператора F через визначення 3.3. Теорема Тарського про фіксовану точку [70] забезпечує існування відношення подібності відразу, оскільки цей оператор є монотонним. Щоб побудувати відношення подібності, визначимо послідовність еквівалентності наступним чином

$$
\begin{aligned}
E^{(0)} &= \text{ відношення рівності на} \overline{\mathsf{T}}_{\mathsf{M}} \\
E^{(n+1)} &= F(E^{(n)}) \qquad\qquad \text{для будь-якого } n \in \mathbb{N}.
\end{aligned}
$$

За побудовою F визначена послідовність є неспадною, тобто

$$E^{(0)} \subset E^{(1)} \subset \ldots \subset E^{(n)} \subset \ldots \subset E^{(\infty)} = \bigcup_{n=0}^{\infty} E^{(n)}.$$

Очевидно, що $F(E^{(\infty)}) = E^{(\infty)}$ і для будь-якої фіксованої точки E^* $E^{(\infty)} \subset E^*$.

Таким чином, $E^{(\infty)}$ є відношенням подібності.

Нижче ми використовуємо позначення $t' \simeq t''$ для посилання на твердження "t' і t'' схожі".

Щоб пояснити структуру відношення подібності, введемо $\mathrm{nch}_t : N_t \dashrightarrow \mathbb{N}_+$ для будь-якого $t \in \overline{\mathsf{T}}_{\mathsf{M}}$ наступним чином:

$$\{n \in \mathbb{N} \mid \langle \boldsymbol{n}, n \rangle \in \operatorname{dom} t\} = [\mathrm{nch}_t\, \boldsymbol{n})$$

щоразу $\boldsymbol{n} \in N_t$.

Далі, нехай S_t утворено $\sigma = \langle \sigma_{\boldsymbol{n}} \in S_{\mathrm{nch}_t \boldsymbol{n}} \mid \boldsymbol{n} \in N_t \rangle$ тоді

$$\langle n \rangle^\sigma = \langle \sigma_{\langle\rangle} n \rangle$$
$$\langle \boldsymbol{n}, n \rangle^\sigma = \langle \boldsymbol{n}^\sigma, \sigma_{\boldsymbol{n}} n \rangle$$

для будь-якого $\boldsymbol{n} \in N_t$ and $0 \leq n < \mathrm{nch}_t \boldsymbol{n}$.

Також для $t \in \overline{\mathsf{T}}_{\mathsf{M}}$ and $\sigma \in S_t$, $t^\sigma = \lambda \boldsymbol{n} \,.\, t\, \boldsymbol{n}^\sigma \in \overline{\mathsf{T}}_{\mathsf{M}}$.

Тепер давайте визначимо $[t] \subset \overline{\mathsf{T}}_{\mathsf{M}}$ наступним чином $[t] = \{t^\sigma \mid \sigma \in S_t\}$.

Це також можна легко перевірити $[t'] = [t'']$ або $[t'] \cap [t''] = \emptyset$ для будь-якого $t', t'' \in \overline{\mathsf{T}}_{\mathsf{M}}$.

Іншими словами, ми визначили розбиття $\overline{\mathsf{T}}_{\mathsf{M}}$ і, отже, відношення еквівалентності між елементами $\overline{\mathsf{T}}_{\mathsf{M}}$.

Таким чином, $t', t'' \in \overline{\mathsf{T}}_{\mathsf{M}}$ подібні, якщо $[t'] = [t'']$.

Означення 3.4 *Ми будемо використовувати* T_{M} *для позначення фактор-множини* $\overline{\mathsf{T}}_{\mathsf{M}}$ *w.r.t. подібність, а його елементи можна інтерпретувати як абстрактні непорожні помічені дерева без листя.*

3.4.2 Побудова фінальної системи

З побудови $t \in \overline{\mathsf{T}}_{\mathsf{M}}$ очевидно, що для будь-якого $n \in \mathbb{N}$ такого, що $\langle n \rangle \in \mathrm{dom}\, t$, відображення $t_n = \lambda \boldsymbol{n} \,.\, t\langle n, \boldsymbol{n} \rangle \in \overline{\mathsf{T}}_{\mathsf{M}}$ і що набір таких n є кінцевий. Отже, для цього t можна визначити $p_t : \mathsf{M} \times \overline{\mathsf{T}}_{\mathsf{M}} \xrightarrow{0,1}$ наступним чином

$$p_t(m, t') = \sum_{n=0}^{\infty} \left[\langle n \rangle \in \mathrm{dom}\, t \wedge t' = t_n \wedge m = \mathrm{pr}_{\mathsf{M}}(t\langle n \rangle) \right] \cdot \mathrm{pr}_{[0,1]}(t\langle n \rangle)$$

де $\mathrm{pr}_{\mathsf{M}} : \mathsf{M} \times [0,1] \to \mathsf{M}$ є проекцією $\mathsf{M} \times [0,1]$ на M.

Звичайно, p_t визначає кінцевий розподіл ймовірностей елементів $\mathsf{M} \times \overline{\mathsf{T}}_{\mathsf{M}}$.

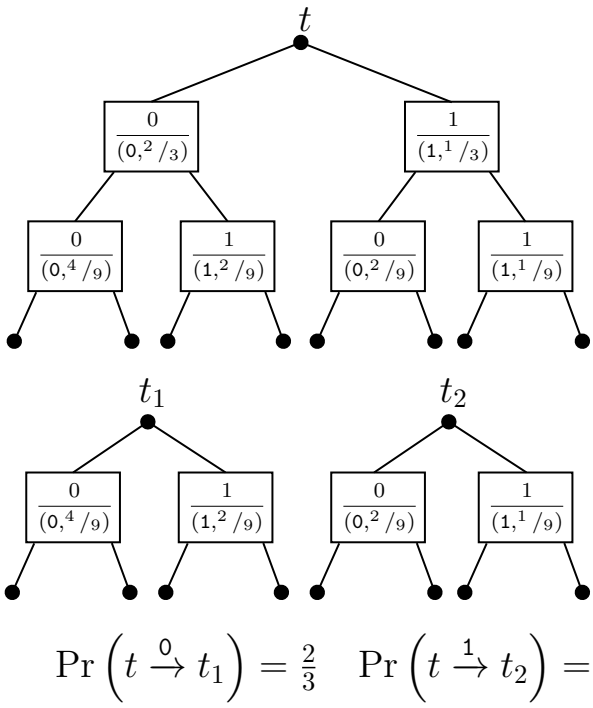


Рис. 3.3. Ілюстрація конструкції

Дійсно,

$$\begin{aligned}
&\sum_{m\in\mathbf{M}}\sum_{t'\in\overline{\mathsf{T}}_{\mathbf{M}}} p_t(m,t') \\
&\quad= \sum_{m\in\mathbf{M}}\sum_{t'\in\overline{\mathsf{T}}_{\mathbf{M}}}\sum_{n=0}^{\infty}\left[\langle n\rangle \; in \operatorname{dom} t \wedge t' = t_n \wedge m = \mathrm{pr}_{\mathbf{M}}(t\langle n\rangle)\right]\cdot \mathrm{pr}_{[0,1]}(t[n]) \\
&\quad= \sum_{n=0}^{\infty}[\langle n\rangle \in \operatorname{dom} t]\cdot \sum_{m\in\mathbf{M}}\left[m = \mathrm{pr}_{\mathbf{M}}(t\langle n\rangle)\right]\cdot \mathrm{pr}_{[0,1]}(t\langle n\rangle) \\
&\quad= \sum_{n=0}^{\infty}[\langle n\rangle \in \operatorname{dom} t]\cdot \mathrm{pr}_{[0,1]}(t\langle n\rangle) = 1 \quad \text{через (2г) з } \boldsymbol{n} = \langle\rangle.
\end{aligned}$$

Приклад 3.4 *Давайте проілюструємо цю концепцію за допомогою дерева t із прикладу 3.3 (див. Рис. 3.3).*

Тепер зауважимо, що запропонована конструкція задовольняє наступну умову, якщо $t, s \in \overline{\mathsf{T}}_{\mathbf{M}}$ і $[t] = [s]$ (тобто $s = t^\sigma$ для деякого $\sigma \in S_t$), тоді

1. $\{n \in \mathbb{N} \mid \langle n\rangle \in N_s\} = \{\sigma_{\langle\rangle} n \mid \langle n\rangle \in N_t\}$;

2. $s_{\sigma_{\langle\rangle} n}$ подібний до t_n для будь-якого $n \in \mathbb{N}$ такого, що $\langle n\rangle \in N_t$; і

3. для будь-якого $0 \leq n < \mathrm{nch}_t\langle\rangle$,

$$\Pr\left(t \xrightarrow{\mathrm{pr}_{\mathbf{M}}(t\langle n\rangle)} t_n\right) = \mathrm{pr}_{[0,1]}(t[n])$$
$$= {}'pr_{[0,1]}(s[\sigma n]) = \Pr\left(s \xrightarrow{\mathrm{pr}_{\mathbf{M}}(s\langle\sigma_{\langle\rangle} n\rangle)} s_{\sigma_{\langle\rangle} n}\right).$$

Ця умова гарантує, що значення p_t не зміниться при заміні t будь-яким подібним до нього елементом .
Іншими словами, можна визначити випадкову систему з виходом $\boldsymbol{\mu}$ наступним чином

$$\underline{\boldsymbol{\mu}} = \mathbf{T}_{\mathbf{M}}$$
$$\Pr\left([t] \xrightarrow[\boldsymbol{\mu}]{m} [t']\right) = \sum_{m' \in \mathbf{M}} \sum_{n=0}^{\infty} \left[m' = m \wedge [t_n] = [t']\right] \cdot \mathrm{pr}_{[0,1]}(t\langle n\rangle).$$

3.4.3 Побудова родини морфізмів

Тепер ми готові побудувати індексоване сімейство

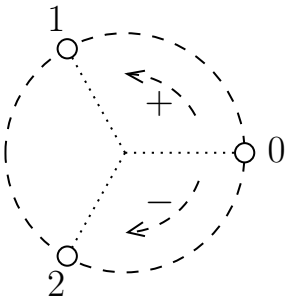
$$\boldsymbol{\omega} = \{\boldsymbol{\omega}_{\boldsymbol{\xi}} : \boldsymbol{\xi} \to \boldsymbol{\mu} \mid \boldsymbol{\xi} \in \boldsymbol{\mathcal{R}}_{\mathbf{M}}\text{-система}\}$$

$\boldsymbol{\mathcal{R}}_{\mathbf{M}}$-морфізмів. Нижче ми пояснимо конструкцію на прикладі, а потім визначимо її в загальному випадку.

Зауваження 3.5 *Для ілюстрації методу побудови сімейства морфізмів можна використати описані вище приклади, але вони не настільки виразні. Тому ми наводимо новий приклад, який вважаємо більш доречним.*

Приклад 3.5 *Розглянемо випадкову систему $\boldsymbol{\pi}$ з виходом із набору $\{+, -\}$, яка показана на Рис. 3.4.*

Цю систему можна розглядати як систему ротора з трьома положеннями, яка обертається навколо свого центру на $120°$ проти (перехід від x до $x+1 \mod 3$) або за годинниковою стрілкою (перехід від x в $x-1 \mod 3$) за умови, що напрямки обертання є рівноімовірними. Система також надсилає повідомлення про реалізований напрямок обертання, але



$$\Pr\left(x \xrightarrow[\boldsymbol{\pi}]{+} (x+1 \bmod 3)\right) = \frac{1}{3}$$
$$\Pr\left(x \xrightarrow[\boldsymbol{\pi}]{-} (x+1 \bmod 3)\right) = \frac{1}{6}$$
$$\Pr\left(x \xrightarrow[\boldsymbol{\pi}]{+} (x-1 \bmod 3)\right) = \frac{1}{6}$$
$$\Pr\left(x \xrightarrow[\boldsymbol{\pi}]{-} (x-1 \bmod 3)\right) = \frac{1}{3}$$

Рис. 3.4. Система $\boldsymbol{\pi}$, розглянута в прикладі 3.5

ці повідомлення правильні з імовірністю $^2/_3$ *і неправильні з імовірністю* $^1/_3$*. Формальне визначення системи представлено на Рис. 3.4.*

Тепер можна побудувати дерево, використовуючи відношення "батько-діти показане на Рис. 3.5. Кожен вузол на малюнку позначено трійкою $\frac{s}{(m,p)}$*, де* $s \in \underline{\boldsymbol{\pi}}$*, m є елементом набору повідомлень* $\{+,-\}$*, а* $0 < p \leq 1$ *— це ймовірність переходу від батьківського вузла до розглянутого вузла.*

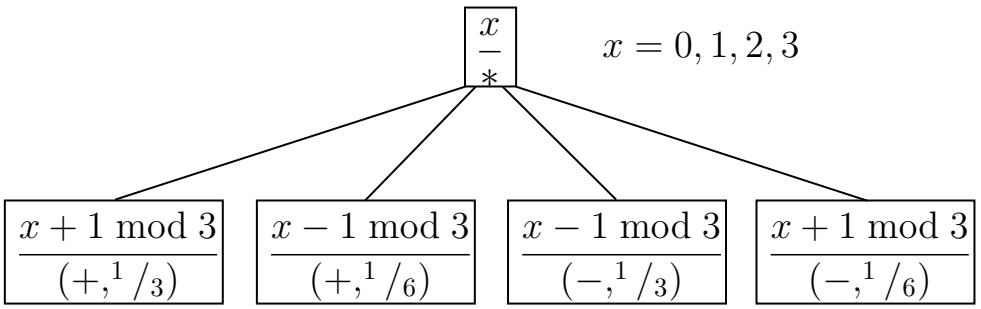


Рис. 3.5. Відношення "батько-діти"для дерева, що відповідає системі $\boldsymbol{\pi}$

Якщо розглядати вузли та ребра між ними, беручи до уваги лише стани системи, що відповідають цим вузлам, то ми отримаємо дерево, показане на Рис. 3.6.

Аналізуючи це дерево станів, можна побачити наявність залежностей, характерних для системи $\boldsymbol{\pi}$*. Враховуючи нашу мету, а саме побудувати остаточну випадкову систему з виходом, нам потрібно усунути залежності такого роду. Для цього ми використовуємо перетворення дерева, показане на Рис. 3.7. Звичайно, результат цього перетворення залежить від обраного методу перерахування вузлів, але клас подібності цього результату не залежить від вибору методу перерахування.*

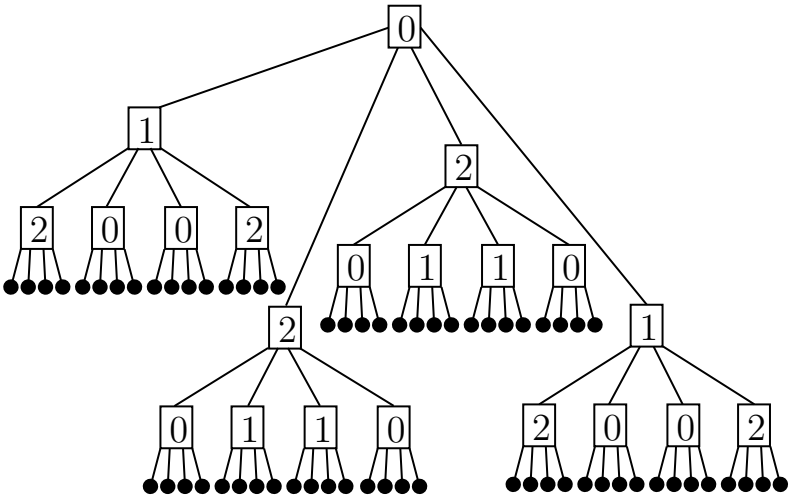


Рис. 3.6. Дерево станів для системи $\boldsymbol{\pi}$

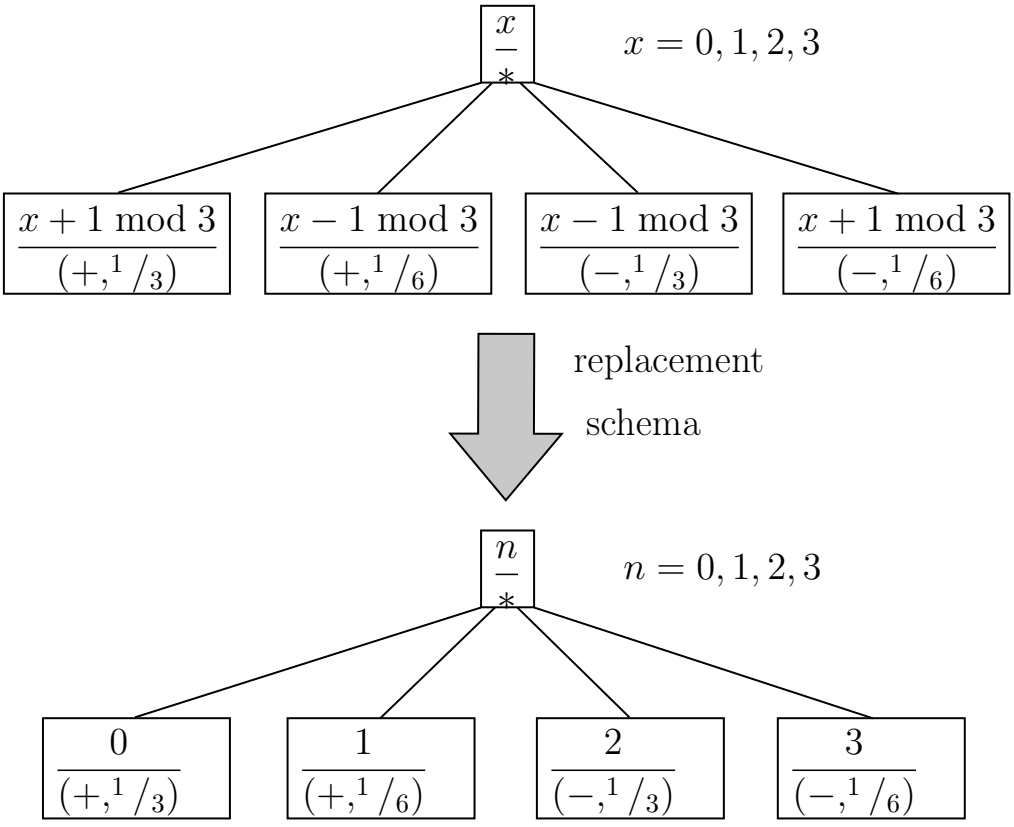


Рис. 3.7. Перетворення системи $\boldsymbol{\pi}$ в систему $\boldsymbol{\mu}$

Отже, клас подібності дерева, побудованого з коренем, визначеним станом x, є необхідним значенням $\omega_{\boldsymbol{\pi}} x \in \mathbf{T}_{\mathbf{M}}$.

Нарешті, нам потрібно перевірити (3.9) у нашому випадку, але це вірно за допомогою побудови дерев, пов'язаних зі станами $\boldsymbol{\pi}$.

Таким чином, показано, що деревоподібна структура представлення даних добре справляється із завданнями формалізації випадкових систем з виходами.

Тепер наведемо загальну конструкцію на основі цього прикладу. Нехай $\mathbf{M}$ — кінцевий набір повідомлень, а $\boldsymbol{\xi}$ — випадкова система з виходом, тоді для побудови необхідного морфізму $\omega_{\boldsymbol{\xi}} : \boldsymbol{\xi} \to \boldsymbol{\mu}$ необхідно зробити наступне

1. для $x \in \underline{\boldsymbol{\xi}}$, перерахувати елементи скінченної множини
$$\left\{(m, x') \in \mathbf{M} \times \underline{\boldsymbol{\xi}} \mid \Pr\left(x \xrightarrow[\boldsymbol{\xi}]{m} x'\right) > 0\right\} = \left\{(m_0, x_0), \ldots, (m_{n-1}, x_{n-1})\right\};$$
2. призначити $t\langle k\rangle \leftarrow \Pr\left(x \xrightarrow[\boldsymbol{\xi}]{m_k} x_k\right)$ для $0 \leq k < n$;
3. виконати кроки 1 і 2 для балів, отриманих у поточній гілці на попередньому кроці.

Цей нескінченний процес генерує $t \in \overline{\mathbf{T}}_{\mathbf{M}}$ і призначає $\omega_{\boldsymbol{\xi}} x \leftarrow [t]$.

Зауваження 3.6 *Конструкція, показана вище, досить складна, оскільки ми ввели додаткові коефіцієнти різниці для вузлів. Однак це необхідно, оскільки вузол не ідентифікується однозначно лише за допомогою траєкторії, яка постачається повідомленнями системи, і якщо ми не розрізняємо приховані стани системи, то ми виявимо, що умови для побудови морфізму не виконуються.*

3.4.4 Фінальна система для рандомної дискретної системи з виходами

Наші міркування про мічені дерева без листя дозволяють нам зробити таке припущення:

Нехай $\mathbf{M}$ — скінченний набір повідомлень, тоді для будь-якої випадкової системи з виходом $\boldsymbol{\xi}$ рівняння

$$\Pr\left(fx \xrightarrow[\boldsymbol{\mu}]{m} [t]\right) = \Pr\left(\omega_{\boldsymbol{\xi}} x \xrightarrow[\boldsymbol{\mu}]{m} [t]\right) \qquad \text{де } x \in \underline{\boldsymbol{\xi}}\,,\ m \in \mathbf{M} \text{ і } t \in \overline{\mathbf{T}}_{\mathbf{M}}$$

відносно $\boldsymbol{\mathcal{R}}_{\mathbf{M}}$-морфізму $f : \boldsymbol{\xi} \to \boldsymbol{\mu}$ має рівно один розв'язок, а саме $\omega_{\boldsymbol{\xi}}$.

Іншими словами, гіпотеза полягає в тому, що $\boldsymbol{\mu}$ є кінцевою $\boldsymbol{\mathcal{R}}_{\mathbf{M}}$-системою, а ω є відповідним сімейством анаморфізмів.

Висновки по Розділу 3

У розділі досліджені моделі дискретних детермінованих систем з вихідними даними за допомогою універсальних коалгебр. Були описані морфізми для таких систем та визначена фінальна коалгебра.

Враховуючі, що на практиці при роботі складних систем неможливо зібрати всю інформацію для поточного стану системи, а отже неможливо досягнути повної детермінованості системи були представлені підходи до впровадження стохастичності у створену модель. Було використано монаду розподілу ймовірностей Джирі для впровадження стохастичності до моделі. Такий підхід було використано вперше, що дозволило створити модель рандомної дискретної системи з виходами.

Було проведено дослідження щодо існування фінальної системи для такої моделі за допомогою властивостей ендофунктора зберігати декартові квадрати та сформульована сама фінальна система для цієї категорії систем. Вона була представлена як нескінченне дерево без листя, розмічене імовірностями переходів.

Таким чином, можна зробити висновок, що в рамках цього розділу метод синтезу моделей для динамічного аналізу (імітаційного моделювання) складних систем з використанням універсальних коалгебр дістав подальшого розвитку.

Основні положення цього розділу викладені у публікаціях автора [13].

РОЗДІЛ 4. Практичне застосування універсальних коалгебр для специфікації та аналізу поведінки систем

4.1 Особливості складних програмних систем та програмних компонентів з перспективи їх специфікації

Складна програмна система – це комплекс програмно - апаратного забезпечення, яке має великий обсяг коду, складну архітектуру, численні взаємодіючі компоненти та підсистеми. До подібних систем відносять кіберфізичні, розподілені системи та системи геолокації. Подібні рішення розробляються для виконання складних завдань і часто використовуються у великих організаціях чи інфраструктурах. Розвиток таких систем потребує інтеграції передових методів проектування, аналізу, тестування та підтримки [71].

У цьому розділі буде надано комплексний огляд складних програмних систем, включаючи їх характеристики, проблеми специфікації та аналізу, а також сучасні підходи до їх розробки та верифікації.

Основні характеристики складних програмних систем включають:

1. Масштабність. Складні програмні системи мають значний обсяг коду та велику кількість компонентів, що взаємодіють між собою.

2. Розподіленість. Складні програмні системи часто функціонують у розподілених середовищах, де різні частини системи працюють на різних апаратних платформах.

3. Інтеграція. Складні програмні системи інтегруються з іншими системами та сервісами, що ускладнює їхню архітектуру.

4. Динамічність. Складні програмні системи потребують постійного оновлення та адаптації до змінних вимог і умов експлуатації.

5. Безпека та надійність. Складні програмні системи повинні забезпечувати високий рівень безпеки та надійності, оскільки часто функціонують у критичних сферах, таких як медицина, енергетика чи фінанси.

Для успішної розробки та впровадження у виробничі процеси складних програмних систем використовуються сучасні методи та технології [72], що забезпечують ефективність і надійність системи, а саме:

1. Агентно-орієнтоване моделювання. Агентно-орієнтовані підходи дозволяють моделювати поведінку окремих компонентів системи як автономних агентів, що взаємодіють між собою. Це сприяє кращому розумінню динамічних аспектів системи.

2. Мікросервісна архітектура. Мікросервісна архітектура дозволяє розділити складну систему на менші, автономні сервіси, що можуть розроблятися, тестуватися та розгортатися незалежно один від одного. Це спрощує управління змінами та забезпечує гнучкість.

3. Формальні методи. Використання формальних методів для специфікації та верифікації СПС дозволяє забезпечити точність і надійність системи. Формальні методи, такі як методи на основі логіки та алгебри, сприяють виявленню помилок на ранніх етапах розробки.

4. Моделювання та симуляція. Моделювання та симуляція використовуються для аналізу поведінки СПС в різних умовах. Це дозволяє виявити можливі проблеми та оптимізувати систему до її фактичного впровадження.

Однак слід зазначити, що процес специфікації подібних систем є доволі комплексним у першу чергу через їх складність та високі вимоги до безпечної та безперебійної роботи. У процесі специфікації складних систем виникає численна кількість проблем, які можуть суттєво вплинути на успіх проекту. Нижче описані основні проблеми, що виникають під час специфікації складних систем:

1. Неповнота специфікацій. Специфікації складних систем часто виявляються неповними через складність охоплення всіх аспектів функціонування системи. Це призводить до пропуску важливих деталей, що можуть критично вплинути на поведінку системи в реальних умовах.

2. Неоднозначність вимог. Формулювання вимог може бути неоднозначним, що спричиняє різні тлумачення серед розробників, тестувальників та інших зацікавлених сторін. Це ускладнює верифікацію та валідацію системи.

3. Складність моделювання поведінки.Моделювання поведінки складної системи може вимагати великих зусиль через численні взаємодії між компонентами, що потребують докладного опрацювання та симуляції.

4. Непередбачувані взаємодії. Взаємодії між компонентами складної системи можуть спричиняти непередбачувані наслідки, які важко врахувати на етапі специфікації та аналізу поведінки.

5. Недостатня деталізація. Надмірно загальні специфікації не надають достатньої інформації для розробників і тестувальників, що може призводити до різночитань та помилок під час реалізації та тестування системи.

6. Взаємозалежність вимог. Багато вимог складної системи можуть бути взаємозалежними, що ускладнює їхнє розуміння та реалізацію. Зміна однієї вимоги може впливати на інші, що потребує постійного оновлення специфікацій.

7. Складність верифікації. Верифікація складних систем часто є трудомісткою через велику кількість сценаріїв використання та варіантів поведінки, які необхідно протестувати для забезпечення належної якості системи.

8. Вплив зовнішніх факторів. Поведінка складних систем може значно залежати від зовнішніх факторів, таких як навколишнє середовище, мережеві умови або інтеграція з іншими системами, що складно врахувати на етапі специфікації.

Таким чином, різні аспекти системи можуть вимагати різних підходів до специфікації. Наприклад, функціональні вимоги можуть бути зручно описані за допомогою алгебраїчних специфікацій, тоді як динамічна поведінка може потребувати використання моделей станів і переходів або коалгебраїчних методів. Крім того, аспект безпеки може вимагати використання формальних методів для доказу властивостей безпеки, а продуктивність — аналітичних моделей.

Різні специфікації можуть використовувати різні рівні абстракції, що ускладнює їх інтеграцію в єдину модель. Наприклад, високорівневі абстракції, які використовуються для опису архітектури системи, можуть бути важко узгоджені з низькорівневими деталями реалізації.

Слід зазначити, що модульний підхід до проведення специфікаці дозволяє розділити систему на менші, більш керовані частини. Однак, при спробі поєднати ці частини в єдину специфікацію, виникає проблема забезпечення консистентності між різними модулями. Різні модулі можуть використовувати різні підходи до специфікації, що ускладнює їх узгодження.

Отже, особливістю складних програмних систем є практична неможливість побудови та аналізу специфікації системи з єдиної точки зору або використанням єдиного підходу. Таким чином, є необхідність побудови комплексу моделей системи, який характеризує її з різних точок зору. При цьому необхідно забезпечити можливості перевірки узгодженості моделей, побудованих виходячи з різних точок зору.

Коалгебраїчний підхід дозволяє вирішити цю проблему шляхом узгодження різних рівнів абстракції, які використовуються у специфікації системи. Всі рівні — від високорівневих архітектурних специфікацій до низькорівневих деталей реалізації — можуть бути описані в єдиній коалгебра-

їчній моделі. Це забезпечує цілісність та узгодженість всієї системи, спрощуючи її аналіз та підтримку.

Таким чином, саме коалгебраїчний підхід, який фактично інкапсулює в собі точку зору на систему через фіксацію відповідного ендофунктора, дає інструмент для формулювання узгодженності між моделями систем, побудованих виходячи з різних точок зору. Наявність фінальної системи для ендофунктора дає можливість досліджувати властивості будь-якої системи відповідного типу через властивості фінальної системи, що спрощує розробку формальних засобів верифікації системи в процесі її проєктування.

4.2 Метод синтезу імітаційних моделей складних динамічних систем з використанням універсальних коалгебр

Для вирішення проблем, описаних раніше пропонується практичний метод для розв'язання задачі синтезу імітаційних моделей складних динамічних систем (тобто складних програмних систем, яким властива динамічна поведінка) з метою їх подальшого використання для аналізу поведінки та специфікації складних динамічних систем.

Слід окремо зазначити, що наведений метод можна використовувати для широкого спектру складних систем, зокрема розподілених обчислювальних систем, асинхронних систем та кіберфізичних систем.

Запропонований метод синтезу імітаційних моделей складається з наступних етапів:

1. На першому етапі синтезу імітаційної моделі нам необхідно обрати точку зору на систему. Іншими словами необхідно формалізувати систему, для якої необхідно синтезувати модель за допомогою абстракції. Це можуть бути різні системи, зокрема

   (a) детермінована дискретна система

(б) рандомна дискретна система

(в) система з термінальними станами, тощо.

2. Далі необхідно підібрати представлення визначеної системи у термінах теорії універсальних коалгебр за допомогою відповідного ендофунктора.

3. Після визначення відповідного едофунктора для категорії систем необхідно перевірити факт існування фінальної системи за допомогою достатньої умови існування фінальної системи, яку було доведено у Розділі 2. Факт існування фінальної системи дасть можливість використовувати принципи коіндукції для безпосередньо визначення фінальної системи. Необхідно зробити зауваження, що без доведення факту існування фінальної системи використання принципів коіндукції не має сенсу.

4. З використанням методу коіндукції необхідно сформулювати необхідні властивості системи як фінальної системи.

5. На фінальному етапі синтезу імітаційної моделі необхідно побудувати найменшу підсистему фінальної системи, яка задовольняє цим властивостям фінальної системи. Саме ця найменша модель визначає необхідні для імітаційного моделювання властивості категорії систем.

Однак, слід зазначити, що наведений метод синтезу імітаційної моделі систем не вирішує проблему узгодженості імітаційних моделей. Така задача не планувалась до розв'язання у рамках дисертаційного дослідження та є основним напрямком для подальших досліджень на цю тему.

4.3 Практичне застосування теорії коалгебр при дослідження складних розподілених систем

В процесі виконання науково дослідної робіти за темою Integrated rail freight optimisation in Ukraine: Railway sleepers, rolling stock and logistics» (ДР № 0123U102700), у виконанні якої дисертант брав участь як виконавець. Метод, описаний вище було застосовано на практиці.

Крім того даний метод був застосований експериментально при передпроектному створенні системи моніторингу руху вагонопотоків для забезпечення прогнозного часу доставки вантажів на полігоні регіональної філії “Південна залізниця” АТ “Укрзалізниця” (Додаток Б, акт впровадження від 12 грудня 2023р)

Однією зі складових дослідження була побудова моделі на основі штучного інтелекту для прогнозування часу виконання операцій з вагонами різних видів вагонопотоків на залізничній мережі. Під час реалізації зазначеного етапу проекту виникла проблема нестачі даних для навчання моделі щодо працездатності систем геолокації та трекінгу вагонних відправок, а саме, системи моніторингу руху вагонопотоків.

Було прийнято рішення синтезувати імітаційну модель роботи системи моніторингу руху вагонопотоків з використанням методу, наведеного у попередньому розділі.

Під час аналізу вимог до імітаційної моделі було виявлено, що систему моніторингу руху вагонопотоків у рамках цілей основного дослідження можна абстрагувати без втрати адекватності моделі до системи з термінальними станами. Формально таку систему можна представити як систему, побудованому на розширенні натуральних чисел $N = \{0, 1, 2, ...\} \cup \{\infty\}$ з динамікою, побудованою наступним чином:

$$\ldots \longrightarrow n \longrightarrow n-1 \longrightarrow \ldots \longrightarrow 1 \longrightarrow 0 \longrightarrow \downarrow ,$$

іншими словами у нас існує стан, який стабілізує роботи системи (представлений за допомогою ∞) та набір станів (представлені за допомогою натуральних чисел), які призводять до аварійної зупинки системи ($\downarrow$). Таким чином було визначено точки зору на систему моніторингу руху вагонопотоків як на систему з термінальними станами.

Наступним кроком у синтезі імітаційної моделі є визначення системи з термінальними станами за допомогою відповідного ендофунктора. Таким чином було визначено ендофунктор

$$\begin{aligned} X &\mapsto 1 + X && \text{для будь-якох множини } X \\ X \xrightarrow{f} Y &\mapsto 1 + X \xrightarrow{id_1 + f} 1 + Y && \text{для будь-яких множин } X \text{ та } Y \end{aligned}$$

Доведення того факту, що $1+X$ дійсно є ендофунктором є тривіальним. Подальше дослідження цієї системи не несло сенсу, через той факт, що ця система була у деталях описана у дослідженнях Яна Руттена [55].

Зокрема, фінальною системою є $(N, pred)$ з $N = \mathbb{N} \cup \{\infty\}$ та $pred : N \to 1 + N$, що визначено у такий спосіб

$$pred(n) = \begin{cases} \downarrow & \text{якщо } n = 0 \\ n - 1 & \text{якщо } 0 < n \neq \infty \\ \infty & \text{якщо } n = \infty \end{cases}$$

Результатом синтезу імітаційних моделей для систему моніторингу руху вагонопотоків було створення за допомогою мови програмування python двох моделей, які задовільняли властивостям найменших підсистем фінальної системи для категорії систем з термінальними станами, а саме:

1. Стабільна система, яка не має властивостей виходити з ладу. Іншими словами, $pred(n)$ завжди набуває значення ∞

2. Система, що вийде з ладу через скінченну кількість ітерацій її роботи.

Рандомізація таких систем призводить до систем подібних до систем мнонади Джірі з такою зміною ендофунктора:

1. ендофунктор відправляє множину станів X тепер у множину скінченно визначених функцій $p : X \to [0,1]$ таких, що відповідають умові
$$\sum_{x \in X} p(x) \leq 1 \quad \text{замість умови} \quad \sum_{x \in X} p(x) = 1,$$
при цьому $1 - \sum\limits_{x \in X} p(x)$ інтерпоетується як ймовірність зупинки системи;

2. дія ендофунктора на функції співпадає з дією ендофунктора монади Джирі.

Фінальна системи такого ендофунктора також є множиною дерев, проте тепер дерево може мати листи, що відповідають станам перед зупинкою, тобто таким, для яких відповідна функція тотожньо дорівнює нулю.

Окремо слід зазначити, що практичний метод синтезу імітаційних моделей було застосовано на передпроектному етапі розробки турбодетандерної техніки для забезпечення розробка, виготовлення та впровадження детандер – компресорів, детандер – генераторів, включаючи комбіновані детандери, та інше газопромислове обладнання на виробництві ПАТ “Турбогаз” (Додаток Б, акт впровадження від 27 грудня 2023р.)

Висновки по Розділу 4

У розділі подано проблематику аналізу поведінки та специфікації складних програмних систем. Специфіка таких систем полягає у тому, що їх неможливо формалізувати з використанням однієї моделі. Тому, як правило, систему розглядають з різних точок зору і будують моделі у відповідності до них. Але такий підхід потребує уніфікованого інструментарію специфікації моделей різного типу. Саме запропонований коалгебраїчний підхід дає можливість створити такий інструментарій.

Було розглянуто метод синтезу моделі рандомної системи, як підсистеми фінальної системи відповідної категорії систем.

Було наведено результати практичного застосування запропонованого методу в процесі виконання науково дослідних робіт за темою Integrated rail freight optimisation in Ukraine: Railway sleepers, rolling stock and logistics» (ДР № 0123U102700), у виконанні якої дисертант брав участь як виконавець.

Основні положення цього розділу викладені у публікаціях автора [12].

ВИСНОВОК

У дисертаційній роботі поставлена та вирішена актуальна науково-технічна задача розробки теоретичних засад та практичних підходів до уніфікації методів специфікації поведінки дискретних динамічних систем з урахуванням їх різноманітності та високих вимог до безпеки (з точки зору завчасного виявлення критичних станів) та прогнозованості їх роботи.

В результаті проведених досліджень та розробок математичних моделей отримано наступні наукові та практичні результати:

1. Проведено аналіз сучасного стану методологій формальної специфікації програмних продуктів, в тому числі з використанням універсальних коалгебр, що дало змогу виділити ряд основних недоліків та переваг різних формальних підходів до проведення специфікації складних систем. У свою чергу це забезпечило обгрунтування коалгебраїчного підходу як одного з найбільш перспективних з огляду на мету дисертаційного дослідження.

2. Розроблено коалгебраїчні моделі для категорій дискретних динамічних систем, що породжують вихідні данні. Побудовані моделі було використано як основу для формального представлення рандомних систем з виходами.

3. Досліджено властивості ендофунктора дискретної динамічної системи з метою з'ясування факту наявності відповідної фінальної системи. На основі проведених досліджень було доведено достатню умову існування фінальної системи для відповідної категорії системи Цей результат було отримано вперше.

4. Сформульовано монаду Джирі з використання концепцій теорії універсальних коалгебр, що дало змогу створити визначити рандомні

системи саме як композицію детермінованої системи з виходами та монади Джірі.

5. Вперше використано монаду Джирі для забезпечення рандомізації дискретних динамічних систем з виходами з метою формального визначення рандомних систем. Таким чином, було створено модель рандомної дискретної системи з виходами з використанням універсальних коалгебр.

6. За допомогою доведеної раніше достатньої умови існування фінальної системи було встановлено факт існування фінальної системи для категорії рандомних дискретних систем з виходами та виведено відповідну систему. Показано, що вона має форму нескінченного дерева без листя, розміченого імовірностями переходів.

7. Удосконалено метод синтезу моделей для динамічного аналізу складних систем шляхом використання техніки універсальних коалгебр шляхом моделювання в фінальній системі.

Результати дисертаційного дослідження були також використані в навчальному процесі кафедри Теоретичної і прикладної інформатики Харківського національного університету імені В.Н.Каразіна та на факультеті Управління процесами перевезень Української державної академії залізничного транспорту (акт від 4 грудня 2023р.).

Також результати дисертаційного досліджена були практично впроваджені для моделювання роботи сортувальної станції з метою моніторингу руху вагонопотоків для забезпечення прогнозного часу доставки вантажів на полігоні регіональної філії “Південна залізниця” АТ “Укрзалізниця” (акт впровадження від 12 грудня 2023р.); та на виробництві ПАТ “Турбогаз”. Використання методик коалгебраїчної специфікації на етапі проектування турбодетандерної техніки дало можливість підвищити надійність техніки, що виготовляється (акт впровадження від 27 грудня 2023р.)

Додатково результати досліджень були використані у НДР «Integrated rail freight optimisation in Ukraine: Railway sleepers, rolling stock and logistics» (ДР No 0123U102700), у якому дисертант брав участь як виконавець.

Таким чином, сукупність отриманих у дисертації нових наукових результатів, позитивна оцінка їхньої достовірності, наукової та практичної значущості дають змогу вважати сформульовану наукову задачу удосконалення уніфікованих методів специфікації та аналізу дискретних динамічних систем різного типу включно з розподіленими за рахунок використання універсальних коалгебр – розв'язаною, а поставлену мету – досягнутою.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

[1] Tony Hoare та ін. “The Verified Software Initiative: A Manifesto”. В: *Theories of Programming: The Life and Works of Tony Hoare*. 1-е вид. New York, NY, USA: Association for Computing Machinery, 2021, с. 81—92. ISBN: 9781450387286. URL: https://doi.org/10.1145/3477355.3477361.

[2] Theo Theunissen, Uwe van Heesch та Paris Avgeriou. “A mapping study on documentation in Continuous Software Development”. В: *Information and Software Technology* 142 (2022), с. 106733. ISSN: 0950-5849. DOI: https://doi.org/10.1016/j.infsof.2021.106733. URL: https://www.sciencedirect.com/science/article/pii/S095058492100183X.

[3] Maurice E. Dawson та ін. “INTEGRATING SOFTWARE ASSURANCE INTO THE SOFTWARE DEVELOPMENT LIFE CYCLE (SDLC)”. В: 2010. URL: https://api.semanticscholar.org/CorpusID:61567871.

[4] Till Mossakowski та ін. “Algebraic-Coalgebraic Specification in CoCasl”. В: *Recent Trends in Algebraic Development Techniques*. За ред. Martin Wirsing, Dirk Pattinson та Rolf Hennicker. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, с. 376—392. ISBN: 978-3-540-40020-2.

[5] Noa Ragonis та Mordechai Ben-Ari. “A long-term investigation of the comprehension of OOP concepts by novices”. В: *Computer Science Education* 15.3 (2005), с. 203—221. DOI: 10.1080/08993400500224310. eprint: https://doi.org/10.1080/08993400500224310. URL: https://doi.org/10.1080/08993400500224310.

[6] Edmund M. Clarke. “Model checking”. В: *Foundations of Software Technology and Theoretical Computer Science*. За ред. S. Ramesh та G.

Sivakumar. Berlin, Heidelberg: Springer Berlin Heidelberg, 1997, с. 54—56. ISBN: 978-3-540-69659-9.

[7] Radhakisan Baheti та Helen Gill. “Cyber-physical systems”. В: *The impact of control technology* 12.1 (2011), с. 161—166.

[8] Janos Sztipanovits та ін. “Toward a Science of Cyber–Physical System Integration”. В: *Proceedings of the IEEE* 100.1 (2012), с. 29—44. DOI: 10.1109/JPROC.2011.2161529.

[9] Grygoriy Zholtkevych та Artem Panchenko. “About One Possible Tool for Analysing Safeness of Discrete Dynamic Systems”. В: *2023 13th International Conference on Dependable Systems, Services and Technologies (DESSERT)*. 2023, с. 1—7. DOI: 10.1109/DESSERT61349.2023.10416480.

[10] Artem Panchenko та Grygoriy Zholtkevych. “An Approach to Construct Final Random System with Output”. В: *Information and Communication Technologies in Education, Research, and Industrial Applications*. За ред. Vadim Ermolayev та ін. Cham: Springer International Publishing, 2022, с. 3—22. ISBN: 978-3-031-20834-8.

[11] Panchenko A Zholtkevych G. “Coalgebraic Understanding of Random Systems with Output”. В: *7th International Conference on ICT in Education, Research and Industrial Applications. Integration, Harmonization and Knowledge Kherson, Ukraine* Volume I: Main Conference, PhD Symposium, and Posters (November 2021).

[12] Artem Panchenko та ін. “Predicting the estimated time of cargo dispatch from a marshaling yard”. В: *Eastern-European Journal of Enterprise Technologies* 4.3 (106) (серп. 2020), с. 6—15. DOI: 10.15587/1729-4061.2020.209912. URL: https://journals.uran.ua/eejet/article/view/209912.

[13] Artem Panchenko та Hryhorii Zholtkevych. “The technique of modeling Cyberphysical systems using Coalgebra”. В: *Bulletin of V.N. Karazin*

Kharkiv National University, series «Mathematical modeling. Information technology. Automated control systems» 58 (черв. 2023), с. 47—53. DOI: 10.26565/2304-6201-2023-58-05. URL: https://periodicals.karazin.ua/mia/article/view/23500.

[14] K.E. Wiegers та J. Beatty. *Software Requirements*. Best practices. Microsoft Press, 2013. ISBN: 9780735679665. URL: https://books.google.com.ua/books?id=40lDmAEACAAJ.

[15] Alaa Abdalazeim та Farid Meziane. "A review of the generation of requirements specification in natural language using objects UML models and domain ontology". В: *Procedia Computer Science* 189 (2021). AI in Computational Linguistics, с. 328—334. ISSN: 1877-0509. DOI: https://doi.org/10.1016/j.procs.2021.05.102. URL: https://www.sciencedirect.com/science/article/pii/S1877050921012266.

[16] E. Hull, K. Jackson та J. Dick. *Requirements Engineering*. Practitioner series. Springer, 2005. ISBN: 9781852338794. URL: https://books.google.com.ua/books?id=e7ZhVD3JejAC.

[17] M. Frappier та H. Habrias. *Software Specification Methods: An Overview Using a Case Study*. Formal Approaches to Computing and Information Technology (FACIT). Springer London, 2012. ISBN: 9781447107019. URL: https://books.google.com.ua/books?id=amEECAAAQBAJ.

[18] Frauke Paetsch, Armin Eberlein та Frank Maurer. "Requirements engineering and agile software development". В: *WET ICE 2003. Proceedings. Twelfth IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, 2003.* (2003), с. 308—313. URL: https://api.semanticscholar.org/CorpusID:13610125.

[19] K. Pohl та C. Rupp. *Requirements Engineering Fundamentals: A Study Guide for the Certified Professional for Requirements Engineering Exam - Foundation Level - IREB Compliant*. Rocky nook computing. Rocky

Nook, 2015. ISBN: 9781937538774. URL: https://books.google.com.ua/books?id=bM1YrgEACAAJ.

[20] Wenliang Mao та ін. "Energy-Efficient Industrial Internet of Things: Overview and Open Issues". В: *IEEE Transactions on Industrial Informatics* 17.11 (2021), с. 7225—7237. DOI: 10.1109/TII.2021.3067026.

[21] Md Tarique Jamal Ansari, Dhirendra Pandey та Mamdouh Alenezi. "STORE: Security Threat Oriented Requirements Engineering Methodology". В: *Journal of King Saud University - Computer and Information Sciences* 34.2 (2022), с. 191—203. ISSN: 1319-1578. DOI: https://doi.org/10.1016/j.jksuci.2018.12.005. URL: https://www.sciencedirect.com/science/article/pii/S1319157818306876.

[22] Brian Fitzgerald та Klaas-Jan Stol. "Continuous software engineering: A roadmap and agenda". В: *J. Syst. Softw.* 123 (2017), с. 176—189. URL: https://api.semanticscholar.org/CorpusID:206538464.

[23] M. Zelkowitz. *Advances in Computers: Advances in Software Engineering*. ISSN. Elsevier Science, 2004. ISBN: 9780080471907. URL: https://books.google.com.ua/books?id=N-06uoJ9iSsC.

[24] Ramtin Jabbari та ін. "What is DevOps? A Systematic Mapping Study on Definitions and Practices". В: *Proceedings of the Scientific Workshop Proceedings of XP2016*. XP '16 Workshops. Edinburgh, Scotland, UK: Association for Computing Machinery, 2016. ISBN: 9781450341349. DOI: 10.1145/2962695.2962707. URL: https://doi.org/10.1145/2962695.2962707.

[25] M.A. Boden. *AI: Its Nature and Future*. Oxford University Press, 2016. ISBN: 9780198777984. URL: https://books.google.com.ua/books?id=yDQTDAAAQBAJ.

[26] Alistair Cockburn. *Writing Effective Use Cases*. 1st. USA: Addison-Wesley Longman Publishing Co., Inc., 2000. ISBN: 0201702258.

[27] Garm Lucassen та ін. “The Use and Effectiveness of User Stories in Practice”. В: *Requirements Engineering: Foundation for Software Quality*. За ред. Maya Daneva та Oscar Pastor. Cham: Springer International Publishing, 2016, с. 205—222. ISBN: 978-3-319-30282-9.

[28] G. Schneider та J.P. Winters. *Applying Use Cases: A Practical Guide*. Addison-Wesley Object Technology Series. Pearson Education, 2001. ISBN: 9780789745453. URL: https://books.google.com.ua/books?id=mOCZ1xjgQyMC.

[29] Henry Edison, Xiaofeng Wang та Kieran Conboy. “Comparing Methods for Large-Scale Agile Software Development: A Systematic Literature Review”. В: *IEEE Transactions on Software Engineering* 48.8 (2022), с. 2709—2731. DOI: 10.1109/TSE.2021.3069039.

[30] D. Pilone та N. Pitman. *UML 2.0 in a Nutshell*. In a Nutshell (o’Reilly) Series. O’Reilly Media, 2005. ISBN: 9780596007959. URL: https://books.google.com.ua/books?id=Xg8wLPmt5CMC.

[31] Francisco Javier Lucas Martínez та Ambrosio Toval Álvarez. “A Precise Approach for the Analysis of the UML Models Consistency”. В: *Perspectives in Conceptual Modeling*. За ред. Jacky Akoka та ін. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, с. 74—84. ISBN: 978-3-540-32239-9.

[32] Yonglei Tao та Chenho Kung. “Formal definition and verification of data flow diagrams”. В: *Journal of Systems and Software* 16.1 (1991), с. 29—36. ISSN: 0164-1212. DOI: https://doi.org/10.1016/0164-1212(91)90029-6. URL: https://www.sciencedirect.com/science/article/pii/0164121291900296.

[33] Rahul Karmakar. “Formal Verification Techniques: A Comparative Analysis for Critical System Design”. В: *Intelligent Systems Design and Applications*. За ред. Ajith Abraham та ін. Cham: Springer International Publishing, 2022, с. 93—102. ISBN: 978-3-030-96308-8.

[34] John S. Fitzgerald, Peter Gorm Larsen та Marcel Verhoef. “Vienna Development Method”. В: *Wiley Encyclopedia of Computer Science and Engineering*. John Wiley Sons, Ltd, 2008, с. 1—11. ISBN: 9780470050118. DOI: https://doi.org/10.1002/9780470050118.ecse447. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/9780470050118.ecse447. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/9780470050118.ecse447.

[35] I. Van Horebeek та J. Lewi. *Algebraic Specifications in Software Engineering: An Introduction*. Springer Berlin Heidelberg, 2012. ISBN: 9783642750304. URL: https://books.google.com.ua/books?id=mLerCAAAQBAJ.

[36] Brian Dobing та Jeffrey Parsons. “How UML is used”. В: *Commun. ACM* 49.5 (трав. 2006), с. 109—113. ISSN: 0001-0782. DOI: 10.1145/1125944.1125949. URL: https://doi.org/10.1145/1125944.1125949.

[37] Hubert Garavel, Maurice H. ter Beek та Jaco van de Pol. “The 2020 Expert Survey on Formal Methods”. В: *Formal Methods for Industrial Critical Systems*. За ред. Maurice H. ter Beek та Dejan Ničković. Cham: Springer International Publishing, 2020, с. 3—69. ISBN: 978-3-030-58298-2.

[38] Matt Luckcuck та ін. “Formal Specification and Verification of Autonomous Robotic Systems: A Survey”. В: *ACM Comput. Surv.* 52.5 (вер. 2019). ISSN: 0360-0300. DOI: 10.1145/3342355. URL: https://doi.org/10.1145/3342355.

[39] Yannick Moy та ін. “Testing or Formal Verification: DO-178C Alternatives and Industrial Experience”. В: *IEEE Software* 30.3 (2013), с. 50—57. DOI: 10.1109/MS.2013.43.

[40] Brian R. Larson, Patrice Chalin та John Hatcliff. “BLESS: Formal Specification and Verification of Behaviors for Embedded Systems with Software”. В: *NASA Formal Methods*. За ред. Guillaume Brat, Neha

Rungta та Arnaud Venet. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, с. 276—290. ISBN: 978-3-642-38088-4.

[41] Ben Potter, Jane Sinclair та David Till. *An introduction to formal specification and Z*. Prentice-Hall, Inc., 1992.

[42] Fulvio Babich та Lia Deotto. "Formal methods for specification and analysis of communication protocols". В: *IEEE Communications Surveys Tutorials* 4.1 (2002), с. 2—20. DOI: 10.1109/COMST.2002.5341329.

[43] Doron A. Peled. "Formal Methods". В: *Handbook of Software Engineering*. За ред. Sungdeok Cha, Richard N. Taylor та Kyochul Kang. Cham: Springer International Publishing, 2019, с. 193—222. DOI: 10.1007/978-3-030-00262-65. URL: https://doi.org/10.1007/978-3-030-00262-65.

[44] Bernhard Beckert та ін. "Intelligent Systems and Formal Methods in Software Engineering". В: *IEEE Intelligent Systems* 21.6 (2006), с. 71—81. DOI: 10.1109/MIS.2006.117.

[45] H.H. Hansen та F. Zanasi. *Coalgebraic Methods in Computer Science: 16th IFIP WG 1.3 International Workshop, CMCS 2022, Colocated with ETAPS 2022, Munich, Germany, April 2-3, 2022, Proceedings*. Lecture Notes in Computer Science. Springer International Publishing, 2022. ISBN: 9783031107368. URL: https://books.google.com.ua/books?id=RjV9EAAAQBAJ.

[46] Corina Cırstea. "A coalgebraic equational approach to specifying observational structures". В: *Theoretical Computer Science* 280.1 (2002). Coalgebraic Methods in Computer Science, с. 35—68. ISSN: 0304-3975. DOI: https://doi.org/10.1016/S0304-3975(01)00020-2. URL: https://www.sciencedirect.com/science/article/pii/S0304397501000202.

[47] Andrea Corradini. "A completeness result for equational deduction in coalgebraic specification". В: *Recent Trends in Algebraic Development*

Techniques. За ред. Francesco Parisi Presicce. Berlin, Heidelberg: Springer Berlin Heidelberg, 1998, с. 190—205. ISBN: 978-3-540-69719-0.

[48] Bart Jacobs. "Exercises in Coalgebraic Specification". В: *Algebraic and Coalgebraic Methods in the Mathematics of Program Construction: International Summer School and Workshop Oxford, UK, April 10–14, 2000 Revised Lectures*. За ред. Roland Backhouse, Roy Crole та Jeremy Gibbons. Berlin, Heidelberg: Springer Berlin Heidelberg, 2002, с. 237—281. ISBN: 978-3-540-47797-6. DOI: 10.1007/3-540-47797-7_7. URL: https://doi.org/10.1007/3-540-47797-7_7.

[49] M.S. Mahmoud та M.G. Singh. *Discrete Systems: Analysis, Control and Optimization*. Communications and Control Engineering. Springer Berlin Heidelberg, 2012. ISBN: 9783642823275. URL: https://books.google.com.ua/books?id=BuTnCAAAQBAJ.

[50] O. Galor. *Discrete Dynamical Systems*. Springer Berlin Heidelberg, 2007. ISBN: 9783540367765. URL: https://books.google.com.ua/books?id=HbqYFpVDCBUC.

[51] K. Denecke та S.L. Wismath. *Universal Algebra and Coalgebra*. G - Reference,Information and Interdisciplinary Subjects Series. World Scientific, 2009. ISBN: 9789812837455. URL: https://books.google.com.ua/books?id=NgTAzhC8jVAC.

[52] J. Ding та A. Zhou. *Statistical Properties of Deterministic Systems*. Tsinghua University Texts. Springer Berlin Heidelberg, 2010. ISBN: 9783540853671. URL: https://books.google.com.ua/books?id=gcVnbYgMU-wC.

[53] B. Mitchell. *Theory of Categories*. ISSN. Elsevier Science, 1965. ISBN: 9780080873299. URL: https://books.google.com.ua/books?id=hgJ3pTQSAdOC.

[54] Corina Cırstea. "A coalgebraic equational approach to specifying observational structures". В: *Theoretical Computer Science* 280.1 (2002).

Coalgebraic Methods in Computer Science, с. 35—68. ISSN: 0304-3975. DOI: https://doi.org/10.1016/S0304-3975(01)00020-2. URL: https://www.sciencedirect.com/science/article/pii/S0304397501000202.

[55] J.J.M.M. Rutten. "Universal coalgebra: a theory of systems". В: *Theoretical Computer Science* (2000).

[56] JIŘÍ ADÁMEK, STEFAN MILIUS та JIŘÍ VELEBIL. "A general final coalgebra theorem". В: *Mathematical Structures in Computer Science* 15.3 (2005), с. 409—432. DOI: 10.1017/S0960129505004731.

[57] Joachim Lambek. "A Fixpoint Theorem for complete Categories." В: *Mathematische Zeitschrift* 103 (1968), с. 151—161. URL: http://eudml.org/doc/170906.

[58] D. Sangiorgi. *Introduction to Bisimulation and Coinduction*. Cambridge University Press, 2011. ISBN: 9781139502832. URL: https://books.google.com.ua/books?id=ysDHav-iiMwC.

[59] E.P. de Vink та J.J.M.M. Rutten. "Bisimulation for probabilistic transition systems: a coalgebraic approach". В: *Theoretical Computer Science* 221.1 (1999), с. 271—293. ISSN: 0304-3975. DOI: https://doi.org/10.1016/S0304-3975(99)00035-3. URL: https://www.sciencedirect.com/science/article/pii/S0304397599000353.

[60] Davide Sangiorgi. "On the origins of bisimulation and coinduction". В: 31.4 (трав. 2009). ISSN: 0164-0925. DOI: 10.1145/1516507.1516510. URL: https://doi.org/10.1145/1516507.1516510.

[61] Alexander Kurz та Rolf Hennicker. "On institutions for modular coalgebraic specifications". В: *Theoretical Computer Science* 280.1 (2002). Coalgebraic Methods in Computer Science, с. 69—103. ISSN: 0304-3975. DOI: https://doi.org/10.1016/S0304-3975(01)00021-4. URL: https://www.sciencedirect.com/science/article/pii/S0304397501000214.

[62] Claudio Hermida та Bart Jacobs. “Structural Induction and Coinduction in a Fibrational Setting”. В: *Information and Computation* 145.2 (1998), с. 107—152. ISSN: 0890-5401. DOI: https://doi.org/10.1006/inco.1998.2725. URL: https://www.sciencedirect.com/science/article/pii/S0890540198927250.

[63] Jesse Hughes та Bart Jacobs. “Simulations in coalgebra”. В: *Theoretical Computer Science* 327.1 (2004). Selected Papers of CMCS ’03, с. 71—108. ISSN: 0304-3975. DOI: https://doi.org/10.1016/j.tcs.2004.07.022. URL: https://www.sciencedirect.com/science/article/pii/S030439750400444X.

[64] Neil Ghani та ін. “Algebras, Coalgebras, Monads and Comonads”. В: *Electronic Notes in Theoretical Computer Science* 44.1 (2001). CMCS 2001, Coalgebraic Methods in Computer Science (a Satellite Event of ETAPS 2001), с. 128—145. ISSN: 1571-0661. DOI: https://doi.org/10.1016/S1571-0661(04)80905-8. URL: https://www.sciencedirect.com/science/article/pii/S1571066104809058.

[65] Gert K. Pedersen. “Pullback and Pushout Constructions in C*-Algebra Theory”. В: *Journal of Functional Analysis* 167.2 (1999), с. 243—344. ISSN: 0022-1236. DOI: https://doi.org/10.1006/jfan.1999.3456. URL: https://www.sciencedirect.com/science/article/pii/S0022123699934560.

[66] Daniele Turi та Jan J. M. M. Rutten. “On the foundations of final coalgebra semantics: non-well-founded sets, partial orders, metric spaces”. В: *Mathematical Structures in Computer Science* 8 (1998), с. 481—540.

[67] Artem Panchenko та Grygoriy Zholtkevych. “Coalgebraic Understanding of Random Systems with Output”. В: *ICTERI 2021: ICT in Education, Research, and Industrial Applications*. За ред. V. Ermolayev та ін. Т. 3013. CEUR Workshop Proceedings. 2021, с. 296—306.

[68] Donald Ervin Knuth. “Two notes on notation”. В: *American Mathematical Monthly* 99 (1992), с. 403—422.

[69] David J. Smith. “Chapter 8 - Methods of Modeling”. В: *Reliability, Maintainability and Risk (Ninth Edition)*. За ред. David J. Smith. Ninth Edition. Butterworth-Heinemann, 2017.

[70] Alfred Tarski. “A lattice-theoretical fixpoint theorem and its applications”. В: *Pacific J. Math.* 2.5 (1955), с. 285—309.

[71] Alexander F. Siegenfeld та Yaneer Bar-Yam. “An Introduction to Complex Systems Science and Its Applications”. В: *Complexity* 2020.1 (2020), с. 6105872. DOI: https://doi.org/10.1155/2020/6105872. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1155/2020/6105872. URL: https://onlinelibrary.wiley.com/doi/abs/10.1155/2020/6105872.

[72] Jiaxin Wu та Pingfeng Wang. “Risk-averse optimization for resilience enhancement of complex engineering systems under uncertainties”. В: *Reliability Engineering System Safety* 215 (2021), с. 107836. ISSN: 0951-8320. DOI: https://doi.org/10.1016/j.ress.2021.107836. URL: https://www.sciencedirect.com/science/article/pii/S0951832021003562.

Додаток А.

СПИСОК ПУБЛІКАЦІЙ ЗДОБУВАЧА ЗА ТЕМОЮ ДИСЕРТАЦІЇ

Статті у наукових фахових виданнях, що входять до міжнародних наукометричних баз

1. Grygoriy Zholtkevych, Artem Panchenko. About One Possible Tool for Analysing Safeness of Discrete Dynamic Systems. 2023. 13th International Conference on Dependable Systems, Services and Technologies (DESSERT). pp. 1-7.
DOI:http://dx.doi.org/10.1109/DESSERT61349.2023.10416480
URL: https://ieeexplore.ieee.org/document/10416480
Ключові слова: discrete system, actual system state, forecasted system state, dynamical transformation, endofunctor, universal coalgebra, coalgebraic morphism, colimit, eventual coalgebra
(Особистий внесок: виведення частини моделей на основі універсальних коалгебр для розповсюджених динамічних систем а також написання частини тексту та його переклад
Особистий внесок Grygoriy Zholtkevych: формалізація фінальної коалгебри для дискретної детермінованої системи а також написання частини тексту та його переклад)

2. Grygoriy Zholtkevych, Artem Panchenko. An Approach to Construct Final Random System with Output. Communications in Computer and Information Science. 2022., Vol. 1698. Pp. 3 – 22.
DOI: https://doi.org/10.1007/978-3-031-20834-8_1
URL: https://link.springer.com/chapter/10.1007/978-3-031-20834-8_1
Ключові слова: coalgebra, coalgebraic morphism, final system, anamorphism
(Особистий внесок: розробка теоритичних засад існування фінальної коалгебри для обраного типу динамічних систем, написання частини тексту та переклад а також виступ на конференції

Особистий внесок Grygoriy Zholtkevych: формалізація фінальної коалгебри для рандомних дискретних систем з вхідними даними а також написання частини тексту та переклад)

3. Grygoriy Zholtkevych, Artem Panchenko. Coalgebraic Understanding of Random Systems with Output. CEUR Workshop Proceedings. 2021. Vol 3031. Pp. 296 – 306.

Ключові слова: discrete system, system with output, random system, coalgebraic approach

URL: https://ceur-ws.org/Vol-3013/20210296.pdf

(Особистий внесок здобувача розробка підходів до впровадження стохастичності до детермінованої моделі за допомогою монади вірогіднісного розподілу Джирі, написання частини тексту та переклад а також виступ на конференції

Особистий внесок Zholtkevych: формалізація моделі рандомної дискретної системи з виходами з використанням універсальних коалгебр а також написання частини тексту та переклад)

4. Panchenko A., Prokhorchenko A., Panchenko S., Dekarchuk O., Gurin D., Medvediev I. Predicting the estimated time of cargo dispatch from a marshaling yard. Eastern-European Journal of Enterprise Technologies.2020. Vol. 4. Is. 3 (106). Pp. 6–1

DOI https://doi.org/10.15587/1729-4061.2020.209912

URL: https://journals.uran.ua/eejet/article/view/209912

Ключові слова: railroad, marshaling yard, cargo dispatch, expected departure time, machine learning

(Особистий внесок здобувача: розробка імітаційної моделі залізниці з оглядом на її природу, яку можна охарактеризувати як розподілену, написання частини тексту та переклад

Особистий внесок Prokhorchenko A: аналіз поточного стану проблеми, що досліджується, постановка задачі дослідження а також написання частини тексту та переклад

Особистий внесок Panchenko S: створення моделі на основі машинного

навчання для прогнозування часу доставки вантажу а написання частини тексту та переклад

Особистий внесок Dekarchuk O: дослідження вантажообігу на залізниці з метою покращення роботи імітаційної моделі а також написання частини тексту та переклад

Особистий внесок Gurin D: генерування тестових даних для навчання моделей машинного навчання а також написання частини тексту та переклад

Особистий внесок Medvediev I: крос валідація роботи моделі машинного навчання а також написання тексту та переклад)

Статті у наукових фахових виданнях України

5. Grygoriy Zholtkevych, Artem Panchenko. The technique of modeling Cyberphysical systems using Coalgebra. Bulletin of V.N. Karazin Kharkiv National University,. Series «Mathematical Modeling. Information Technology. Automated Control Systems»,. 2023. Vol. 58. Pp. 47-53.
DOI: https://doi.org/10.26565/2304-6201-2023-58-05
URL: https://periodicals.karazin.ua/mia/article/view/23500
Ключові слова: Cyber-Physical Systems, Coalgebra, Dynamic Systems Modeling, Category Theory, Final Coalgebra *(Особистий внесок здобувача: вивчення теоритичних засад використання декартових квадратів для доведення існування фінальної коалгебри для побудованих моделей динамічних систем, написання частини тексту та переклад*

Особистий внесок Grygoriy Zholtkevych:моделювання динамічних систем за допомогою універсальних коалгебр та визначення фінальних коалгебр для дискретних динамічних систем)

Додаток Б.

Акти про практичне застосування отриманих результатів.

ЗАТВЕРДЖЕНО
Голова Правління Приватного
Акціонерного Товариства
«ТУРБОГАЗ»

Олена ПАСТУХОВА

«27» грудня 2023 року

АКТ
про впровадження результатів дисертаційної роботи ПАНЧЕНКА Артема Сергійовича на здобуття ступеня доктора філософії за спеціальністю 122 – Комп'ютерні науки (Галузь знань 12: Інформаційні науки) за темою «КОАЛГЕБРАЇЧНІ ЗАСОБИ СПЕЦИФІКАЦІЇ ТА АНАЛІЗУ СТАТИСТИЧНИХ ОБМЕЖЕНЬ ПОВЕДІНКИ РОЗПОДІЛЕНИХ СИСТЕМ»

До основних результатів дисертаційної роботи Артема ПАНЧЕНКА, що експериментально були впроваджені на передпроєктному етапі розробки турбодетандерної техніки для забезпечення розробки, виготовлення та впровадження турбодетандерів (компресорів, генераторів, включаючи комбіновані), та інше газопромислове обладнання на виробництві Приватного Акціонерного Товариства «ТУРБОГАЗ» належать:

- метод побудови математичної моделі роботи газопромислового обладнання з використанням універсальних коалгебр, враховуючи достатню умову збереження критеріїв подібності;

- інжиніринг і розробка конструкторської документації з використанням методу динамічного аналізу (моделювання) складних систем з використанням універсальних коалгебр.

Заступник Голови Правління,
головний інженер
ПрАТ «ТУРБОГАЗ» **Максим НОВІКОВ**

ЗАТВЕДЖЕНО
Перший заступник директора
виконавчого регіональної філії
«Південна залізниця» АТ «Укрзалізниця»
Михайло КУЗЬМЕНКО
12 грудня 2023р

АКТ

щодо впровадження результатів дисертаційної роботи на здобуття ступеня доктора філософії за спеціальністю 122 - Комп'ютерні науки (Галузь знань 12: Інформаційні науки) Панченка Артема Сергійовича

До основних результатів дисертаційної роботи Артема Панченка , що експериментально були впроваджені при передпроектному створенні системи моніторингу руху вагонопотоків для забезпечення прогнозного часу доставки вантажів на полігоні регіональної філії "Південна залізниця" АТ "Укрзалізниця", належать

- метод побудови імітаційної моделі роботи сортувальної станції з використанням універсальних коалгебр та враховуючи достатню умову збереження ендофунктором категорії множин слабких декартових квадраті, що дозволяє встановити факт існування фінальної коалгебри, що забезпечило більш ефективне прогнозування часу виконання операцій з вагонами різних видів вагонопотоків;
- метод прогнозування ETD (очікуваного часу відправлення) вантажної відправки на сортувальній станції з використанням методу динамічного аналізу (імітаційного моделювання) складних систем з використанням універсальних коалгебр та методології прогнозування, що має в основі деревоподібну структуру.
- метод проектування системи моніторингу руху вагонопотоків на основі паралельних обчислень прогнозних моделей ETD сортувальних станцій на розгалужених полігонах залізничної мережі, з використанням техніки обчислення фінальних систем.

Впровадження зазначених результатів надало можливість дослідити систему руху вагонопотоків на полігоні регіональної філії "Південна залізниця" АТ "Укрзалізниця".

Отримані результати дозволяють забезпечити точність прогнозування ETD сортувальних станцій на розгалужених полігонах залізничної мережі на рівні, близькому до 90% з похибкою 4 - 9%. Методи імітаційного моделювання з використанням універсальних коалгебр дають змогу зменшити потенційну кількість помилок при розробці систем моніторингу руху вагонопотоків до 15%. Це може дозволити зменшити показник середнього часу простою вагона - транзит з переробкою до 10% від існуючих показників.

Начальник регіонального
Центру управління рухом регіональної філії
«Південназалізниця»АТ«Укрзалізниця» Дмитро ШВАРЬОВ

Заступник начальника служби
роботи станцій регіональної філії
«Південна залізниця» АТ «Укрзалізниця» Дмитро ТРОЙНИКОВ

ЗАТВЕРДЖЕНО

Проректор з науково-педагогічної роботи

Українського державного університету

залізничного транспорту

_____ Артур КАГРАМАНЯН

"14" грудня 2023 р.

АКТ

про впровадження результатів дисертаційної роботи
Панченка Артема Сергійовича на здобуття ступеня доктора філософії за спеціальністю 122 - Комп'ютерні науки (Галузь знань 12: Інформаційні науки)
за темою "КОАЛГЕБРАЇЧНІ ЗАСОБИ СПЕЦИФІКАЦІЇ ТА АНАЛІЗУ СТАТИСТИЧНИХ ОБМЕЖЕНЬ ПОВЕДІНКИ РОЗПОДІЛЕНИХ СИСТЕМ" у навчальному процесі
Українського державного університету залізничного транспорту (УкрДУЗТ)

Основні результати дисертаційної роботи Артема Панченка впроваджено у навчальний процес УкрДУЗТ на факультеті "Управління процесами перевезень" кафедри "Управління експлуатаційною роботою" у дисциплінах "Інформаційні технології в управлінні міжнародними перевезеннями", "Сучасні інформаційні технології в управлінні залізничними підрозділами" та "Управління експлуатаційною роботою" при вивченні систем та підходів до проєктування систем моніторингу руху вагонопотоків та математичних імітаційних моделей на залізниці, зокрема:

- при вивченні принципів планування роботи станції використовувався метод побудови імітаційної моделі роботи сортувальної станції з використанням універсальних коалгебр;
- при вивченні систем моніторингу руху вагонопотоків використовувався метод прогнозування очікуваного часу відправлення (ETD) вантажної відправки на сортувальній станції з використанням методу динамічного аналізу (імітаційного моделювання) складних систем з використанням універсальних коалгебр та методології прогнозування, що має в основі деревоподібну структуру.
- при вивченні методів проектування систем моніторингу руху вагонопотоків використовувалися техніки обчислення фінальних систем.

Дані розробки використовуються при підготовці до дипломування бакалаврів і

факультеті “Управління процесами перевезень” за освітніми програмами “Організація перевезень і управління на транспорті”, “Організація міжнародних перевезень”.

Декан факультету
“Управління процесами перевезень”
УкрДУЗТ, к.т.н., доцент

Максим КУЦЕНКО

Онлайн сервіс створення та перевірки кваліфікованого та удосконаленого електронного підпису

ПРОТОКОЛ
створення та перевірки кваліфікованого та удосконаленого електронного підпису

Дата та час: 19:01:04 19.06.2024

Назва файлу з підписом: PhD Panchenko final.pdf.asice
Розмір файлу з підписом: 2.1 МБ

Перевірені файли:
Назва файлу без підпису: PhD Panchenko final.pdf
Розмір файлу без підпису: 2.2 МБ

Результат перевірки підпису: Підпис створено та перевірено успішно. Цілісність даних підтверджено

Підписувач: ПАНЧЕНКО АРТЕМ СЕРГІЙОВИЧ
П.І.Б.: ПАНЧЕНКО АРТЕМ СЕРГІЙОВИЧ
Країна: Україна
РНОКПП: 3541507573
Організація (установа): ФОП ПАНЧЕНКО АРТЕМ СЕРГІЙОВИЧ
Код ЄДРПОУ: 3541507573
Посада: КЕРІВНИК
Час підпису (підтверджено кваліфікованою позначкою часу для підпису від Надавача): 19:01:03 19.06.2024
Сертифікат виданий: КНЕДП АЦСК АТ КБ "ПРИВАТБАНК"
Серійний номер: 5E984D526F82F38F0400000039A637015E50A504
Алгоритм підпису: ДСТУ 4145
Тип підпису: Удосконалений
Тип контейнера: Підпис та дані в архіві (розширений) (ASiC-E)
Формат підпису: З повними даними для перевірки (XAdES-B-LT)
Сертифікат: Кваліфікований

Версія від: 2024.04.15 13:00