МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

МАТЕМАТИЧНИЙ ФАКУЛЬТЕТ
Кафедра прикладної математики і механіки

КВАЛІФІКАЦІЙНА РОБОТА МАГІСТРА

на тему: **«АЛГОРИТМІЗАЦІЯ ПРОЦЕСУ МАТЕМАТИЧНОГО МОДЕЛЮВАННЯ СКЛАДНИХ СИСТЕМ: РОЗВ'ЯЗАННЯ ПРОБЛЕМИ ОБЧИСЛЮВАЛЬНОЇ СКЛАДНОСТІ»**

Виконав: студент 2 курсу, групи 8.1138
спеціальності 113 прикладна математика
(шифр і назва спеціальності)
освітньої програми прикладна математика
(назва освітньої програми)
М.С. Нагорний
(ініціали та прізвище)

Керівник доцент кафедри прикладної математики і механіки, доцент, к.ф.-м.н. Кондрат'єва Н.О.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Рецензент декан математичного факультету, професор, д.т.н. Гоменюк С.І.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Запоріжжя – 2020

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет математичний

Кафедра прикладної математики і механіки

Рівень вищої освіти магістр

Спеціальність 113 прикладна математика
(шифр і назва)

Освітня програма прикладна математика

ЗАТВЕРДЖУЮ

Завідувач кафедри прикладної математики і механіки, д.т.н. професор

_____________ Грищак В.З.
(підпис)

« 29 » 05 2019 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТОВІ

Нагорному Максиму Сергійовичу
(прізвище, ім'я та по-батькові)

1. Тема роботи (проекту) Алгоритмізація процесу математичного моделювання складних систем: розв'язання проблеми обчислювальної складності

керівник роботи (проекту) Кондрат'єва Наталія Олександрівна, к.ф.-м.н., доцент
(прізвище, ім'я та по-батькові, науковий ступінь, вчене звання)

затверджені наказом ЗНУ від « 29 » 05 2019 р. № 811-с

2. Строк подання студентом роботи 27.12.19

3. Вихідні дані до роботи 1. Постановка задачі.
2. Перелік літератури.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
   1. Постановка задачі.
   2. Основні теоретичні відомості.
   3. Розв'язання проблеми обчислювальної складності складних систем.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)
презентація

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		Завдання видав	Завдання прийняв

7. Дата видачі завдання 29.05.19

К А Л Е Н Д А Р Н И Й   П Л А Н

№	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1.	Розробка плану роботи.	02.09.19-08.09.19	
2.	Збір вихідних даних.	09.09.19-22.09.19	
3.	Обробка методичних та теоретичних	23.09.19-13.10.19	
	джерел.		
4.	Розробка першого розділу.	14.10.19-3.11.19	
5.	Розробка другого розділу.	4.11.19-17.11.19	
6.	Розробка третього розділу.	18.11.19-1.12.19	
7.	Практична частина	2.12.19-15.12.19	
8.	Оформлення та нормоконтроль	16.12.19-27.12.19	
	кваліфікаційної роботи.		
9.	Захист кваліфікаційної роботи.	16.01.20	

Студент ________________ М.С. Нагорний
(підпис) (ініціали та прізвище)

Керівник роботи ________________ Н.О. Кондрат’єва
(підпис) (ініціали та прізвище)

Нормоконтроль пройдено

Нормоконтролер ________________ В.В. Леонтьєва
(підпис) (ініціали та прізвище)

РЕФЕРАТ

Кваліфікаційна робота магістра «Алгоритмізація процесу математичного моделювання складних систем: розв’язання проблеми обчислювальної складності»: 113 с., 19 рис., 12 табл., 19 джерел, 6 додатків.

ВИРІШУЮЧІ ФОРМИ, ЕМПІРИЧНІ ДАНІ, ЗМІННІ, ЗМІСТОВНІ ПІДМАСКИ, МАТЕМАТИЧНА МОДЕЛЬ, МІРИ НЕЧІТКОСТІ, НАПРАВЛЕНА СИСТЕМА, НЕДЕТЕРМІНОВАНІСТЬ, НЕЙТРАЛЬНА СИСТЕМА, ПАРАМЕТРИ, ПОВЕДІНКА, ПОРОДЖЕНІ ДАНІ, РЕШІТКА, СКЛАДНІСТЬ, СПРОЩЕННЯ.

Об’єкт дослідження – математична модель складної системи.

Мета роботи – алгоритмізація процесу математичного моделювання складних систем та розгляд проблеми обчислювальної складності.

Метод дослідження – аналітичний.

Для реалізації даної цілі дослідження розглядаються наступні питання: класифікація етапів проведення емпіричного дослідження, розгляд методології дослідження складних систем за допомогою визначення початкової системи на об’єкті та її параметрів, побудови початкової системи і приведення її до систем з поведінкою нейтрального та направленого типу. Після цього, для отриманих систем розглядається питання визначення оптимальних систем з поведінкою за допомогою змістовних підмасок та виділення ентропій для кожної з них. Останнім етапом є спрощення визначених систем даних за допомогою першого і другого родів спрощення складності. Практична частина присвячена розгляду алгоритмізації всіх висвітлених питань у теоретичній частині. Для її реалізації розробляються наступні програмні продукти: «Elementary system with mask», «Directed generating system with behavior», «Content masks for elementary system with mask», «Entropies for content masks», «Entropies of content masks and 1 simplication», «Entropies for content masks and 2 simplication».

SUMMARY

Master qualifying paper «Algorithmization of the process of mathematical modeling of complex systems: solving the problem of computational complexity»: 113 pages, 19 figures, 12 tables, 19 references, 6 supplements.

RESOLVING FORMS, EMPIRICAL DATA, VARIABLES, INTENSIONAL SUB-MASKS, MATHEMATICAL MODEL, CARELESSNESS' MEASURES, DIRECTED SYSTEM, NON-DETERMINATION, NEUTRAL SYSTEM, PARAMETERS, BEHAVIOR, GENERATED DATA, GRID, COMPLEXITY, SIMPLIFICATION.

The object of the study is mathematical model of the entangled system.

The aim of the study is to algorithmize the process of mathematical modeling of complex systems and to consider the problem of computational complexity.

The method of research is analytical.

In order to realize this research goal, the following questions are considered: classification of stages of conducting empirical research, reviewing the methodology of researching complex systems by determining the initial system on the object and its parameters, building the initial system and bringing it to systems with behavior of neutral and directional type. After that, the obtained systems are considered to determine the optimal behavior systems with the help of meaningful sub-masks and the entropy allocation for each of them. The last step is to simplify certain data systems by simplifying the first and second types of complexity. The practical part is devoted to the consideration of the algorithmization of all illuminated issues in the theoretical part. The following software products are being developed for its implementation: «Elementary system with mask», «Directed generating system with behavior», «Content masks for elementary system with mask», «Entropies for content masks», «Entropies of content masks and 1 simplication», «Entropies for content masks and 2 simplication».

ЗМІСТ

ВСТУП

На сьогоднішній день існує та продовжує невпинно зростати велика кількість інформації різної фізичної природи. Отриману інформацію потрібно за допомогою різних підходів та методів зуміти обробити, проаналізувати та отримати певні результати, які мають адекватно описати початкову модель, побудовану на основі емпіричних даних. Крім того, зазвичай, початкова система даних, отримана в результаті збору інформації, виявляється занадто складною для обчислювального алгоритму, навіть при використанні потужних паралельних обчислювальних систем. У такому випадку необхідно використовувати деякий математичний апарат, який дозволяє спростити отриману систему даних з мінімізацією втрат важливої інформації для досягнутого спрощення складності. Проблематика рівня складності виникає, оскільки реальні об’єкти мають нескінченну кількість властивостей в загальному випадку, з яких дослідник на основі власного досвіду має вибрати суттєві властивості, які найбільш точно описують заданий об’єкт.

Одним із підходів, який дозволяє аналізувати системи даних, а також пропонує критерії спрощення отриманих математичних моделей реальних об’єктів, є системний аналіз, основною перевагою якого є досліджування систем на об’єкті, які мають не тільки кількісні, а й якісні характеристики. На базі системного аналізу лежить системний підхід, який застосовується до вирішення поставленої проблеми та має технічну основу, яка визначає процес автоматизації та алгоритмізації побудови системи на об’єкті та подальшого її спрощення за заданими критеріями, які будуть розглянуті у третьому розділі роботи.

Отже, першим етапом роботи ставиться визначення системи на об’єкті, яка формується завдяки виділеним властивостям під час спостережень за реальним об’єктом та на основі термінологічного апарату формується у вигляді таблиці, яка складається з емпіричних даних. Згодом починається процес обробки даних на основі вибрано апарату. У випадку, коли визначена обробка даних не задовольняє дослідника, що може бути пов’язано із занадто великим проміжком використаного часу, то задається спрощення, відповідно до якого дані аналізуються повторно. Крім того, після спрощення початкової системи дослідник може порівняти отримані результати із результатами до спрощення та на основі цього порівняння вибрати новий спосіб обробки, чи навіть визначити нову множину суттєвих властивостей з мінімальними втратами важливої інформації.

Мета роботи полягає в розв’язанні проблеми обчислювальної складності, відповідно до якої необхідно побудувати математичну модель складного об’єкту за допомогою методології системного аналізу та виконати спрощення отриманої системи на основі двох критеріїв спрощення.

Для досягненої мети в роботі визначені наступні завдання:

– дати характеристику етапам проведення дослідження на основі емпіричних даних;

– формалізувати побудову та дослідження системи на об’єкті, та її подальшого розбиття на системи з поведінкою різних типів;

– визначити види нечітких мір та на основі ймовірнісної міри ввести Шенноновську ентропію, як ступінь недетермінованості для аналізу оптимальності систем різних типів;

– визначити підходи до розв’язання проблеми обчислювальної складності на основі критеріїв спрощення;

– розробити програмні продукти, які дозволяють автоматизувати та алгоритмізувати побудову та спрощення визначених систем.

Об’єктом дослідження є математична модель складної системи.

Для реалізації поставлених задач у першому розділі роботи розглядуються та класифікуються етапи проведення емпіричного дослідження.

У другому розділі наводяться основні теоретичні аспекти теорії систем, такі як поняття об'єктів, змінних і параметрів, формування початкової системи, систем з поведінкою нейтрального та направленого типів із породженими змінними, а також задається класифікації мір нечіткостей та виділяється клас ймовірнісних невизначеностей, відповідно до якого вводиться ступінь недетермінованості систем.

Третій розділ присвячено одному з можливих розв'язків проблеми обчислювальної складності за допомогою двох критеріїв спрощення початкової системи.

У роботі особисто автором для алгоритмізації та автоматизації побудови визначених типів систем з їх подальшим спрощенням та знаходженням їх складності, розроблено програмні продукти: «Elementary system with mask», «Directed generating system with behavior», «Content masks for elementary system with mask», «Entropies for content masks», «Entropies of content masks and 1 simplication», «Entropies for content masks and 2 simplication».

1 ВІДОМОСТІ ПРО СТАН ПИТАННЯ І ПОСТАНОВКА ПРОБЛЕМИ. ОСНОВНІ ЕТАПИ ПОБУДОВИ МАТЕМАТИЧНОЇ МОДЕЛІ ОБ’ЄКТА ЗА ЕМПІРИЧНИМИ ДАНИМИ

Даний розділ присвячено висвітленню питань, які пов’язані з поставленою проблематикою, один з варіантів розв’язку якої базується на основі системного аналізу, що містить методологічний апарат, який дозволяє отримати розв’язки систем з високою обчислювальною складністю, завдяки спрощенню початкової системи з можливістю її подальшого аналізу та без істотних втрат важливої інформації при досягнутому рівні спрощення. Крім того, у 1 розділі наведено етапи проведення емпіричних досліджень. Визначено підходи до формування математичної моделі об’єкта за емпіричними даними.

1.1 Стан питання і постановка проблеми

Починаючи с XX ст. число комплексних проектів і проблем, які потребують участі спеціалістів, які у свою чергу володіють знаннями у суміжних сферах діяльності, почала невпинно зростати. Це вплинуло на появу і розвиток узагальнюючих наукових напрямів, а поняття системи перетворилося в спеціальну загальнонаукову категорію.

Теорія систем і системний підхід виникли, як узагальнення дисциплін, пов’язаних з дослідженням проектування складних об’єктів різної природи. Їх основоположником вважають біолога Л. фон Берталанфі [1-5], який в 30-і рр. XX ст. ввів поняття відкритої системи і сформулював основні ідеї та закономірності узагальнюючого напряму, названого теорією систем. Пізніше основоположниками розвитку теорії систем були В. Г. Афанасьєв, В. М. Садовський, В. С. Тюхтін, А. І. Уйомов [1-5] та ін. Ними були розроблені концептуальні основи, термінологічний апарат, досліджено закономірності функціонування і розвитку складних систем. Філософами В. М. Садовським, А. І. Уйомовим, Ю. А. Урманцевим, В. С. Тюхтіним і ін. був запропонований ряд варіантів теорії систем.

Зокрема, сучасна теорія систем свідчить про те, що до XX ст. наука займалась в основному вирішенням дуже простих систем, які зазвичай складались не більше ніж з 2 змінних. Під час аналізу таких систем, які вважались складними, у них намагались виявити приховане просте рішення. Це було реалізовано завдяки виділенню з множини усіх факторів кількох суттєвих чинників, залишаючи несуттєвими усі інші. Таке розділення факторів дозволяло досліднику ввести експериментально виправдані спрощення і, таким чином, розглядати досліджувані характеристики ізольовано від інших.

Проте, в умовах розвитку науково-технічного прогресу кількість змінних системи і як наслідок складність систем, які були математичними моделями реальних об'єктів, почали невпинно зростати.

Узагалі, у випадках систем з підвищеною складністю, для їх розв'язку або виконується пошук більш простої системи, або намагаються спростити початкову систему, на основі методологічного апарату. Однак, у більшості випадків, при розгляді складних систем, процес спрощення складності системи уникнути неможливо. Так, наприклад, якщо задача, пов'язана з деякою дуже складною системою, яку є можливість розв'язати за допомогою обчислювальної техніки без спрощення, то фінальна відповідь все одно має бути спрощена для того, щоб користувач міг нею скористатися. У загальному випадку, спрощення складної системи пов'язане з мінімізацією втрат важливої інформації для досягнутого спрощення складності.

Найбільш конструктивним із напрямів системних досліджень, який містить методологію спрощення обчислювальної складності систем, вважається системний аналіз, перше виникнення якого відбулось у роботах корпорації RAND в зв'язку із завданнями військового управління в 1960-і рр. [5]. Поширення у вітчизняній літературі системний аналіз набув після перекладу книги С. Оптнера [1].

Узагальнюючи різні точки зору системний аналіз є прикладним напрямом теорії систем, основними властивостями якого є:

– використання в якості методу дослідження розчленування великої невизначеності на більш доступні для огляду частини, які краще піддаються дослідженню, при збереженні цілісного (системного) уявлення про об'єкт дослідження і проблемної ситуації (завдяки поняттям мети і утворенням цілей).

– розглядання процесу постановки задачі і використання не тільки формальних методів, а й методів якісного аналізу;

– можливість застосування у тих випадках, коли має місце велика початкова невизначеність проблемної ситуації.

У 70-і рр. ХХ ст. виникла ще одна причина необхідності появи системного аналізу [2]. У міру продовження розвитку науково-технічного прогресу ускладнилися вироби і технології виробництва промислової продукції, розширилася її номенклатура і асортимент, збільшилась частота змінюваності виробів і технологій, зросла наукоємність продукції, підвищились потреби населення. Все це може призвести до ускладнення взаємовідносин людини з природою, виснаження ресурсів Землі, екологічним проблемам. В результаті ускладнилися процеси управління економікою, виникла необхідність управління самим науково-технічним прогресом.

Зокрема, у США, з 50-х рр. ХХ ст. проводилися інтенсивні наукові дослідження з цієї проблеми в спеціальних, не прибуткових корпораціях (типу корпорації RAND). Результатом цих досліджень стало створення першої методики системного аналізу – ПАТЕРН, в основі якої лежать формування та аналіз «дерева цілей» [2].

На сьогоднішній день, в умовах розвитку науки, техніки та інформаційних технологій, роль методів і моделей системного аналізу, як найбільш конструктивного напряму системних досліджень, зростає, відповідно підвищується необхідність розвитку цих методів і наближення їх до практичних потреб.

1.2 Проведення класичного емпіричного дослідження

Згідно з [6-13] перед проведенням емпіричного дослідження потрібно визначити:

а) об'єкт дослідження, який є частиною світу, що визначається як одне ціле протягом відомого періоду часу і є прийнятним для будь-якого конкретного дослідження;

б) мета дослідження об'єкта, яка повинна задавати набір питань про заданий об'єкт, на які дослідник хоче отримати відповіді;

в) обмеження, при яких проводиться дослідження. Вони можуть характеризувати можливості вибору інструментів, а також мати деякі обмежені фінансові можливості і час, людські ресурси і потужність обчислювальної техніки, правові, моральні та інші норми, яких має дотримуватися дослідник під проведення дослідження.

Основні етапи емпіричного дослідження, згідно Дж. Кліру [10], зображені на рис. 1.1.

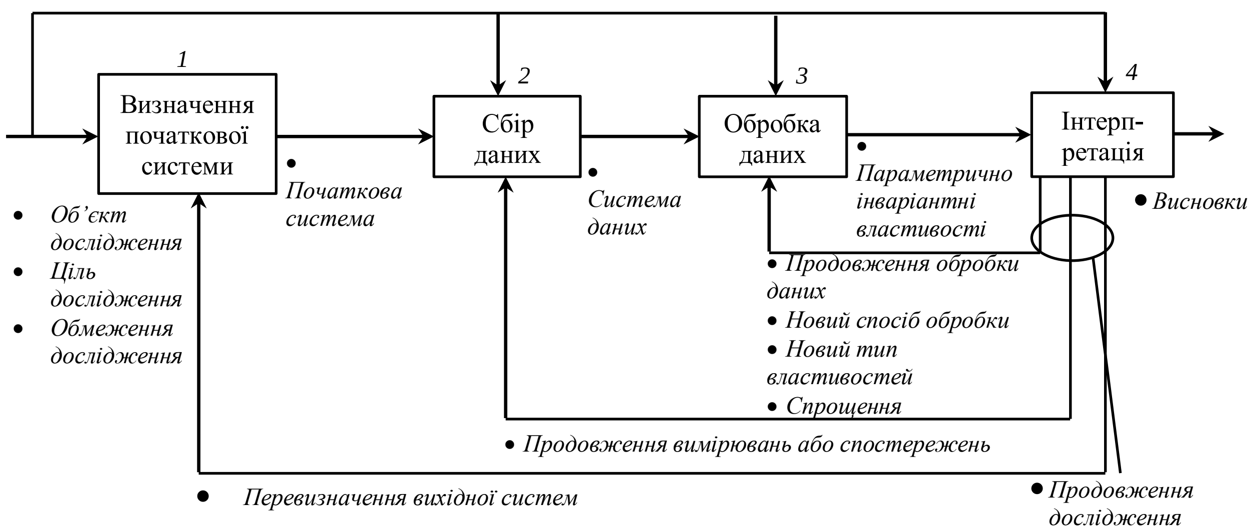

Рисунок 1.1 – Основні етапи емпіричних досліджень систем

На першому етапі будь-якого емпіричного дослідження розглядають визначення початкової системи на заданому об'єкті [8]. Вибір такої системи потребує у дослідника досвіду і знань в досліджуваній області, а також спеціального інструмента дослідження. Це пов'язано з вибором єдиної початкової системи з усіх можливих. При чому, ця система має якнайкраще відповідати цілі дослідження і задовольняти обмеженням, які задаються в системі.

При такому виборі можуть виникнути дві проблеми [7]: вибір властивостей і суттєвих властивостей (баз), та вибір каналів спостереження для них. Вибір властивостей і баз об'єкту є найважливішим рішенням, яке приймається в процесі емпіричного дослідження, оскільки цей вибір має вплив на всі наступні етапи цього процесу. Будь-який вибір властивостей для емпіричного дослідження завжди спирається на відповідну теорію, якою має володіти дослідник.

Після етапу визначення властивостей і баз, починають процес введення для них каналу спостереження, який буде задавати розбиття початкової множини проявів властивостей або значень параметра. Згідно з [6], будь-яке таке розбиття називають роздільною формою. Іноді визначити математично роздільні форми не є можливим, але завжди можливо визначити, чи є одна роздільна форма уточненням іншої. Подібне порівняння відбувається не на базі математичної теорії, а за допомогою порівняння відповідних процедур вимірювання.

Наступним кроком є збір даних. Цей процес [10] зводиться до спостереження або вимірювання відібраних властивостей при певних значеннях баз і зберігання цих спостережень в деякій формі. У випадку, коли дослідник має можливість корегувати деякі властивості, він може це використати. Тоді, властивості, якими він збирається керувати називають вхідними властивостями [8], що задають спрямовану початкову систему. Після цього, дослідник проводить ряд експериментів, в яких, згідно з винайденою експериментальною стратегією задаються початкові властивості системи. В результаті, отримується система даних.

На підставі визначеної системи даних відбувається обробка та аналіз експериментальних даних. Під час цього процесу розглядається визначення параметрично-інваріантних властивостей змінних, які дозволяють представляти дані у обмеженій формі. Для цього може використовуватися або повна множина даних для виведення якісних параметрів інваріантних властивостей, або спочатку обробляється тільки частина даних, а решта зберігається для подальшої перевірки.

Отже, починає існувати відповідний ряд параметрично-інваріантних властивостей, які мають спільні прояви. Кожна така властивість описує обмеження, накладене на змінні початкової системи, які не змінюються в межах параметричної множини.

Якщо дані оброблені, а також визначені параметрично-інваріантні властивості, їм необхідна інтерпретація відносно мети дослідження. Тому проводиться аналіз їх ефективності для пошуку відповідей на поставлені в дослідженні питання. Якщо відповідь отримана, дослідження вважається успішним і завершеним, а дослідник, нарешті, може підвести висновки роботи і розробити остаточний звіт. Якщо ж це не так, то можна спробувати обробити дані іншим способом. Цей процес може повторюватись кілька разів, як для того ж рівня пошуку параметрично-інваріантних властивостей, так і для інших епістемологічних рівнів. У результаті, дослідник отримує набір систем, які породжують системи більш високого рівня, кожна з яких адекватно представляє дані.

Відповідно до [6-14], якщо система, отримана в результаті проведення аналізу даних, занадто складна для повного дослідження, то за допомогою засобів спрощення різних типів по критеріям, зазначеним користувачем, отримують її більш простий аналог.

Після обробки даних та інтерпретації отриманих властивостей дослідник може зібрати додаткові дані для підвищення якості отриманих властивостей, або переглянути їх на основі нових даних. Подібний повторний збір даних змінює систему даних, але залишає без змін початкову систему. Однак, дослідник може перевизначити початкову систему повністю. У цьому випадку необхідно повторити весь процес для нової початкової системи.

Отже, всю процедуру емпіричного дослідження систем [6-14], описують наступним чином:

а) існує об’єкт, мета і обмеження емпіричного дослідження; об’єкт визначається початковою системою;

б) для початкової системи збираються дані і задаються в зручному вигляді (наприклад масиву даних);

в) дані обробляються для визначення параметрично-інваріантних властивостей;

г) отримані властивості інтерпретуються відповідно до мети дослідження і виконуються остаточні висновки або дослідження повторюється з етапів, зазначених вище.

Для програмного продукту ставиться умова вміння аналізувати всі задачі, які пов’язані з етапом обробки даних.

Отже, у даному розділі розглянуто необхідність розв’язку проблеми обчислювальної складності систем, на основі методології системного аналізу, який дозволяє мінімізувати втрати важливої інформації, при обраному спрощенні складності систем. Крім того, особливу увагу приділено теоретичній базі, пов’язаній з побудовою досліджуваного об’єкта за допомогою емпіричних даних.

Виділені теоретичні методики будуть застосовуватись при автоматизації та алгоритмізації спрощення системи на основі емпіричних даних у 4 розділі роботи.

2 ТЕОРЕТИЧНА МЕТОДИКА ПОБУДОВИ МАТЕМАТИЧНОЇ МОДЕЛІ СИСТЕМИ ЗА ЕМПІРИЧНИМИ ДАНИМИ З ЇЇ ПОДАЛЬШИМ АНАЛІЗОМ

Даний розділ роботи присвячено висвітленню етапів побудови та дослідження початкової системи на об’єкті, з подальшим формуванням на її основі примітивних систем нульового епістемологічного рівня, а також систем з поведінкою різних типів. Крім того, у 2 розділі надані визначення і методи отримання змінних та їх параметрів, визначено різні типи систем даних, розглянуто класи нечітких мір та введено критерій оптимальності математичної моделі досліджуваного об’єкта.

2.1 Побудова початкової системи на об’єкті

2.1.1 Формування першої примітивної системи – системи на об’єкті

Перш ніж розглядати питання формування першої примітивної системи на об’єкті, введемо основні поняття та терміни, які застосовуються при побудові вказаної системи.

Об’єкт, як відомо [8] є частиною світу, яка виділяється як одне ціле протягом деякого проміжку часу. Ґрунтуючись на цьому визначенні, об’єкти бувають матеріальними, які у свою чергу діляться на природні та штучні, й абстрактними.

Очевидно, що всі навколишні об’єкти мають безліч властивостей, кожну з яких можна досліджувати. Тому, майже всі об’єкти неможливо вивчити до кінця, що приводить до пошуку обмежено малого числа характеристик, які найточніше описують заданий об’єкт. Після їх вибору необхідно визначити процедуру вимірювання (спостереження) кожної властивості окремо, яка, в свою чергу, задає абстрактну змінну, яка є відображенням відповідної властивості.

Отже, можна прийти до висновку, що систему потрібно розглядати не як реальну річ, а як відтворення деяких обмежених властивостей об'єкта [9]. З кожною такою властивістю об'єкта пов'язана множина її проявів. Щоб визначити можливі зміни проявів властивості потрібна її множина спостережень. При цьому, окремі властивості, які здійснюються за допомогою однієї і тієї ж процедури спостереження, мають відрізнятися одна від одної. Будь-яку істотну властивість, яка використовується для визначення відмінностей в спостереженнях одної і тої ж властивості, називають базою [10]. Типовими базами, є час, простір, група та їх комбінації [7].

Згідно з [10], правильно відібрані бази мають відповідати вимогам:

а) вони мають застосовуватись до всіх властивостей системи, на якій визначені;

б) вибір бази має ґрунтуватись на відповідній меті, для якої будується дана система;

в) кожен елемент множини бази має визначати один і тільки один прояв довільної властивості;

Після визначення властивостей, баз та їх взаємозв'язків, можна визначити систему об'єкта, яка представляє собою множину властивостей, таких, що з кожним із них пов'язані відповідно множини їх проявів і баз, з якими пов'язана множина елементів такої системи на об'єкті.

Система об'єкта [8-10], зазвичай набуває вигляду

$$O = \left(\{(a_i, A_i) | i \epsilon N_n\}, \{b_j, B_j | j \epsilon N_m\}\right), \qquad (2.1)$$

де $N_n = \{1,2, \dots, n\}$, $N_m = \{1,2, \dots, m\}$; a_i і A_i, – властивість і множина її проявів відповідно; b_j і B_j – база і множина її елементів; O – система об'єкта.

Множини A_i і B_j з рівняння (2.1) для деяких властивостей і баз іноді визначаються нескладно, однак на практиці у багатьох випадках ці множини невідомі. Проте, їх завжди можна пов'язати з повністю визначеними множинами за допомогою конкретних процедур спостереження або вимірювання.

2.1.2 Формування другої та третьої примітивних систем, – конкретної та загальної системи, що представляють данні

Спочатку введемо поняття, які пов'язані з формуванням конкретної та загальної системи, що представляють данні.

Згідно Дж. Кліру [10], змінна є образом властивості, який визначається конкретною процедурою спостереження або вимірювання. Будь-яка змінна потребує назву, яка відрізняє її від інших розглянутих змінних, і пов'язує з множиною величин, через які вона себе проявляє. Такі величини називають станами змінної, а всю їхню множину – множиною станів.

Параметр, у свою чергу, згідно з [8], є операційним уявленням бази. Кожен параметр має унікальне ім'я, з яким пов'язана множина, яку називають його параметричною множиною, а її елементи – значеннями параметра.

Різні спостереження однієї і тієї ж змінної мають відрізнятися за значеннями параметрів [9]. При використанні більше одного параметра, їх загальна параметрична множина складається з декартового добутку окремих параметричних множин. Кожне конкретне значення параметра має ідентифікувати лише одне спостереження відповідних змінних.

Розбіжність у властивостях змінних або параметрів, які мають істотне методологічне значення, як відомо, називають методологічними відмінностями [8-10].

Узагальнені змінні і параметри є абстрактними величинами, які не визначаються через властивості або бази. Їх множини станів і параметричні множини задаються стандартним чином.

Відомо, що множина станів узагальненої змінної має відображатися ізоморфно [10, 14] в елементи множини станів конкретної змінної. Будь-яке ізоморфне відображення такого роду називають конкретизацією узагальненої змінної [8-10] (або узагальненого параметра), а зворотне відображення – абстрагуванням конкретної змінної (або конкретного параметра).

Розглянемо більш детально конкретизацію та абстрагування конкретних змінних. Нехай v_i, V_i, $\mathbb{V}_i$ – узагальнена змінна, її множина станів і множина математичних властивостей, визначених на v_i, відповідно. Через $\dot{v}_i$, $\dot{V}_i$, $\mathbb{V}_i$ позначають ті ж характеристики конкретної змінної, що конкретизують змінну v_i. Через w_j, W_j, $\mathbb{W}_j$ задають узагальнений параметр, множину його станів і множину математичних властивостей, визначених на параметрі w_j, відповідно, а через $\dot{w}_j$, $\dot{W}_j$, $\mathbb{W}_j$ – ті ж характеристики конкретного параметра, отримані за допомоги конкретизації параметра w_j.

Як відомо [10], каналом спостереження називають будь-яку операція, яка вводить конкретну змінну як образ властивості. Канал спостереження, за допомогою якого властивість a_i представляється змінною $\dot{v}_i$, реалізується [10] функцією

$$o_i: A_i \to \dot{V}_i. \tag{2.2}$$

Ця функція є гомоморфною відносно властивостей множин A_i і $\dot{V}_i$. Аналогічно, функція

$$\omega_j: B_j \to \dot{W}_j \tag{2.3}$$

задає уявлення бази b_j параметром $\dot{w}_j$.

Функції (2.2) і (2.3) є гомоморфними, щодо властивостей множин A_i і $\dot{V}_i$ та властивостей бази, а також властивостей множини $\mathbb{W}_j$ відповідно.

Функції o_i і ω_j можуть бути явно заданими каналами спостереження для деяких властивостей і баз. Проте, коли множини A і B невідомі, можна задати ці функції явно.

Відповідно до [8-10], задана узагальнена змінна v_i конкретизується змінної $\dot{v}_i$ тоді і тільки тоді, коли функція

$$e_i : V_i \to \dot{V}_i \tag{2.4}$$

існує і ізоморфна відносно математичних властивостей $\mathbb{V}_i$ [5-7].

Аналогічно, узагальнений параметр w_j конкретизується параметром $\dot{w}_j$ тоді і тільки тоді, коли функція

$$\varepsilon_j : W_j \to \dot{W}_j \tag{2.5}$$

існує і є ізоморфною відносно $\mathbb{W}_j$.

Функції, які є зворотними функціям e_i і ε_j, тобто

$$e_i^{-1} : \dot{V}_i \to V_i; \tag{2.6}$$

$$\varepsilon_j^{-1} : \dot{W}_j \to W_j, \tag{2.7}$$

задають абстрагування відповідно $\dot{v}_i$ і $\dot{w}_j$ [5-7, 12, 13].

Для того, щоб визначити нечіткий канал спостереження, спочатку встановлюють чіткий канал спостереження o_i, який є окремим випадком нечіткого. Якщо

$$\tilde{o}_i(x, y) = \begin{cases} 1, \text{якщо } o_i(x) = y; \\ 0, \text{в іншому випадку,} \end{cases}$$

то $\tilde{o}_i$ задає чітку функцію з A_i в $\dot{V}_i$, яка є ідентичною o_i [7].

При розгляді баз вводять функцію [8-10]

$$\widetilde{\omega}_j: B_j \times \dot{W}_j \to [0, 1], \tag{2.8}$$

яка опирається на співвідношенні (2.3), де $\widetilde{\omega}_i(x, y)$ є степенем достовірності того, що x належить блоку розбиття B_j/ω_j, який представлений значенням y параметра $\dot{w}_j$.

Отже, на практиці достатньо використовувати чіткий канал спостереження ω_j для баз. Однак, для властивостей можна застосовувати як чіткі, так і нечіткі канали спостереження (o_i і $\tilde{o}_i$), між якими вибирається більш доцільний для конкретної задачі.

Як відомо [8-10, 13, 14], властивості, конкретні і загальні змінні, а також бази, конкретні і загальні параметри є компонентами відповідно трьох примітивних систем, серед яких система об'єкта, конкретна та загальна системи, що представляють данні, які разом з відношеннями між ними утворюють початкову систему. Перша з цих трьох формально визначається рівнянням (2.1). Дві інші є примітивними системи, що мають той же вигляд, що і система об'єкта, різниця між якими полягає в тому, що їх компонентами є змінні і параметри, а не властивості і бази.

У роботах [8-10], зазначається, що $\dot{I}$ і I позначають конкретну та загальну системи відповідно, що представляють данні і задаються у вигляді рівностей

$$\dot{I} = \left(\left\{\left(\dot{v}_i, \dot{V}_i\right) | i \in N_n\right\},\ \left\{\left(\dot{w}_j, \dot{W}_j\right) | j \in N_m\right\}\right); \tag{2.9}$$

$$I = \left(\left\{(v_i, V_i) | i \in N_n\right\},\ \left\{\left(w_j, W_j\right) | j \in N_m\right\}\right), \tag{2.10}$$

де для $\forall i \in N_n$ і $j \in N_m$ властивість a_i відповідає змінним $\dot{v}_i$, v_i, а база b_j – параметрам $\dot{w}_j$, w_j.

За допомогою повного каналу спостереження, який складається з каналів спостереження, визначених для кожної властивості окремо або бази системи об'єкта, задають відношення між системою об'єкта і конкретної системою, що представляє данні. Чіткий повний канал спостереження позначають через Q і згідно з [10] задають у вигляді формули

$$Q = \left(\left\{\left(A_i, \dot{V}_i, o_i\right) \middle| i \in N_n\right\}, \left\{\left(B_j, \dot{W}_j, w_j\right) \middle| j \in N_m\right\}\right). \tag{2.11}$$

Відношення між конкретною та загальною системами, що представляють дані задаються набором відображень конкретизації. Цей набір називається каналом конкретизації / абстрагування $\mathcal{E}$ [8-10] і визначається, як

$$\mathcal{E} = \left(\left\{\left(\dot{V}_i, V_i, e_i\right) \middle| i \in N_n\right\}, \left\{\left(\dot{W}_j, W_j, \varepsilon_j\right) \middle| j \in N_m\right\}\right). \tag{2.12}$$

2.1.3 Формальне визначення початкової системи на об'єкті та зазначення її основних типів

Оскільки було сформульовано систему об'єкта O (2.1), конкретну систему $\dot{I}$ (2.9) і загальну систему I, що представляють данні, а також повний канал спостереження Q (2.11) та канал конкретизації / абстрагування $\mathcal{E}$ (2.12), то, як відомо з робіт [8, 10], формальне визначення початкової системи на об'єкті набуває вигляду:

$$S = \left(O, \dot{I}, I, Q, \mathcal{E}\right). \tag{2.13}$$

Початкова система є джерелом емпіричних даних [10], тобто джерелом абстрактних уявлень про явища реального світу.

При визначенні системи на об'єкті розрізняють два типи змінних (або відповідних властивостей) – вхідні і вихідні [8-10, 13]. Значення вхідних змінних задаються ззовні, при цьому фактори, які мають вплив на їх значення називають середовищем, в той час як, значення вихідних змінних початкової системи розглядаються всередині системи.

Згідно робіт [8-10], на основі вхідних та вихідних змінних системи нульового епістемологічного рівня за типами ділять на:

– спрямовані, у випадку, коли вони містять вхідні та вихідні змінні;

– нейтральні, у яких відсутнє розділення змінних на вхідні та вихідні.

Очевидно, що початкова система, яка визначається рівнянням (2.13), як і три її примітивні системи $(O, \dot{I}, I)$ є нейтральними. Проте, існує направлений аналог таких систем. Для того, щоб їх перетворити в спрямовані системи, необхідно, щоб всі їх змінні (і властивості) були розділені на вхідні або вихідні. Для системи I таке представлення можна виконати за допомогою функції

$$u: N_n \to \{0,1\},$$

такої, що при $u(i) = 0$ змінна v_i є вхідною, а при $u(i) = 1$ – вихідною.

Будь-яку n-ку

$$u = (u(1), u(2), \dots, u(n)),$$

що задає статус всіх змінних системи, називають визначником входу-виходу [8-10, 13, 14]. Для n змінних всього може бути 2^n представлень входів-виходів.

Змінним $\dot{v}_i$, і властивостям a_i відповідає той же визначник входу-виходу. Примітивні системи $O, \dot{I}, I$ перетворюються в спрямований аналог при додаванні до них визначника входу-виходу. Спрямовані аналоги нейтральних систем позначають наступним чином:

$$\hat{O} = \left(\{(a_i, A_i)|i \in N_n\}u\{\left(b_j, B_j\right)|j \in N_m\}\right); \quad (2.14)$$

$$\hat{\dot{I}} = \left(\{\left(\dot{v}_i, \dot{V}_i\right)|i \in N_n\}u\{\left(\dot{w}_j, \dot{W}_j\right)|j \in N_m\}\right); \quad (2.15)$$

$$\hat{I} = \left(\{(v_i, V_i)|i \in N_n\}u\{\left(w_j, W_j\right)|j \in N_m\}\right), \quad (2.16)$$

де $\hat{O}, \hat{\dot{I}}, \hat{I}$ – спрямовані аналоги нейтральних систем $O, \dot{I}, I$ [7, 10]. Тоді, згідно з роботами [8-10], спрямована початкова система визначається п'ятіркою

$$\hat{\mathrm{S}} = \left(\hat{O}, \hat{\dot{\mathrm{I}}}, \hat{\mathrm{I}}, \mathbb{Q}, \mathbb{E}\right). \quad (2.17)$$

2.2 Формування системи даних

Початкова система визначається згідно з [10], як алгоритм, за допомогою якого можуть бути зроблені спостереження відібраних властивостей. Якщо канал спостереження чіткий, то будь-яке реальне спостереження відображається у вигляді впорядкованої пари, що складається із значення повного параметра, при якому було зроблено спостереження, і зафіксованого повного стану змінних. Оскільки, при одному значенні параметра може бути зроблено тільки одне спостереження, множина цих упорядкованих пар є функцією, що відображає повну параметричну множину в повну множину станів [10]. Тобто, ця функція представляє собою чіткі дані, які відповідно до [8] задаються функцією

$$d\colon \mathrm{W} \to \mathrm{V}, \quad (2.18)$$

де

$$\mathrm{W} = W_1 \times W_2 \times \ldots \times W_m;$$
$$\mathrm{V} = V_1 \times V_2 \times \ldots \times V_n.$$

У той час, як система I описує тільки потенційні стани змінних, функція d дає інформацію про їх дійсні стани на необмеженій параметричній множині. Систему I у поєднанні з функцією d розглядають, як систему більш високого епістемологічного першого рівня, яку називають системою даних [9], позначають D і задають у вигляді:

$$D = (I, d). \tag{2.19}$$

З (2.19) можна отримати систему даних з семантикою, яку знаходять за допомогою рівності [10]:

$${}^{S}D = (S, d), \tag{2.20}$$

яку отримують шляхом заміни системи I відповідною початковою системою S, де d – та ж сама функція, що і в рівнянні (2.19).

Залежно від розглянутої задачі, функція d може бути визначена принаймні трьома різними способами [9]. По-перше, вона може бути результатом спостережень або вимірювань. По-друге, її можна вивести з систем більш високих рівнів. По-третє, вона може бути визначена самим користувачем.

Так як системи даних D, ${}^{S}D$ визначені через нейтральну систему I, що представляє дані і нейтральну початкову систему S, то вони є нейтральними. Їх спрямовані аналоги мають вигляд:

$$\widehat{D} = \left(\hat{I}, d\right); \tag{2.21}$$
$${}^{S}\widehat{D} = \left(\hat{S}, d\right), \tag{2.22}$$

які називаються системами даних без семантики і з семантикою відповідно [7].

2.3 Формування систем з поведінкою нейтрального на направленого типів

2.3.1 Побудова нейтральної системи та породжуючої системи з поведінками

Термін «поведінка» у роботах [8-10, 15] використовується для отримання характеристики загального параметрично-інваріантного обмеження на змінні узагальненої системи, що представляє дані і на деякі додаткові абстрактні змінні. Додаткові змінні визначаються на параметричній множині за допомогою правила зрушення [10]. Таке правило застосовують або до змінної з початкової системи, або до внутрішньої змінної. Системи, що містять такі обмеження, називають породжуючими системами [8-10]. Поведінка є одною з форм подання цього обмеження.

Сусідство на упорядкованій параметричній множині називають маскою [10] і визначають через змінні, параметричну множину і набір правил зрушення на ній. Правило зрушення r_j у роботах [8-10] задається як однозначна функція

$$r_j\colon \mathrm{W} \to \mathrm{W}. \tag{2.23}$$

Так як, параметрична множина є повністю упорядкованим набором цілих додатних чисел, то будь-яке правило зрушення задають у вигляді:

$$r_j(w) = w + \rho, \tag{2.24}$$

де ρ – ціла константа, при чому $\rho \in Z$. Якщо $\rho = 0$, то r_j називається тотожним правилом зрушення [10].

Нехай задано узагальнену систему I, що представляє дані та визначається рівнянням (2.10). Через V позначають множину змінних з I, а через R – набір правил зрушення, що розглядаються для цих змінних. Тоді множину змінних $S = \{s_1, s_2, \dots\}$, називають вибірковими змінними [8-10] і визначають за допомогою рівнянь

$$s_{k,w} = v_{i,r_j(w)}, \tag{2.25}$$

де $v_i \in V$ – деякі змінні, $r_j \in R$ – правила зрушення, $s_{k,w}$ – стан вибіркової змінної s_k при значенні параметра w, $v_{i,j_i}(w)$ – стан змінної v_i отриманий для заданого w при застосуванні правила зрушення r_j.

Для повністю впорядкованої параметричної множини, правила зрушення якої мають вигляд (2.24), рівняння (2.25) може бути записано в більш певному вигляді

$$s_{k,w} = v_{i,w+\rho}. \tag{2.26}$$

Оскільки будь-яке правило зрушення з набору R може бути застосоване до будь-якої змінної з множини V, то множина всіх можливих вибіркових змінних представляється декартовим добутком $V \times R$ [9]. Насправді, вибіркові змінні, які розглядаються, характеризуються відношенням

$$M \subseteq V \times R \tag{2.27}$$

таким, що будь-якій парі $(v_i, r_j) \in M$ відповідає одне рівняння з (2.25). Відношення M представляє схему сусідства на параметричній множині, у термінах якого визначені вибіркові змінні і називають маскою [10]. Для введення ідентифікаторів вибіркових змінних k вводять однозначну функцію кодування, яка задається у вигляді відношення

$$\lambda: M \to N_{|M|}, \tag{2.28}$$

де $|M|$ – це потужність множини M.

У свою чергу, декартовий добуток

$$C = S_1 \times S_2 \times \ldots \times S_{|M|}$$

представляє собою повну множину станів вибіркових змінних [9].

Якщо ж виділені параметричні множини повністю впорядковані, то їх, наприклад, можна позначити через T, а їх елементи t, $t \in T$. Тоді рівняння (2.26) можна замінити еквівалентним:

$$s_{k,t} = v_{i,t+\rho}. \tag{2.29}$$

На практиці, буває зручно розбити маску M на підмаски M_i [13-14], кожна з яких пов'язана з однією змінною v_i з подібної системи. Формально

$$M_i = \{(\alpha, \beta) | (\alpha, \beta) \in M, \alpha = v_i\}. \tag{2.30}$$

У матричному поданні M, підмаски M_i є рядками.

У будь-якій масці один стовпчик відповідає тотожному правилу зрушення (при $\rho = 0$). Вибіркові змінні, що пов'язані з цим стовпчиком, ідентичні базовим змінним початкової системи, що представляє дані. Його в масках називають довідником [10, 13, 14].

Найпростіший спосіб задання певної маски полягає в перерахуванні всіх повних станів відповідних вибіркових змінних. У загальному вигляді подібний перелік є підмножиною декартового добутку C, який визначається функцією

$$f_B: C \to \{0,1\}, \tag{2.31}$$

такою, що $f_B(\text{c}) = 1$, якщо стан c входить до переліку, і $f_B(\text{c}) = 0$ в протилежному випадку. Таку функцію f_B називають функцією вибору станів вибіркових змінних з множини всіх потенційних. Оскільки, подібний вибір дає деякі відомості про поведінку цих змінних, функцію f_B називають функцією поведінки [8-10].

Функція f_B є параметрично інваріантною, оскільки вона визначає стани c, що зустрічаються у матриці, але не визначає значення параметра, при якому вони мають місце. До того ж, f_B незмінна для всіх типів функцій поведінки і визначається через маску. Звідси випливає, що деяка система F_B, що характеризує параметрично-інваріантне обмеження на множину змінних через функції поведінки, визначається трійкою

$$F_B = (I, M, f_B), \tag{2.32}$$

де I – узагальнена система, що представляє дані; M – маска, визначена на I; f_B – функція поведінки, визначена через M і I.

Таку систему називають системою з поведінкою [10].

Будь-яка система з поведінкою, визначена (2.32), не містить опису того, як використовувати обмеження на змінні системи I для породження даних. Згідно робіт [8-10, 13, 14], для розробки такого опису вибіркові змінні розбивають на дві підмножини:

а) стан яких породжується з обмеження, їх називають породженими;

б) стан яких використовуються як умова в процесі генерації, тобто породжуючі змінні.

Для початкової системи з поведінкою одним із способів розбиття змінних на породжені та породжуючі є визначення для даної маски M двох підмасок M_g і $M_{\bar{g}}$ [10], які характеризуються рівністю:

$$M_G = \left(M, M_g, M_{\bar{g}}\right), \quad (2.33)$$

де $M_g, M_{\bar{g}} \subset M$; $M_g \cup M_{\bar{g}} = M$; $M_g \cap M_{\bar{g}} = \emptyset$.

Рівність (2.33) називають маскою породження, тобто це маска M і її розбиття на підмаски M_g і $M_{\bar{g}}$.

Спосіб представлення стану змінних $g \in G$ [10], що визначається через стан змінних $\tilde{g} \in G$, записують у вигляді функції

$$f_{GB}\colon \overline{\mathrm{G}} \times \mathrm{G} \to \{0,1\}, \quad (2.34)$$

де

$$f_{GB}(\bar{g}, g) = \begin{cases} 1, \text{якщо g або } \bar{\text{g}} \text{ визначені}, \\ 0, \text{якщо g або } \bar{\text{g}} \text{ не визначені}. \end{cases}$$

Цю функція називають породжуючою функцією поведінки [8-10].

Якщо маску M і функцію f_B з (2.32) замінити на M_G і f_{GB} відповідно, то отримують альтернативну систему

$$F_{GB} = \left(I, M_g, f_{GB}\right), \quad (2.35)$$

яку називають породжуючою системою з поведінкою.

Отже, процес побудови породжуючої системи F_{GB} складається з двох етапів [8-10, 15]:

а) для деякого значення $t \in T$ заданий стан $\tilde{g} \in \tilde{G}$; для визначення стану $g \in G$ при тому ж значенні, використовується функція f_{GB};

б) значення t замінюється на нове і переходять до етапу а) знову.

Зауваження: по-перше, на етапі а) передбачається, що при заданому значенні t, стан $\tilde{g}$ – відомий. При першому виконанні цього етапу він визначається користувачем як початкова умова. Проте, після цього все інше повністю визначається станами $\bar{g}$ і g, які пов'язані з попереднім значенням t. При цьому передбачається, що значення t повинні на етапі б) змінюватися відповідно до порядку, заданому на множині T. Таким чином, значення t замінюються або на $t+1$, або на $t-1$. У першому варіанті початкова умова має бути визначена для найменшого можливого значення t, а в другому – для найбільшого можливого значення t.

По-друге, існує тільки два змістовних розбиття маски M на M_g і $M_{\bar{g}}$, кожне з яких відповідає одному з двох порядків породження. Це випливає з необхідності породження даних в одному з двох порядків. Якщо дивитись на M_g графічно, то M_g – це множина елементів M, що знаходяться у правому або лівому краю маски M.

По-третє, для $\forall \bar{g} \in \bar{G}$ існує принаймні один стан $g \in G$, такий що $f_{GB}(\bar{g}, g) = 1$. Якщо допускається тільки один стан, то для будь-якої початкової умови дані породжуються однозначно. Такі системи називаються детермінованими [15]. При допусканні більш ніж одного стану, породження даних стає проблематичним, оскільки породжений стан не завжди однозначно визначений. Для детермінованих систем уявлення (2.34) функції поведінки f_{GB} може бути замінена на

$$f_{GB}: \bar{G} \rightarrow G. \tag{2.36}$$

2.3.2 Формування направленої та породжуючої системи з поведінкою

Для введення направлених систем з поведінкою множину вибіркових змінних ділять на 2 підмножини [8-10]:

а) множина вибіркових змінних, які визначаються навколишньою середою; тобто вхідні змінні v_i, для яких визначник входу – $u(i) = 0$;

б) інші вибіркові змінні, які пов'язані з заданою маскою.

Ці дві підмножини вибіркових змінних визначають шляхом розділення маски M на дві підмаски: M_e визначає змінні, які задаються навколишньою середою, а $M_{\bar{e}}$ – всі інші. Відомо, що трійка

$$\widehat{M} = (M, M_e, M_{\bar{e}}), \qquad (2.37)$$

де

$$M_e, M_{\bar{e}} \subset M, M_e \cup M_{\bar{e}} = M, M_e \cap M_{\bar{e}} = \emptyset,$$

визначає маску направленої системи з поведінкою [10].

Функція поведінки направлених систем характеризується відношенням:

$$\hat{f}_B: E \times \bar{E} \to [0,1], \qquad (2.38)$$

де $f_B(\bar{e}|e)$ – умовна ймовірність.

Тоді, направлена система з поведінкою [10] має вигляд

$$\widehat{F}_B = \left(\hat{I}, \widehat{M}, \hat{f}_B\right). \qquad (2.39)$$

Породжуючу функцію для направлених систем задають за допомогою розбиття $M_{\bar{e}}$ на дві підмножини – $M_g, M_{\bar{g}}$, для породжених та породжуючих змінних відповідно. Тоді, породжуюча маску для направлених систем [9, 15] задається, як

$$\widehat{M}_G = (M; M_e, M_g, M_{\bar{g}}), \tag{2.40}$$

де $\{M_e, M_g, M_{\bar{g}}\}$ – розбиття M, причому $\{M_g, M_{\bar{g}}\}$ розглядаються, як розбиття $M_{\bar{e}}$.

Для детермінованих систем f_{GB} можна записати, як

$$\hat{f}_{GB}: \mathrm{E} \times \overline{\mathrm{G}} \to \mathrm{G}.$$

Отже, направлена породжуюча система з поведінкою, згідно з [10, 15], визначається, як

$$\hat{F}_{GB} = \left(\hat{I}, \widehat{M}, \hat{f}_{GB}\right). \tag{2.41}$$

2.4 Визначення ступеня недетермінованості досліджуваної системи

Перед тим як розглядати ступінь недетермінованості систем, розглянемо основні класи нечітких мір. Відповідно до [17, 18] нечіткі міри можна поділити на 2 великих класи:

а) міри правдоподібності;

б) міри довіри (впевненості).

Міри правдоподібності містять підмножину ймовірнісних мір, підкласом яких є чітка можливість. Міри довіри (впевненості), у свою чергу, містять підмножину мір необхідності, які мають підклас чіткої необхідності (впевненості). Крім того, множини мір правдоподібності та мір довіри можуть перетинатися. У такому разі вони утворюють ймовірнісні міри.

Класи нечітких мір розглядаються у якості методологічних відмінностей і використовуються у породжуючих системах, а також у системах більш високих епістемологічних рівнів.

В роботі ступінь недермінованості системи визначається на підставі Шенноновської ентропії, яка будується на ймовірнісних невизначеностях. Тому є доцільним наведення основних характеристик та властивостей ймовірнісних нечіткостей.

Нехай P – множина всіх розподілів ймовірностей [10], які можуть бути визначені на скінченних множинах альтернативних виходів. Тоді, ймовірна міра нечіткості [16, 17, 18] – це функція

$$H: P \to [0, \infty),$$

яка має властивості:

а) симетричність – нечіткість є інваріантною відносно зміни ймовірностей;

б) розширюваність – нечіткість не змінюється при додаванні до розглянутої множини виходів, які мають ймовірність рівну нулю;

в) квазіадитивність – нечіткість спільного розподілу ймовірностей має бути не більше суми нечіткостей відповідних розподілів його компонентів;

г) адитивність – для розподілу ймовірностей будь-яких двох незалежних множин виходів, нечіткість спільного розподілу ймовірностей дорівнює сумі нечіткостей окремих розподілів ймовірностей.

д) неперервність – нечіткість має бути неперервною функцією на множині всіх своїх аргументів.

Відомо, що функція

$$H(f(x) | x \in X) = -a \sum_{x \in X} f(x) \log_b f(x)$$

відповідає усім властивостям, які описані вище [10].

Відношення

$$(f(x)|x \in X)$$

називається розподілом ймовірності для певної скінченної множини X альтернативних виходів x, де $a = const$, $b = const$, $b > 0$ і $b \neq 1$. Якщо виконується нормалізуюча властивість

$$H(0{,}5; 0{,}5) = 1,$$

то міра нечіткості визначається за формулою:

$$H(f(x)|x \in X) = -\sum_{x \in X} f(x) \log_2 f(x). \tag{2.42}$$

Функцію (2.42) називають *Шенноновською ентропією* [10, 17], яка визначає нечіткість в бітах. Вигляд $H(f(x)|x \in X)$ можна спростити до $H(x)$, так як скінченна множина X характеризується деяким розподілом ймовірностей.

У нейтральних, направлених, породжених і породжуючих системах, множиною виходів є множини C, G, $\bar{G}$, E, а розподіл ймовірностей задається функціями поведінки f_B, f_{GB}, $\hat{f}_B$, $\hat{f}_{GB}$, які визначаються формулами (2.31), (2.34), (2.38), (2.41) відповідно. Для спрощення запису, опустимо індекси і знак «^», тобто матимемо функції

$$f(c), f(g|\bar{g}), f(\bar{e}, e), f(g|e, \bar{g}),$$

Крім того, безумовною ймовірністю є

$$f(\bar{g}) = \sum_{c \succ \bar{g}} f(c), \tag{2.43}$$

де $c \succ \bar{g}$ означає, що $\bar{g}$ є підмножиною станів c [8-10, 13, 14].

Для направлених систем безумовні ймовірності знаходяться за формулою:

$$f(\bar{g}|e) = \sum_{\bar{e} \succ \bar{g}} f(\bar{e}|e). \tag{2.44}$$

Умовні ймовірності, які характеризують процес породження даних, пов'язані з основними і безумовними ймовірностями наступним чином:

$$f(g|\bar{g}) = \frac{f(c)}{f(\bar{g})}; \tag{2.45}$$

$$f(g|e, \bar{g}) = \frac{f(\bar{e}|e)}{f(\bar{g}|e)}, \tag{2.46}$$

де (2.45) задає зв'язок для нейтральних, а (2.46) для направлених систем.

За відомою породжуючою маскою для нейтральної системи, через яку визначаються множини станів G і $\overline{\mathrm{G}}$, породжуюча нечіткість $H(\mathrm{G}|\overline{\mathrm{G}})$ [10], задається як

$$H(\mathrm{G}|\overline{\mathrm{G}}) = -\sum_{\bar{g} \in \overline{\mathrm{G}}} f(\bar{g}) \sum_{g \in \mathrm{G}} f(g|\bar{g}) \log_2 f(g|\bar{g}). \tag{2.47}$$

Значення (2.47) визначає ступінь недетермінованості заданої нейтральної породжуючої системи з поведінкою.

Для направлених систем породжуючу нечіткість $H(\mathrm{G}|\mathrm{E} \times \overline{\mathrm{G}})$ знаходять за формулою

$$H(\mathrm{G}|E \times \overline{\mathrm{G}}) = -\sum_{e \in \mathrm{E}} \sum_{\bar{g} \in \overline{\mathrm{G}}} f(e, g) \sum_{g \in \mathrm{G}} f(g|e, \bar{g}) \log_2 f(g|e, \bar{g}), \tag{2.48}$$

яку доцільно використовувати, якщо направлена система отримана з нейтральної [10].

Формули (2.47) і (2.48) можна спростити. Так, наприклад, рівняння (2.47) може бути представленим наступним чином:

$$H(\mathrm{G}|\overline{\mathrm{G}}) = -\sum_{g\in\mathrm{G}} f(g|\bar{g})\log_2 f(g|\bar{g}) = -\sum_{\bar{g}\in\overline{\mathrm{G}}}\sum_{g\in\mathrm{G}} f(\bar{g})f(g|\bar{g})\log_2 f(g|\bar{g}) =$$

$$= -\sum_{\bar{g}\in\overline{\mathrm{G}}}\sum_{g\in\mathrm{G}} f(c)\frac{\log_2 f(g|\bar{g})}{f(\bar{g})} = H(C) + \sum_{\bar{g}\in\overline{\mathrm{G}}}\sum_{g\in\mathrm{G}} f(c)\log_2 f(\bar{g}) =$$

$$= H(C) + \sum_{\bar{g}\in\overline{\mathrm{G}}}\log_2 f(\bar{g})\sum_{g\in\mathrm{G}} f(c) = H(C) + \sum_{\bar{g}\in\overline{\mathrm{G}}} f(\bar{g})\log_2 f(\bar{g}) =$$

$$= H(C) - H(\overline{\mathrm{G}}).$$

Аналогічно, формулу (2.48) можна замінити рівнянням:

$$H(\mathrm{G}|\mathrm{E}\times\overline{\mathrm{G}}) = H(C) - H(\mathrm{E}\times\overline{\mathrm{G}}). \qquad (2.49)$$

Отже, в другому розділі було розглянуто поняття об’єкту дослідження, змінних і параметрів, а також систем, що їх представлять. Особливу увагу приділено побудові системи на об’єкті та подальшого формування на її основі породжуючих систем з поведінкою направленого і нейтрального типів. Крім того, було наведено основні класи нечітких мір, підкласом яких є ймовірнісні невизначеності, на базі яких було розглянуто поняття Шеннoновської ентропії, а також застосування вказаної ентропії для визначення ступеня недетермінованості систем, як критерія їх оптимальності.

За допомогою теоретичної частини другого розділу буде виконано аналіз автоматизації та алгоритмізації спрощення системи на об’єкті, що розглядається у 4 розділі роботи.

3 СПРОЩЕННЯ СИСТЕМ З ПОВЕДІНКОЮ

Перед спрощенням систем з поведінкою необхідно узагальнити поняття поведінки та власне поняття самої системи з поведінкою, із виділенням її видів і основних методологічних відмінностей для систем різних епістемологічних рівнів.

Дж. Клір в [10] використовує термін «породжуюча система» для загальної назви систем епістемологічної ієрархії 2 рівня. У даних системах обмеження на розглянуті змінні описуються з різних сторін і передбачається, що ці обмеження узагальнені, параметричні і незалежні. У даній роботі були використані і охарактеризовані 2 типи породжуючих систем, які можуть бути як нейтрального, так і спрямованого типу. Отже, системи з поведінкою можуть ділитися на типи:

а) базові: нейтральні (рівняння 2.32), спрямовані (рівняння 2.39);

б) породжуючі: нейтральні (рівняння 2.35), спрямовані (рівняння 2.41).

Переваги таких систем полягають в можливості використання масок будь-якого виду, які визначені методологічним апаратом. Крім того, в таких системах відсутня власна надлишковість, яка могла би випливати через накладання в масці поточного і наступного стану.

Згідно з [8-10, 13, 14] існують різні методологічні відмінності для породжуючих систем. Ці відмінності були виділені для систем нижчих епістемологічних рівнів. Найважливішими серед них є:

– відмінність між чіткими і нечіткими каналами спостереження, які визначають чіткі або нечіткі дані і вимагають застосування різних методів обробки даних;

– відмінність між нейтральними і спрямованими системами;

– упорядкованість параметричної множини;

– упорядкованість усіх множин станів, що значно спрощує процедури для породжуючих систем, а також зменшує час обробки не повністю визначених наборів даних.

Також в [8-10, 13] наводяться методологічні відмінності породжуючих систем, застосовні лише для систем більш високого епістемологічного рівня:

– детермінованість і недетермінованість систем;

– для недетермінованих систем розрізняють типи нечітких мір, які характеризують параметрично інваріантне обмеження на розглянуті вибіркові змінні;

– за заданою маскою можна виділити породжуючі системи без пам'яті та системи, які залежать від минулого.

Для породжуючих систем існує проблема обчислювальної складності, яка пов'язана з великою кількістю змістовних підмасок, які утворюються в результаті розбиття маски M на підмаски M_g і $M_{\bar{g}}$ відповідно до (2.33). Одним із підходів для вирішення цієї проблеми є спрощення складності системи даних, якому і присвячено 3 розділ роботи.

3.1 Підходи до розв'язання проблеми обчислювальної складності. Виділення критеріїв спрощення.

Для розгляду проблеми складності, спочатку, згідно Дж. Кліру [10], наведемо у таблиці 3.1 кількість $N(n, \Delta M)$ змістовних підмасок для систем з поведінкою із маскою M, які визначаються в залежності від глибини маски ΔM та кількості змінних n початкової системи.

Очевидно, що в деяких випадках, навіть якщо користувач має потужну паралельну систему обчислювальної техніки, кількість змістовних підмасок виявляється все одно занадто великою, щоб піддатися обчислювальній обробці та аналізу за відносно короткий проміжок часу. Саме через такі

Таблиця 3.1 – Кількість змістовних підмасок $N(n, \Delta M)$

n \ ΔM	1	2	3	4	5	6	7
1	1	2	4	8	16	32	64
2	1	8	40	176	736	3008	12160
3	1	26	316	3032	26416	220256	$1{,}8 \cdot 10^6$
4	1	80	2320	48224	872896	$1{,}5 \cdot 10^7$	$\sim 2{,}4 \cdot 10^8$
5	1	242	16564	742568	$\sim 2{,}7 \cdot 10^7$	$\sim 9{,}6 \cdot 10^8$	$\sim 10^{10}$
6	1	728	116920	$\sim 1{,}1 \cdot 10^7$	$\sim 8{,}8 \cdot 10^8$	$\sim 10^{11}$	$\sim 10^{12}$
7	1	2186	821356	$\sim 1{,}7 \cdot 10^8$	$\sim 10^{10}$	$\sim 10^{12}$	$\sim 10^{15}$
8	1	6560	$\sim 5{,}8 \cdot 10^6$	$\sim 10^9$	$\sim 10^{12}$	$\sim 10^{14}$	$\sim 10^{17}$
9	1	19682	$\sim 4 \cdot 10^7$	$\sim 10^{11}$	$\sim 10^{13}$	$\sim 10^{16}$	$\sim 10^{19}$
10	1	59048	$\sim 2{,}8 \cdot 10^8$	$\sim 10^{12}$	$\sim 10^{15}$	$\sim 10^{18}$	$\sim 10^{21}$

Продовження таблиці 3.1

n \ ΔM	8	9	10
1	128	256	512
2	48896	196096	785408
3	$\sim 1{,}5 \cdot 10^7$	$\sim 1{,}2 \cdot 10^8$	$\sim 10^9$
4	$\sim 10^9$	$\sim 10^{11}$	$\sim 10^{12}$
5	$\sim 10^{12}$	$\sim 10^{13}$	$\sim 10^{15}$
6	$\sim 10^{14}$	$\sim 10^{16}$	$\sim 10^{18}$
7	$\sim 10^{17}$	$\sim 10^{19}$	$\sim 10^{21}$
8	$\sim 10^{19}$	$\sim 10^{21}$	$\sim 10^{24}$
9	$\sim 10^{21}$	$\sim 10^{24}$	$\sim 10^{27}$
10	$\sim 10^{24}$	$\sim 10^{27}$	$\sim 10^{30}$

складнощі при обчисленні, часто, на певному етапі обробки початкової системи даних, виникає необхідність спрощення породжуючих систем, які відповідають початковій. Також можливий випадок, коли особисто користувач потребує спрощення системи через складність її аналізу і розуміння. Крім того, спрощення початкової системи даних корисні за методологічними міркуваннями.

Розрізняють два основні методи спрощення систем даних з подальшим спрощенням відповідних їм породжуючих систем [8-10]:

а) вилучення деяких змінних із відповідної подібної системи;

б) визначення всіх можливих класів еквівалентності станів змінних.

3.1.1 Спрощення першого роду

Розглянемо перший із основних методів спрощення систем даних, який базується на вилучені деяких вибіркових змінних із відповідної подібної системи.

Нехай множина змінних породжуючої системи V складається з n змінних і будь-яка підмножина V (крім порожньої), представляє змістовне спрощення, яке називають спрощенням 1 роду. Для цієї системи буде існувати $2^n - 2$ нетривіальних спрощень 1 роду [8-10], які можуть бути частково впорядкованими. Якщо включити початкову множину V і порожню множину, то в такому випадку множина спрощень з частковим упорядкуванням утворює решітку. Зазвичай цю решітку називають решіткою змінних або V-решіткою, позначають $\mathcal{L}_V$ і описують у вигляді

$$\mathcal{L}_V = (\mathcal{P}(V), \subseteq)$$

або

$$\mathcal{L}_V = (\mathcal{P}(V),\cap,\cup).$$

У такому випадку, для множини V можна задати функцію поведінки для заданої системи з поведінкою зі змінними, – f_B. У разі спрощення цієї системи за допомогою скорочення множини V до підмножини V' спрощену процедуру поведінки f_B визначають в [13, 14] за допомогою проекції

$$f'_B(\alpha) = [f_B \downarrow V'](\alpha), \tag{3.1}$$

яка визначається за допомогою рівняння:

$$[f_B \downarrow Z](x) = \sum_{c \succ x} f_B(c), \tag{3.2}$$

де f_B – розподіл ймовірностей.

У якості прикладу використання спрощення першого роду розглянемо завдання спрощення ймовірнісної функції поведінки, за допомогою виключення змінної v_3 із системи даних, заданої емпіричними даними, які наведено у таблиці 3.2. Крім того, у таблиці наведемо ймовірності відповідних станів $f_B(c)$.

Таблиця 3.2 – Система даних, задана емпіричними даними та відповідною функцією розподілу ймовірностей

	v_1	v_2	v_3	$f_B(c)$
$c =$	0	0	0	0.20
	0	1	1	0.05
	0	2	2	0.04
	1	1	1	0.09

Продовження таблиці 3.2

	v_1	v_2	v_3	$f_B(c)$
$c =$	1	1	2	0.06
	2	1	0	0.12
	2	1	1	0.10
	2	1	2	0.07
	2	2	0	0.15
	2	2	1	0.04
	2	2	2	0.08

Для спрощення системи даних, на основі першого роду, необхідно використати рівняння (3.2), для обчислення проекції $f_B \downarrow \{v_1, v_2\}$. Згодом ідентифікуються блоки станів, нерозпізнаних після застосування функції спрощення. Після цього ймовірності всіх станів у кожній групі підсумовуються і в результаті отримуємо спрощену функцію поведінки, яка продемонстрована на таблиці 3.3.

Таблиця 3.3 – Результат спрощення системи даних на основі спрощення 1 роду

	v_1	v_2	$f_B \downarrow \{v_1, v_2\}(x)$
$x =$	0	0	0.20
	0	1	0.05
	0	2	0.04
	1	1	0.15
	2	1	0.29
	2	2	0.27

3.1.2 Спрощення другого роду

У випадку, коли до задачі застосовують спрощення другого роду, відбувається зменшення числа станів, що виділяються для окремих змінних. Їх можна описати [10, 13, 14] за допомогою визначення функції

$$\sigma_{i,j}: V_i \to V_i', \tag{3.3}$$

де V_i – початкова множина станів (яка складається зі змінних v_i); V_i' – скорочена множина станів змінної v_i; $\sigma_{i,j}(x)$ – новий стан, заданий початковому стану x, а i, j – змінні ідентифікатори, які необхідні для розрізнення функцій в (3.3), застосованих до множини станів однієї і тієї ж змінної. У разі, коли $\sigma_{i,j}(x) = \sigma_{i,j}(y)$ стани x і y з V_i при спрощенні є ідентичними. Функція (3.3) повинна бути гомоморфною, щодо всіх математичних властивостей початкової множини V_i, які вважаються суттєвими для розглянутої задачі. Функцію (3.3), яка є гомоморфізмом, називають спрощуючою функцією.

3.2 Визначення та розрахунок кількості вирішуючих форм при спрощенні другого роду

Будь-яка функція спрощення індукує розбиття на множині V_i. Це розбиття зазвичай позначають $V_i/\sigma_{i,j}$. Будь-яке таке розбиття складається з груп станів V_i, які неможливо розрізнити при даному спрощенні. Таке розбиття, яке зберігає суттєві властивості V_i, називають вирішуючою формою [8-10, 13, 14].

Якщо вирішуючі форми визначені на множині станів V_i, то їх можна впорядкувати за допомогою відношення уточнення, яке має бути визначено на розбитті даної множини. Таке відношення є відношенням часткового порядку і утворює решітку [10]. Нехай K і P – задані розбиття, які визначені на одній і тій же множині. У такому випадку, як відомо, K є уточненим розбиттям P тоді і тільки тоді, коли для будь-якої групи $k \in K$ існує група $p \in P$, при умові, що $k \subseteq p$. У разі, якщо K уточнює розбиття P, то P називають укрупненим розбиттям K. Через $\mathcal{L}_{V_i}$ позначають решітку вирішуючих форм, заданих на множині станів V_i. Тоді $\mathcal{L}_{V_i}$ буде вирішуючою решіткою і її визначають у вигляді [10]:

$$\mathcal{L}_{V_i} = \left(\left\{V_i/\sigma_{i,j}\right\}, \leq\right)$$

або

$$\mathcal{L}_{V_i} = \left(\left\{V_i/\sigma_{i,j}\right\}, \times, +\right),$$

де $\times$ – добуток розбиття; $+$ – сума розбиття.

Якщо початкова множина станів не має математичних властивостей, які необхідні для збереження, то вирішуюча форма може приймати будь-яке розбиття. Тоді вирішуюча решітка буде містити всі розбиття, визначені на цій множині станів. Нехай множина станів має k станів, тоді можна визначити за допомогою формули число вирішуючих форм в решітці, згідно робіт [8-10, 13, 14]:

$$\Lambda_k = \sum_{i=0}^{k-1} \binom{k-1}{i} \Lambda_k, \Lambda_0 = 1. \tag{3.4}$$

Розрахунки кількості вирішуючих форм для невпорядкованих множин, для числа станів k, $k \in [1; 10]$, $k \in N$ представлені у таблиці 3.4.

Таблиця 3.4 – Розрахунок кількості вирішуючих форм для невпорядкованих множин

k	2	3	4	5	6	7	8	9	10
Λ_k	2	5	15	52	203	877	4140	21147	115975

Очевидно, що число вирішуючих форм досить велике, навіть для обмежено малого числа станів.

Так як, найменша уточнена вирішуюча форма сенсу не має, а при найбільшому уточненні спрощення не відбувається, то число осмислених спрощень дорівнює $\Lambda_k - 2$ [10].

Якщо є повністю впорядкована множина станів і при цьому потрібно зберегти упорядкованість при спрощенні, то кількість вирішуючих форм набагато менше числа, визначеного формулою (3.4).

Розглянемо повністю впорядковану множина станів. Нехай $x_1, x_2, \dots, x_k$ – стани і $x_k < x_{k+1} (k = 1, \dots, k-1)$. Тоді для $\forall k \le m-1$ x_k і x_{k+1} можуть або не можуть задавати групу. Тільки ці рішення задають певне розбиття. Таким чином, для m станів приймається $m-1$ бінарне рішення. У такому випадку, для множин станів, які повністю впорядковані число осмислених спрощень, відповідно до [10], записують у вигляді:

$$\Lambda_m = 2^{m-1}. \tag{3.5}$$

Не важко помітити, що така решітка для m станів ізоморфна булевій решітці для впорядкування підмножин будь-якої множини з $m-1$ елемента.

Розрахунки вирішуючих форм для впорядкованих множин, для числа станів $m, m \in [1; 10], m \in N$ показані у таблиці 3.5.

При порівнянні табл. 3.4 і табл. 3.5 очевидно, що число вирішуючих форм упорядкованих множин станів на багато менше, ніж невпорядкованих множин.

Таблиця 3.5 – Розрахунок кількості вирішуючих форм для впорядкованих множин

m	2	3	4	5	6	7	8	9	10
Λ_m	2	4	8	16	32	64	128	256	512

Аналогічно невпорядкованим множинам, число змістовних спрощень впорядкованих множин дорівнює $\Lambda_m - 2$.

Розглянемо приклад змістовних спрощень другого роду для повністю впорядкованої множини станів. Нехай задані загальні етапи освіти середньостатистичної людини. Тоді змінна, яка їх описує може приймати стани:

– a – початкова загальна освіта;
– b – основна загальна освіта;
– c – повна загальна освіта;
– d – вища освіта.

Тоді, очевидне впорядкування цих елементів можна записати за допомогою нерівності $a < b < c < d$. Кількість змістовних вирішуючих форм, згідно (3.5), буде $\Lambda_4 = 2^{4-1} = 8$. Число змістовних спрощень буде $\Lambda_4 = 2^{4-1} - 2 = 8 - 2 = 6$. Решітку вирішуючих форм можна описати за допомогою діаграми Хассе, зображеної на рис. 3.1. Утворені групи в окремих вирішуючих формах можна назвати як:

– $\overline{ab}$ – освіта, не вища основної загальної;
– $\overline{bc}$ – основна або повна загальна освіта;
– $\overline{cd}$ – повна загальна або вища освіта;
– $\overline{abc}$ – будь-яка освіта, крім вищої;
– $\overline{bcd}$ – освіта, вища від початкової загальної.

Для спрощення заданої системи слід рухатися в напрямку, протилежному напрямку стрілок.

$$
\begin{tikzcd}
 & \bar{a}\bar{b}\bar{c}\bar{d} & \\
\overline{ab}\;\bar{c}\;\bar{d} \arrow[ur] & \bar{a}\;\overline{bc}\;\bar{d} \arrow[u] & \bar{a}\;\bar{b}\;\overline{cd} \arrow[ul] \\
\bar{a}\;\overline{bc}\;\bar{d} \arrow[ur] & \overline{ab}\;\overline{cd} \arrow[ul] \arrow[ur] & \bar{a}\;\overline{bcd} \arrow[ul] \\
 & \overline{abcd} \arrow[ul] \arrow[u] \arrow[ur] &
\end{tikzcd}
$$

Рисунок 3.1 – Решітка вирішуючих форм для повністю впорядкованої множини

У якості ще одного прикладу використання спрощення другого роду розглянемо спрощення системи даних із відповідними станами розподілу ймовірнісної функції $f_B(c)$, яка задана у таблиці 3.2. Застосуємо визначену процедуру спрощення другого роду до множини станів $V_1 = V_2$ для змінних v_1 і v_2, що продемонстровано у таблиці 3.6.

Таблиця 3.6 – Спрощення станів V_1 і V_2 відповідно до процедури спрощення другого роду

$V_1 = V_2$	$V_1' = V_2'$
0	1
1	1
2	2

Результат спрощення таблиці 3.2, без ідентифікування нерозпізнаних блоків станів, продемонстровано у таблиці 3.7.

Таблиця 3.7 – Результат спрощення другого роду без ідентифікування нерозпізнаних блоків станів

	v_1	v_2	v_3	$f_B(c)$
$c =$	1	1	0	0.20
	1	1	1	0.05
	1	2	2	0.04
	1	1	1	0.09
	1	1	2	0.06
	2	1	0	0.12
	2	1	1	0.10
	2	1	2	0.07
	2	2	0	0.15
	2	2	1	0.04
	2	2	2	0.08

Тепер виконаємо ідентифікування блоки станів, нерозпізнаних після застосування функції спрощення та підсумуємо ймовірності всіх станів у кожній групі. В результаті отримаємо спрощену функцію поведінки, яка продемонстрована на табл. 3.8.

Таблиця 3.8 – Результат спрощення системи даних за допомогою спрощення другого роду

	v_1	v_2	v_3	$f_B(c)$
$c =$	1	1	0	0.20
	1	2	2	0.04
	1	1	1	0.14

Продовження таблиці 3.8

	v_1	v_2	v_3	$f_B(c)$
$c =$	1	1	2	0.06
	2	1	0	0.12
	2	1	1	0.10
	2	1	2	0.07
	2	2	0	0.15
	2	2	1	0.04
	2	2	2	0.08

3.3 Формування об'єднаної вирішуючої решітки при спрощенні другого роду

Будь-який елемент V-решітки задає певний вибір змінних з початкової системи, яка представляє данні. Для будь-якої такої обраної змінної її вирішуюча решітка складається з усіх можливих вирішуючих форм. У разі вибору кількох змінних їх вирішуючі форми можна об'єднати, при чому таке об'єднання утворює, як відомо [8-10], об'єднану вирішуючу решітку. Математично вона представляє собою добуток окремих вирішуючих решіток і визначається наступним чином: нехай X_1, X_2, … , X_n – множини елементів окремих вирішуючих решіток обраних змінних, а X – множина елементів відповідної вирішуючої решітки. У такому випадку

$$X = X_1 \times X_2 \times \ldots \times X_n,$$

і для двох заданих n-ок

$$(x_1, x_2, \dots, x_n)(y_1, y_2, \dots, y_n) \in \mathrm{X},$$

можна визначити, що

$$(x_1, x_2, \dots, x_n) \leq (y_1, y_2, \dots, y_n) \in \mathrm{X}$$

виконується тоді і тільки тоді, коли $x_j \leq y_j$ для усіх вирішуючих решіток, при $j = \overline{1, n}$. Загальна кількість елементів об'єднаної вирішуючої решітки, згідно [10], знаходять за формулою:

$$|\mathrm{X}| = \prod_{j=1}^{n} |X_j|,$$

але слід зазначити, що тільки деякі з них є змістовними спрощеннями. Наприклад, якщо в комбінацію входить найменша уточнена вирішуюча форма однієї з вирішуючих решіток, то вона сенсу немає. Тоді, загальне число елементів об'єднаної решітки, що представляють змістовні спрощення, визначають за формулою:

$$|\mathrm{X}_S| = \prod_{j=1}^{n} \left(|X_j| - 1\right) - 1. \tag{3.6}$$

Формулу (3.6) можна спростити, у разі якщо всі окремі решітки ідентичні, при чому кожна з них складається з Λ_m роздільних форм:

$$|\mathrm{X}_S| = \prod_{j=1}^{n} (\Lambda_m - 1)^n - 1. \tag{3.7}$$

Більш того, коли множина, на якій побудовані вирішуючі решітки, повністю упорядкована, то

$$|X_S| = (2^{m-1} - 1)^n - 1. \tag{3.8}$$

3.4 Кількість осмислених спрощень другого роду

Розглянемо випадок, коли початкова система, яку необхідно спростити, буде містити n змінних v_1, v_2, …,v_n, яким у відповідність ставлять множини X_1, X_2,…,X_n вирішуючих форм. У такому разі, загальна кількість осмислених спрощень (з урахуванням виключення змінних) $N(X_1, X_2, \dots, X_n)$ можна знайти за формулою [8-10, 13, 14]:

$$N(X_1, X_2, \dots, X_n) = \prod_{j=1}^{n} |X_j| - 2. \tag{3.9}$$

Формула (3.9) може бути спрощена, якщо усі змінні мають одну і ту ж саму множину, тобто $X_1 = X_2 =\ldots= X_n =$ X. Тоді

$$N(X_1, X_2, \dots, X_n) = |X^n| - 2. \tag{3.10}$$

Якщо ж, крім цього, множини станів змінних повністю впорядковані і кожна складається з m станів, то:

$$N(X_1, X_2, \dots, X_n) = 2^{n(m-1)} - 2. \tag{3.11}$$

У разі якщо є уявлення про те, які спрощення початкової системи з поведінкою доцільно виконати, то функція поведінки спрощеної системи визначається через функцію поведінки початкової системи. Якщо відкидається якась кількість змінних, то перш за все обчислюють проекцію (3.1). При необхідності подальшого спрощення за допомогою укрупнення вирішуючих форм деяких з решти змінних, спочатку повинні бути виконані зміни, аналогічні проекції. Укрупнення вирішуючих форм дає набір груп станів, нерозпізнаних для таких форм, при цьому кожному блоку ставлять у відповідність один стан. Ймовірність такого стану знаходять за допомогою суми ймовірностей всіх станів, які входять в групу.

3.5 Загальна постановка питання спрощення для системних задач

Спрощення систем – важливий тип системних задач. Його можна охарактеризувати, як процес зменшення знайденої якимось чином складності системи заданого епістемологічного рівня, який при цьому зберігає максимально великий обсяг інформації, що знаходиться в системі. Існує загальний опис всіх завдань цього класу [8-10, 13, 14]:

а) задана конкретна система x певного епістемологічного рівня;

б) відома множина систем того ж типу Y_x, які розглядаються в якості змістовних спрощень x;

в) накладається набір вимог Q, які розглядаються як властивості систем з множини Y_x, що визначають підмножини Y_Q множини Y_x, такої, що будь-яка система з Y_Q задовольняє усім вимогам, заданим із Q.

Введемо вимогу простоти та вимогу чіткості для множини систем Y_x, які розглядаються, як змістовні спрощення x. Нехай X – система з поведінкою, Y_x – множина усіх змістовних спрощень X, які грунтуються на тій же множині змінних, що і X, а також для множини Q задано дві вимоги:

– системи з Y_Q мають бути максимально простими;

– ступінь породжуючої нечіткості, з Y_Q була максимально малою.

У такому випадку, вимогу 1 називають вимогою простоти, а вимогу 2 – вимогою чіткості [8-10, 13, 14].

Для конкретизації вимоги простоти для систем з поведінкою задають певну міру складності. В ідеалі, обчислювальний алгоритм повинен передбачати можливість користувачеві самостійно задати міру складності, у тому числі і надати варіант за замовчуванням. В якості міри за замовчуванням, згідно [10], можна використовувати число станів ненульової ймовірності або можливості. У такому випадку використовують позначення:

$$|f_B| = |\{c|f_B(c) > 0\}| \tag{3.12}$$

для даної міри складності системи з поведінкою F_B, де f_B – функція поведінки системи. Вимога чіткості, в свою чергу, може виражатися через ймовірнісну нечіткість $H(G|\bar{G})$, яка визначається рівнянням (2.47).

Відповідно до [8-10, 13, 14], можна визначити множину рішень $Y_Q \subset Y_x$, таких що

$$Y_Q = \{\, {}^jF_B \in Y_x | \left(\forall\, {}^kF_B \in Y_x\right)\left({}^kF_B \ {}^*\!\!\leq\ {}^jF_B \Rightarrow {}^jF_B \leq {}^kF_B\right)\}, \tag{3.13}$$

де ${}^jF_B$, ${}^kF_B \in Y_x$ – узагальнене рефлексивне і транзитивне відношення на Y_x, ${}^*\!\!\leq$ – узагальнений порядок вподобань.

Після процедури визначення множини Y_Q допустимих спрощень заданої системи, користувач може використовувати будь-які системи з Y_Q, як спрощення початкової системи. Крім того, у нього є можливість вибрати найбільш потрібну систему або він може скористатися додатковими критеріями для скорочення цієї множини.

Отже, в третьому розділі виділено два критерії спрощення, які дозволяють розв'язати проблему обчислювальної складності для систем даних, при незначних втратах інформації. Так, спрощення першого роду базується на вилучені деяких вибіркових змінних із відповідної подібної системи, а спрощення другого роду на визначенні всіх можливих класів еквівалентності станів змінних. Для спрощення за другим критерієм було введено поняття вирішуючої форми, їх кількості для системи даних та методи формування об’єднаної вирішуючої решітки. Було охарактеризовано загальну постановку питання спрощення для системних задач. Крім того, у 3 розділі узагальнено поняття породжуючих систем та вказано основні методологічні відмінності для систем різних епістемологічних рівнів.

Наведені теоретичні методики будуть застосовуватись при автоматизації та алгоритмізації спрощення системи на основі емпіричних даних у 4 розділі роботи при формуванні програмних продуктів.

4 АВТОМАТИЗАЦІЯ ТА АЛГОРИТМІЗАЦІЯ РОЗВ’ЯЗКУ ПРОБЛЕМИ ОБЧИСЛЮВАЛЬНОЇ СКЛАДНОСТІ

У даному розділі уділено увагу побудові та дослідженню математичної моделі складної системи, а саме її спрощенню за допомогою 2 критеріїв. У якості початкової системи виступає система тренування пілотів на реактивних літах. В основі виконання вказаного завдання покладено методику, запропоновану Дж. Кліром [10] теорії складних систем. Ціллю дослідження процесу виділеної системи на об’єкті є створення програмних продуктів, які дозволяють автоматизувати процес побудови математичної моделі об’єкта, визначити ступінь недетермінованості для отриманих систем, розглянути можливі варіанти розв’язку проблеми обчислювальної складності, на основі спрощення за допомогою 2 критеріїв, які визначені у 3 розділі роботи. Процес автоматизування реалізований на базі мови програмування Pascal. Вибір цієї мови програмування був продиктований особистими вподобаннями автора. Узагалі, мова програмування Pascal вважається структурованою і легкою для читання та аналізування, що дозволяє конвертувати програмні продукти з мови Pascal на будь-які інші мови програмування з невеликими витратами часу. Така задача не була поставлена у ході виконання роботи, але її можна реалізувати у майбутньому.

Для більш зручного представлення процесу побудови математичної моделі за емпіричними даними з її подальшим дослідженням, доцільно розділити його на 2 етапи.

4.1 I етап. Формування математичної моделі початкової системи на об'єкті. Використання ступеня недетермінованості

4.1.1 Побудова початкової системи та системи даних

Вхідними даними для проведення I етапу дослідження є уточнення об'єкту дослідження та формулювання цілі дослідження із визначенням обмежень, які стосуються часових, правових, фінансових характеристик процесу дослідження.

Спочатку необхідно побудувати початкову систему, яка складається з 5 примітивних систем. Перша примітивна система – система на об'єкті O, яка має чотири властивості a_i, $i = \overline{1, n}$, що визначають відповідні характеристики реактивного літака; базовою властивістю є b_j, $j = 1$. При цьому, за базову властивість береться час. Тоді систему на об'єкті можна представити у вигляді чорної скрині:

a_1 – швидкість $\rightarrow$
a_2 – висота $\rightarrow$ $\boxed{\text{ОД}}$ $\rightarrow$ a_4 – відстань
a_3 – курс $\rightarrow$

Прояви властивостей початкової системи O, відповідно до дослідження, набувають вигляд:

$A_1 = \{150, 151, \dots, 270, \dots\}$ вузлів, $A_2 = \{1700, 1701, \dots, 22800, \dots\}$ футів, $A_3 = \{0, 1, \dots, 359, 360\}$ градусів, $A_4 = \{0, 1, \dots, 21, \dots\}$ морських миль.

Так як, за базову властивість приймається час, то можна ввести наступні позначення: b_1 – час, $B_1 = \{1, 2, \dots, 163\}$ секунди.

Наступним кроком є побудова другої примітивної системи $\dot{I}$ – конкретно представленої системи, елементами якої виступають: $\dot{v}_i$, $\dot{V}_i$, $\dot{\omega}_j$, $\dot{W}_j$.

Так, відповідно до 2.2 і 2.3 введемо конкретні змінні і параметри за допомогою каналів спостереження o_i и ω_1 відповідно:

$$o_1(150) = E, o_1(151) = E, \dots , o_1(169) = E, o_1(170) = D, \dots , o_1(219) = D, o_1(220) = C, o_1(221) = C, \dots , o_1(249) = C, o_1(250) = B, o_1(251) = B, \dots , o_1(269) = B, o_1(270) = A, o_1(271) = A, \dots , o_1(280) = A.$$

Тоді, конкретизована змінна $\dot{v}_1$ та множина її станів $\dot{V}_1$, приймуть вигляд: $\dot{v}_1$ – швидкість, $\dot{V}_1 = \{E, D, C, B, A\}$.

Оскільки

$$o_2(1700) = \text{низько}, o_2(1701) = \text{низько}, \dots , o_2(1839) = \text{низько}, o_2(1840) = \text{нижче серед.}, o_2(1841) = \text{нижче серед.}, \dots , o_2(2399) = \text{нижче серед.}, o_2(2400) = \text{серед.}, o_2(2401) = \text{серед.}, \dots , o_2(2699) = \text{серед.}, o_2(2700) = \text{вище серед.}, o_2(2701) = \text{вище серед.}, \dots , o_2(4899) = \text{вище серед.}, o_2(4900) = \text{високо}, o_2(4901) = \text{високо}, \dots , o_2(6899) = \text{високо}, o_2(6900) = \text{дуже високо}, o_2(6901) = \text{дуже високо}, \dots , o_2(22799) = \text{дуже високо}, o_2(22800) = \text{занадто високо}, o_2(22801) = \text{занадто високо}, \dots , o_2(24000) = \text{занадто високо},$$

тоді $\dot{v}_2$ – h, $\dot{V}_2 = \{\text{низько}, \text{нижче серед.}, \text{серед.}, \text{вище серед.}, \text{високо}, \text{дуже високо}, \text{занадто високо}\}$.

Введення третьої змінної виконується за допомогою функції o_3:

$$o_3(0) = \text{I}, o_3(1) = \text{I}, \dots , o_3(92) = \text{I}, o_3(93) = \text{II}, o_3(94) = \text{II}, \dots , o_3(112) = \text{III}, o_3(113) = \text{III}, \dots , o_3(121) = \text{III}, o_3(122) = \text{IV}, o_3(123) = \text{IV}, \dots, o_3(360) = \text{IV}.$$

Тоді $\dot{v}_3$ – курс, $\dot{V}_3 = \{\text{I}, \text{II}, \text{III}, \text{IV}\}$ чверть.

Аналогічно, для введення четвертої змінної використовується функція o_4:

$$o_4(0) = \text{близ.}, o_4(0.1) = \text{близ.}, \dots, o_4(2.5) = \text{близ.}, o_4(2.6) = \text{серед.}, o_4(2.7) = \text{серед.}, \dots, o_4(13.9) = \text{серед.}, o_4(14) = \text{далеко}, o_4(14.1) = \text{далеко}, \dots, o_4(20.9) = \text{далеко}, o_4(21) = \text{дуже далеко}, o_4(21.1) = \text{дуже далеко}, \dots, o_4(25) = \text{дуже далеко}.$$

Канал спостереження ω_1 для бази B_1 набуватиме вигляд:

$$\omega_1(1) = 0001, \omega_1(2) = 0002, \dots, \omega_1(163) = 0163.$$

Конкретизована множина станів $\dot{W}_1$ для $\dot{w}_1$ – час, задається у вигляді:

$$\dot{W}_1 = \{0001, 0002, \dots ,0163\}.$$

Для того, щоб сформувати третю примітивну системи I, спочатку введемо абстрагування змінних і параметрів $\dot{v}_i$ і $\dot{w}_j$ за допомогою функцій e_i^{-1} і ε_1^{-1} відповідно до формул (2.6) та (2.7):

$$e_1^{-1}(E) = 1, e_1^{-1}(D) = 2, e_1^{-1}(C) = 3, e_1^{-1}(B) = 4, e_1^{-1}(A) = 5.$$

У такому разі, узагальнена змінна v_1 – 1 і множину її станів $V_1 = \{1,2,3,4,5\}$.

Введемо змінну v_2 та множину V_2 наступним чином:

$$e_2^{-1}(\text{низько}) = 1, e_2^{-1}(\text{нижче серед.}) = 2, e_2^{-1}(\text{серед.}) = 3, e_2^{-1}(\text{вище серед.}) = 4, e_2^{-1}(\text{високо}) = 5, e_2^{-1}(\text{дуже високо}) = 6, e_2^{-1}(\text{занадто високо}) = 7.$$

Звідси v_2 – 2, $V_2 = \{1,2,3,4,5,6,7\}$.

Для введення змінної v_3 та множини її станів V_3 застосовуємо канал абстрагування із застосуванням функції e_3^{-1} виду:

$$\mathrm{e}_3^{-1}(\mathrm{I}) = 1,\ \mathrm{e}_3^{-1}(\mathrm{II}) = 2,\ \mathrm{e}_3^{-1}(\mathrm{III}) = 3,\ \mathrm{e}_3^{-1}(\mathrm{IV}) = 4.$$

Тому v_3 – 3, $V_3 = \{1,2,3,4\}$.

Аналогічно, для змінної v_4, та множини V_4:

$$e_4^{-1}(\text{близ.}) = 1,\ e_4^{-1}(\text{серед.}) = 2,\ e_4^{-1}(\text{далеко}) = 3,\ e_4^{-1}(\text{дуже далеко}) = 4.$$

Тоді v_4 – 4, $V_4 = \{1,2,3,4\}$.

Для того, щоб отримати наступний компонент абстрактної системи I – абстрактний параметр w та параметричну множину W, необхідно використати функцію виду:

$$\varepsilon_1^{-1}(0001) = 1,\ \varepsilon_1^{-1}(0002) = 2,\ \varepsilon_1^{-1}(0003) = 3,\ \dots,\ \varepsilon_1^{-1}(0163) = 163$$

Тоді, множина станів узагальнюючого параметра – W_1 матиме вигляд:

$$W_1 = \{1,2, \dots 163\}.$$

Отже, відповідно до (2.13) система об'єкта S задається як п'ятірка – $S = \left(O, \dot{I}, I, Q, \mathcal{E}\right)$, яку для більш легшого сприйняття можна звести до таблиці 4.1.

У результаті проведеного експерименту, було отримано емпіричні дані, які представлені в стандартній формі – системі даних D у таблиці 4.2, де a_i, $i = \overline{1, n}$ – властивості, що визначають відповідні характеристики реактивного літака, якими є швидкість, висота над рівне моря, курс і відстань відповідно.

Таблиця 4.1 – Опис досліджуваної системи

		Канали спостереження			
Характеристика		Швидкість	Висота над рівнем моря	Курс	Відстань
Змінна		a_1	a_2	a_3	a_4
Одиниці вимірювання		Вузли	Фути	Градуси відносно магнітної півночі	Морські милі
Інтерпретація стану	1	[150; 170)	[1700–1840)	[0–93)	[0–2.6)
	2	[170; 220)	[1840–2400)	[93–113)	[2.6–14)
	3	[220; 250)	[2400–2700)	[113–122)	[14–21)
	4	[250–270)	[2700–4900)	[122–360)	[21–25)
	5	[270-280)	[4900–6900)		
	6		[6900–22800)		
	7		[22800–24000)		

Таблиця 4.2 – Емпіричні данні для системи на об’єкті

a_1	4 4 4 4 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5
a_2	7 7 7 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6
a_3	2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
a_4	4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 3 3 3 3 3
a_1	5 5 5 5 5 4 4 4 4 4 4 4 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 2 2 2 2 2 2 2 2 2 2 2 2 2
a_2	6 6 6 6 6 6 6 6 6 6 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4
a_3	1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
a_4	3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 2 2 2 2 2 2 2 2 2 2 2
a_1	1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
a_2	4 4 4 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 2 2 2 2 2 2 1 1 1 1 1
a_3	3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
a_4	2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1
a_1	1 1 1 2 2 2 2 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
a_2	1 1 1 2 2 2 2 2 2 3 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4
a_3	3 3 3 3 3 3 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4
a_4	1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2

4.1.2 Побудова математичної моделі системи з поведінкою

На підставі отриманих даних для початкової системи на об'єкті виникає можливість введення системи з поведінкою F_B, відповідно до (2.32), за допомогою маски M з параметром $\rho = \{0{,}1\}$ та визначити функцію поведінки f_B згідно рівняння (2.31). Для автоматизації побудови такої системи, був створений програмний продукт «Elementary system with mask» на мові програмування Pascal. Інтерфейс програми представлений на рис. 4.1.

```
Write the quantity of W:
163
Write the quantity of V:
4
Write the quantity of numbers in mask:
2
Write your mask:
0
1
```

Рисунок 4.1 – Інтерфейс програми

```
S[1,1]=V[1,1]=4;
S[2,1]=V[2,1]=7;
S[3,1]=V[3,1]=2;
S[4,1]=V[4,1]=4;
S[5,1]=V[1,2]=4;
S[6,1]=V[2,2]=7;
S[7,1]=V[3,2]=1;
S[8,1]=V[4,2]=4;
```

Рисунок 4.2 – Вибіркові змінні

Розглянемо більш докладно алгоритм програми «Elementary system with mask». Спочатку програма зчитує кількість конкретизованих множин станів $\dot{V}_i$ та кількість станів узагальнюючого параметра W_1. Згодом задається потужність маски $|M|$ та декілька значень ρ, які визначають правило зрушення $r_j(w)$, відповідно до (2.34). Крім того, при введенні значень ρ

S[1]	S[2]	S[3]	S[4]	S[5]	S[6]	S[7]	S[8]	f[B](c)
4	7	2	4	4	7	1	4	0.00621
4	7	1	4	4	7	1	4	0.00621
4	7	1	4	4	6	1	4	0.00621
4	6	1	4	5	6	1	4	0.00621
5	6	1	4	5	6	1	4	0.19876
5	6	1	4	5	6	1	3	0.00621
5	6	1	3	5	6	1	3	0.05590
5	6	1	3	4	6	1	3	0.00621
4	6	1	3	4	6	1	3	0.02484
4	6	1	3	4	5	1	3	0.00621
4	5	1	3	4	5	1	3	0.00621
4	5	1	3	3	5	2	3	0.00621
3	5	2	3	3	5	2	3	0.03106
3	5	2	3	3	5	3	3	0.00621
3	5	3	3	3	5	3	3	0.04969
3	5	3	3	3	4	3	3	0.00621
3	4	3	3	3	4	3	3	0.00621
3	4	3	3	2	4	3	3	0.00621
2	4	3	3	2	4	3	3	0.00621
2	4	3	3	2	4	3	2	0.00621
2	4	3	2	2	4	3	2	0.06211
2	4	3	2	1	4	3	2	0.00621
1	4	3	2	1	4	3	2	0.01242
1	4	3	2	1	3	3	2	0.00621
1	3	3	2	1	3	3	2	0.16770
1	3	3	2	1	2	3	2	0.00621
1	2	3	2	1	2	3	2	0.03106
1	2	3	2	1	1	3	2	0.00621
1	1	3	2	1	1	3	2	0.01242
1	1	3	2	1	1	3	1	0.00621
1	1	3	1	1	1	3	1	0.02484
1	1	3	1	2	2	3	1	0.00621
2	2	3	1	2	2	3	1	0.02484
2	2	3	1	3	2	4	1	0.00621
3	2	4	1	3	3	4	1	0.00621
3	3	4	1	3	3	4	1	0.01242
3	3	4	1	3	4	4	1	0.00621
3	4	4	1	3	4	4	1	0.01863
3	4	4	1	3	4	4	2	0.00621
3	4	4	2	3	4	4	2	0.12422

Рисунок 4.3 – Математична модель системи з поведінкою

програма може повідомити користувачу, що введене число для маски M є хибним (у випадках, коли користувач не ввів 0 або введені числа не утворюють зростаючої послідовності) і запропонує ввести більш доцільне число ще раз. Після зчитування усіх необхідних даних, а також початкової системи, заданої емпіричними даними у деякому файлі, програма визначає усі вибіркові змінні $s_{k,w}$ відповідно до (2.26) та вносить їх до вказаного файлу. Визначення таких вибіркових змінних при першому застосуванні маски з $\rho = \{0,1\}$, приведено на рис. 4.2. Згодом, програма розбиває початкову систему, відповідно до заданої маски і виконує підрахунок ймовірностей, на основі функції поведінки f_B для початкової системи, яка представлена емпіричними даними на таблиці 4.2. Утворене розбиття з відповідні ймовірностями, які утворюють математичну модель системи з поведінкою представлено на рисунку 4.3.

4.1.3 Побудова математичної моделі направленої системи з поведінкою, що породжує дані

Після побудови системи на об'єкті, яка дозволяє на основі заданої маски визначати функцію поведінки f_B з'являється можливість сформувати початкову направленої систему з поведінкою, що породжує дані, – $\hat{F}_{GB}$, відповідно до формули (2.41), за допомогою визначника входу-виходу $u = (0,0,0,1)$ для маски M з параметром $\rho = \{0,1\}$. Для алгоритмізації побудови наведеної системи було модифіковано програмний продукт «Elementary system with mask» та створено програму «Directed generating system with behavior» на мові програмування Pascal.

Розглянемо поетапно дії, які виконує програма. Спочатку вона зчитує вхідні дані програма з відповідного файлу, після чого накладається маска $\widehat{M}_G$ на початкову систему D відповідно до визначника входу-виходу та правила зрушення (2.24). Після цього, задається множина ідентифікаторів для системи з поведінкою, а згодом для направленої системи з поведінкою, що породжує дані. Також, задаються вибіркові змінні $s_{k,w}$, за допомогою яких визначається функція поведінки $\hat{f}_{GB}$.

Інтерфейс програмного продукту показано на рис. 4.4; множина ідентифікаторів на рис. 4.5; вибіркові змінні для першого зчитування станів змінної системи маскою M на рис. 4.6; вихідна система після накладення маски і визначення розподілу станів вибіркових змінних ймовірностей для функції $\hat{f}_{GB}$ наведено на рис. 4.7.

4.1.4 Формування змістовних підмасок для системи з поведінкою, із визначеною максимальною маскою M

Для алгоритмізація процесу побудови змістовних підмасок для системи з поведінкою, що породжує дані, на підставі емпіричних даних (див. таблицю

```
Write the quantity of W:
163
Write the quantity of V:
4
Write the quantity of numbers in mask:
2
Please, write the values in determinant:
0
0
0
1
Write your mask:
0
1
```

Рисунок 4.4 – Інтерфейс програми

```
lyamba_e(1;0)=1
lyamba_e(2;0)=2
lyamba_e(3;0)=3
lyamba_g_^(4;0)=4
lyamba_e(1;1)=5
lyamba_e(2;1)=6
lyamba_e(3;1)=7
lyamba_g(4;1)=8
```

Рисунок 4.5 – Множина ідентифікаторів

```
S[k,w]_e:
S[1,1]=V[1,1]=4;
S[2,1]=V[2,1]=7;
S[3,1]=V[3,1]=2;
S[k,w]_g_^:
S[4,1]=V[4,1]=4;
S[k,w]_e:
S[5,1]=V[1,2]=4;
S[6,1]=V[2,2]=7;
S[7,1]=V[3,2]=1;
S[k,w]_g:
S[8,1]=V[4,2]=4;
```

Рисунок 4.6 – Вибіркові змінні

4.2) із заданою маскою M і параметром $\rho = \{0{,}1\}$ було розроблено програмний продукт «Content masks for elementary system with mask», на базі програми «Elementary system with mask». Всі отримані змістовні підмаски задовольняють умовам:

– в підмаску входить принаймні один елемент з кожного рядка M-матриці;

– в підмаску повинен бути доданий принаймні один елемент з правилом зрушення.

	S[1]	S[2]	S[3]	S[5]	S[6]	S[7]	f[B](e)
e =	4	7	2	4	7	1	0.00617
	4	7	1	4	7	1	0.00617
	4	7	1	4	6	1	0.00617
	4	6	1	5	6	1	0.00617
	5	6	1	5	6	1	0.25926
	5	6	1	4	6	1	0.00617
	4	6	1	4	6	1	0.02469
	4	6	1	4	5	1	0.00617
	4	5	1	4	5	1	0.00617
	4	5	1	3	5	2	0.00617
	3	5	2	3	5	2	0.03086
	3	5	2	3	5	3	0.00617
	3	5	3	3	5	3	0.04938
	3	5	3	3	4	3	0.00617
	3	4	3	3	4	3	0.00617
	3	4	3	2	4	3	0.00617
	2	4	3	2	4	3	0.07407
	2	4	3	1	4	3	0.00617
	1	4	3	1	4	3	0.01235
	1	4	3	1	3	3	0.00617
	1	3	3	1	3	3	0.16667
	1	3	3	1	2	3	0.00617
	1	2	3	1	2	3	0.03086
	1	2	3	1	1	3	0.00617
	1	1	3	1	1	3	0.04321
	1	1	3	2	2	3	0.00617
	2	2	3	2	2	3	0.02469
	2	2	3	3	2	4	0.00617
	3	2	4	3	3	4	0.00617
	3	3	4	3	3	4	0.01235
	3	3	4	3	4	4	0.00617
	3	4	4	3	4	4	0.14815

	S[4]	f[B](g_^)
g^=	4	0.22840
	3	0.22222
	2	0.43827
	1	0.11111

	S[8]	f[B](g)
g =	4	0.22222
	3	0.22222
	2	0.44444
	1	0.11111

Рисунок 4.7 – Математична модель направленої системи з поведінкою, що породжує дані

Розглянемо реалізацію побудови змістовних підмасок у вигляді блок-схеми (див. рис. 4.8, рис. 4.9), де i – кількість змінних V_i, j – кількість параметрів W_j, e – глибина маски, m_k – елементи маски, $m1_{k,k1}$ – елементи системи даних D, $m2_k$ – частоти C відповідних вибіркових змінних, a_k – елементи непозиційної системи натуральних чисел записані за допомогою 0 та 1 (використан код Грея), $k5$ – лічильник кількості одиниць у a_k, $k7$ – лічильник, який перевіряє виконання вимоги 1, $k8$ – лічильник, який перевіряє виконання вимоги 2, $count$ – лічильник кількості рядків, що не співпадають для відповідної маски, $count1$ – лічильник кількості стовпців, що задовольняють числам Грея, $k2$ – лічильник загальної кількості масок, $k12$ – лічильник, який контролює збіжність елементів, $m3_{k,k1}$ – елементи використаної маски на $m1_{k,k1}$, $bool = \{0,1\}$ (0 – рядки співпали; 1 – рядки не

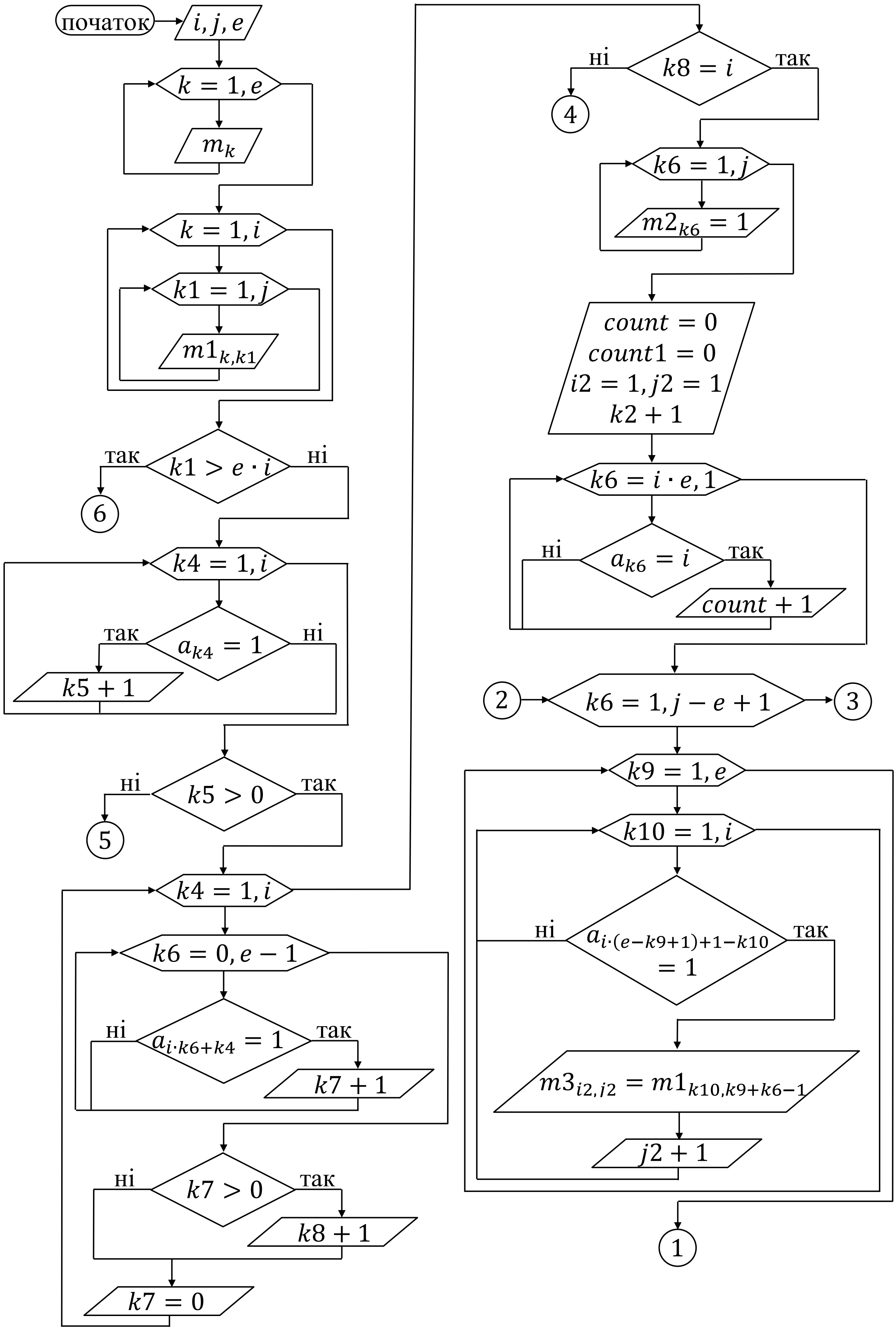

Рисунок 4.8 – Частина №1 блок-схеми алгоритму побудови змістовних підмасок

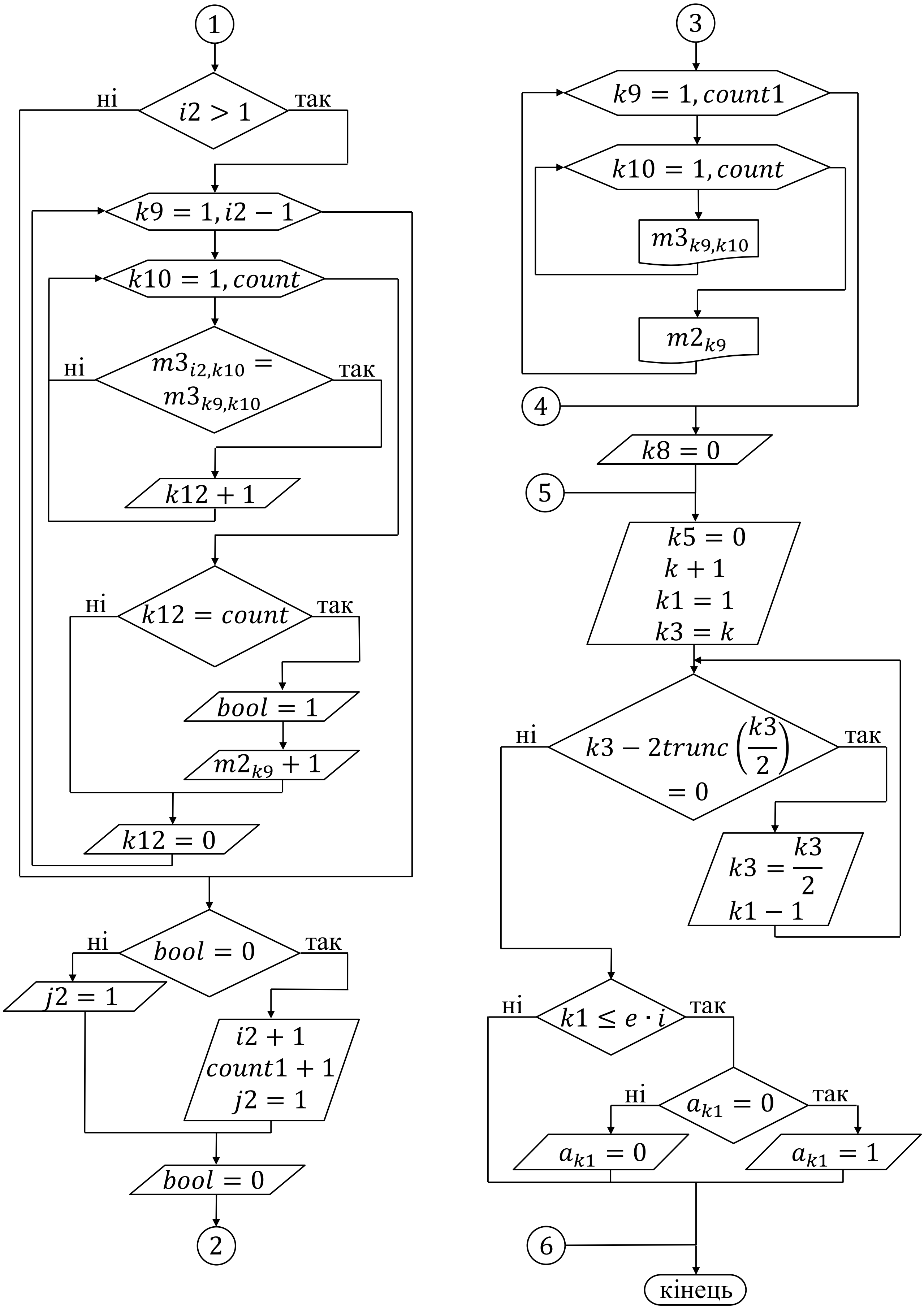

Рисунок 4.9 – Частина №2 блок-схеми алгоритму побудови змістовних підмасок

співпали), $trunc$ – функція, яка округлює дробове число до цілого і відкидає дробову частину.

Під час створення програмного продукту виникла складність, пов’язана з розглядом усіх можливих змістовних підмасок, на основі використання вкладених циклів. Ця проблема була розв’язана за допомогою генерації коду Грея [19], в якому дві сусідні кодові комбінації відрізняються тільки однією цифрою в двійковому розряді. Ці коди дозволяють розглянути усі можливі варіанти змістовних підмасок без використання вкладених циклів.

Інтерфейс програми «Content masks for elementary system with mask» показано на рис. 4.1; одну із змістовних підмасок M, отриману систему F_B і відповідну функцію поведінки f_b представлено на рис. 4.10.

4.1.5 Розрахунок ступеня недетермінованості систем з поведінкою

Для розрахунку ступеня недетермінованості було модифіковано програму «Content masks for elementary system with mask» та створено програмний продукт «Entropies for content masks», який крім генерації змістовних підмасок для систем з поведінкою для маски M і з $\rho = \{0,1\}$, знаходить ентропії згідно з (2.42) і (2.47). Після цього, визначається найменша ентропія для кожного рівня підмасок. Під рівнем розуміється кількість вибіркових змінних, після використання підмаски. Результат генерації записується у вихідний файл, в якому вказуються змістовні підмаски для системи з поведінкою, а також для породжуючої системи з поведінкою та знаходиться відповідні ентропії для кожної з систем. Одна з таких утворених змістовних підмасок з відповідною функцією розподілу та визначеними ентропіями представлена на рис. 4.11.

Після генерації усіх можливих змістовних підмасок програма визначає оптимальні підмаски для кожного рівня складності, на основі вибору

```
 - -
|X| |
 - -
|X| |
 - -
|X| |
 - -
|X| |
 - -
S[5] S[6] S[7] S[8] | f[B](c)
  4    7    1    4  | 0.01235
  4    6    1    4  | 0.00617
  5    6    1    4  | 0.20370
  5    6    1    3  | 0.06173
  4    6    1    3  | 0.03086
  4    5    1    3  | 0.01235
  3    5    2    3  | 0.03704
  3    5    3    3  | 0.05556
  3    4    3    3  | 0.01235
  2    4    3    3  | 0.01235
  2    4    3    2  | 0.06790
  1    4    3    2  | 0.01852
  1    3    3    2  | 0.17284
  1    2    3    2  | 0.03704
  1    1    3    2  | 0.01852
  1    1    3    1  | 0.03086
  2    2    3    1  | 0.03086
  3    2    4    1  | 0.00617
  3    3    4    1  | 0.01852
  3    4    4    1  | 0.02469
  3    4    4    2  | 0.12963
```

Рисунок 4.10 – Приклад змістовної підмаски

```
 - -
|X| |
 - -
|X| |
 - -
|X| |
 - -
|X| |
 - -
S[5] S[6] S[7] S[8] | f[B](c)
  4    7    1    4  | 0.01235
  4    6    1    4  | 0.00617
  5    6    1    4  | 0.20370
  5    6    1    3  | 0.06173
  4    6    1    3  | 0.03086
  4    5    1    3  | 0.01235
  3    5    2    3  | 0.03704
  3    5    3    3  | 0.05556
  3    4    3    3  | 0.01235
  2    4    3    3  | 0.01235
  2    4    3    2  | 0.06790
  1    4    3    2  | 0.01852
  1    3    3    2  | 0.17284
  1    2    3    2  | 0.03704
  1    1    3    2  | 0.01852
  1    1    3    1  | 0.03086
  2    2    3    1  | 0.03086
  3    2    4    1  | 0.00617
  3    3    4    1  | 0.01852
  3    4    4    1  | 0.02469
  3    4    4    2  | 0.12963
H=3.67994
H(G`)=0.00000
H(G|G`)=3.67994
```

Рисунок 4.11 – Результат побудови змістовної підмаски

і визначення ентропії для неї

мінімальної ентропії та виводить отримані ентропії у вихідний файл. Отримані оптимальні математичні моделі відповідно до рівня складності представлено у вигляді графіка на рис. 4.12.

Отриманий графік на рис. 4.12 можна інтерпретувати наступним чином: чим вище знання про систему (об'єкт дослідження), тим нижче ступінь недетермінованості системи, що відповідає теоретичним відомостям, які стосуються інформаційних характеристик систем.

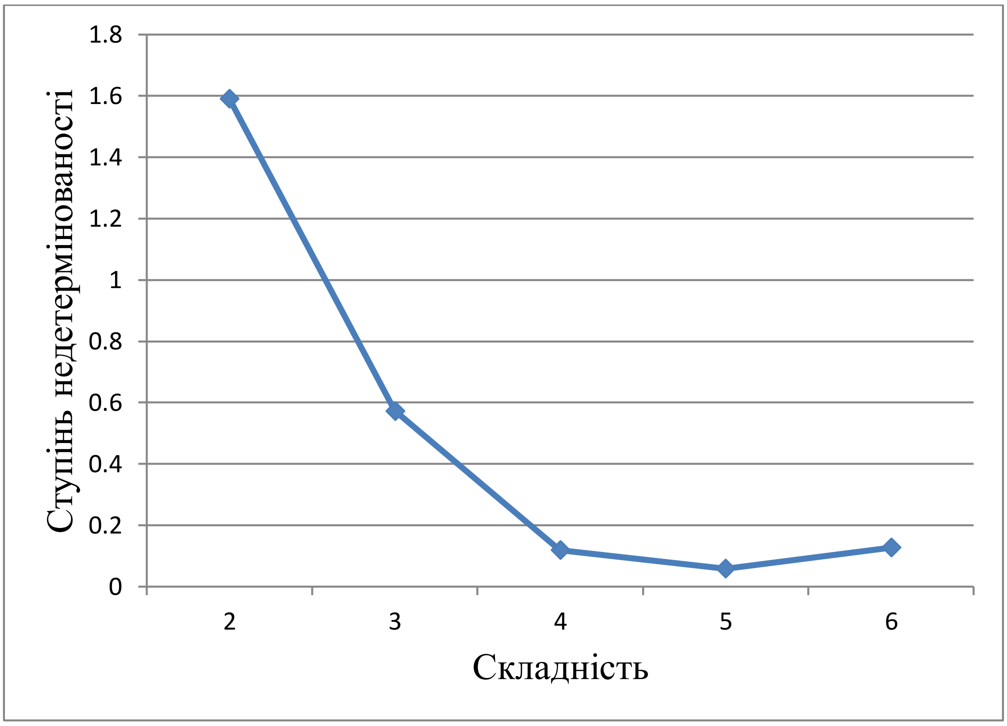

Рисунок 4.12 – Визначення оптимальних систем з поведінкою

4.2 II етап. Спрощення системи за допомогою двох критеріїв

4.2.1 Спрощення першого роду з виділенням оптимальних систем з поведінкою

На основі програми «Entropies for content masks», яка генерує всі змістовні підмаски для систем з поведінкою для маски M з $\rho = \{0,1\}$, а також знаходить ентропії початкової системи та породжуючої системи, при цьому виділяючи ступінь недетермінованості, було створено програмний продукт «Entropies of content masks and 1 simplication». Програма основана на термінологічній базі, яка введена у 3 розділі роботи і пов'язана зі спрощенням обчислювальної складності системи за допомогою спрощення першого роду. Алгоритм програми «Entropies for content masks» залишився незмінним, проте до нього було додано програмний код, відповідно до алгоритму спрощення за першим критерієм, який дозволяє користувачу вибирати самому, який стан V_i він хоче прибрати, а також має вибір за замовчуванням. При цьому, при використанні варіанта за замовчуванням система генерує всі можливі варіанті спрощення за першим родом. Крім того, програма може згенерувати побудову змістовних підмасок без спрощення взагалі, якщо у користувача є бажання порівняти вихідні дані.

Інтерфейс програми представлений на рис. 4.13, де користувач вибрав варіант за замовчуванням, і на рис. 4.14, коли було обрано стан V_1, V_2, а стани V_3, V_4 прибрано.

```
Write the quantity of W:
163
Write the quantity of V:
4
Write the quantity of numbers in mask:
2
Write your mask:
0
1
Do you want to use simplication of 1 degree?
yes
Do you want to remove them by yourself?
no
Okay, than the program will use all removements, which possible
```

Рисунок 4.13 – Інтерфейс програми при варіанті за замовчуванням

При виборі варіанта за замовчуванням програма видає користувачу інформацію про те, що будуть використовуватись усі можливі генерації спрощення першого роду з їх подальшим аналізом. Зважаючи на те, що кількість властивостей початкової системи дорівнює 4, перерахуємо усі можливі варіанти властивостей, які залишаться після використання спрощення першого роду:

```
Write the quantity of V:
4
Write the quantity of numbers in mask:
2
Write your mask:
0
1
Do you want to use simplication of 1 degree?
yes
Do you want to remove them by yourself?
yes
Do you want to remove 1 condition?
no
Do you want to remove 2 condition?
no
Do you want to remove 3 condition?
yes
Do you want to remove 4 condition?
yes
```

Рисунок 4.14 – Інтерфейс програми при спрощенні заданих станів

$$V_1$$

$$V_2$$

$$V_3$$

$$V_4$$

$$V_1V_2$$

$$V_1V_3$$

$$V_1V_4$$

$$V_2V_3$$

$$V_2V_4$$

$$V_3V_4$$

$$V_1V_2V_3$$

$$V_1V_2V_4$$

$$V_1V_3V_4$$

$$V_2V_3V_4$$

Для кожної отриманої підсистеми даних визначається оптимальність на основі ступеня недетермінованості, після чого виводяться ентропії для кожної множини відібраних станів у вихідний файл. На рис. 4.15 зображено графік отриманих оптимальних математичних моделей відповідно до рівня складності для множини станів $V_1 V_2$.

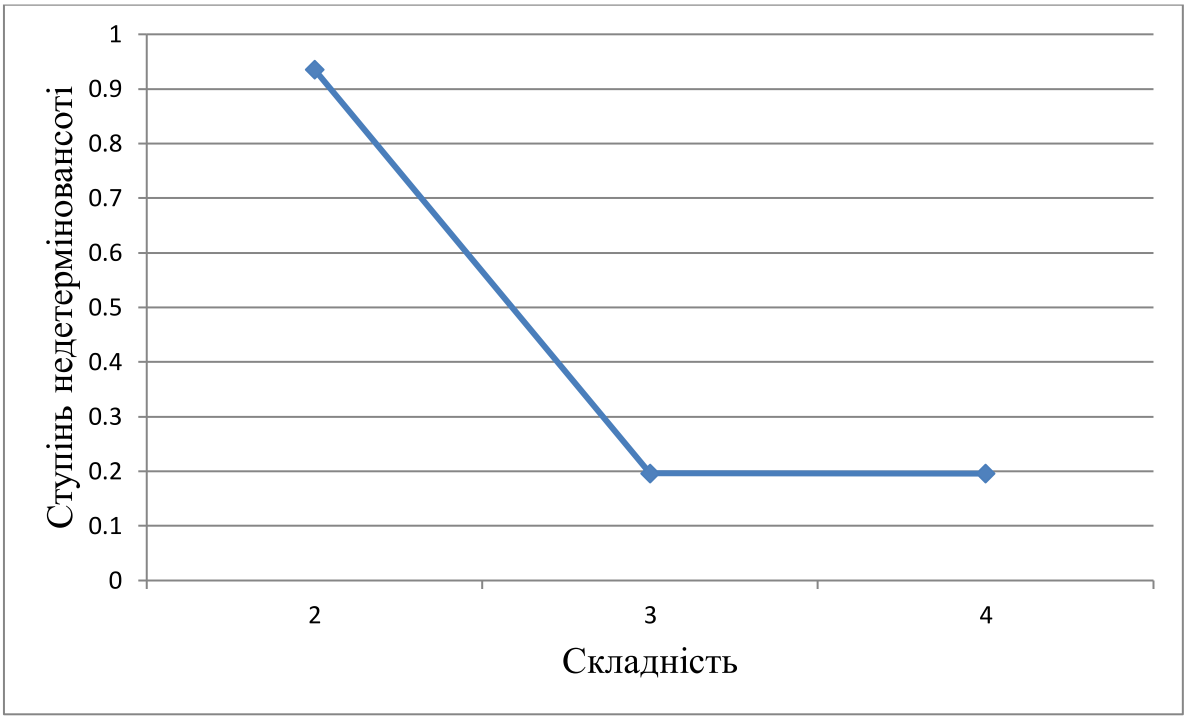

Рисунок 4.15 – Визначення оптимальних систем з поведінкою для множини станів $V_1 V_2$

Отриманий графік на рис. 4.15 уособлює аналогічну ідею, що і в минулому підрозділі роботи, а саме чим більше відомо про об’єкт дослідження, тим нижче ступінь недетермінованості відповідної системи.

4.2.2 Спрощення за другим критеріям з формуванням оптимальних систем з поведінкою

На основі програми «Entropies for content masks», для маски M з $\rho = \{0,1\}$ було створено програмний продукт «Entropies for content masks and 2 simplication», яка дозволяє користувачу використати спрощення 2 роду для системи заданої емпіричними даними. Алгоритм програми дозволяє використовувати користувачу як варіант за замочуванням, так і пропонує задати множину станів V_i яку він бажає спростити. Інтерфейс програми при виборі користувачем варіанту за замовченням представлено на рис. 4.16.

```
Write the quantity of W:
163
Write the quantity of V:
4
Write the quantity of numbers in mask:
2
Write your mask:
0
1
Do you want to use simplication of 2 degree?
yes
Do you want to simpify them by yourself?
no
Okay, than the program will use all simplification, which possible.
```

Рисунок 4.16 – Інтерфейс програми при виборі варіанта за замовчуванням

Для побудови алгоритму програми було використано подвійний код Грея, який дозволяє виділяти вирішуючі решітки при цьому утворюючи підмножини множин станів V_i, які можуть складатися з різної кількості утворених груп. Так, наприклад, розглянемо спрощення другого роду для множини станів V_2 заданої системи. Так як множина V_2 має 7 можливих станів, то кількість вирішуючих решіток, відповідно до таблиці 3.5 буде 64, тоді як кількість осмислених спрощень, – 62, оскільки задані стани V_2 є повністю впорядкованими. Застосуємо визначену процедуру спрощення другого роду до станів множини V_2, які продемонстровані на таблиці 4.3.

Таблиця 4.3 – Спрощення станів множини V_2 відповідно до процедури спрощення другого роду

V_2	V_2'
1	1
2	3
3	3
4	4

Продовження таблиці 4.3

V_2	V_2'
5	4
6	6
7	6

Результат спрощення таблиці 4.2 відповідно до визначеного спрощення представлено на таблиці 4.4.

Таблиця 4.4 – Отримані емпіричні дані після спрощення другого роду

a_1	4 4 4 4 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5
a_2	6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6
a_3	2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
a_4	4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 3 3 3 3 3
a_1	5 5 5 5 5 4 4 4 4 4 4 4 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 2 2 2 2 2 2 2 2 2 2 2 2 2
a_2	6 6 6 6 6 6 6 6 6 6 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4
a_3	1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
a_4	3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 2 2 2 2 2 2 2 2 2 2 2
a_1	1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
a_2	4 4 4 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 1 1 1 1 1
a_3	3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
a_4	2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1
a_1	1 1 1 2 2 2 2 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
a_2	1 1 1 3 3 3 3 3 3 3 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4
a_3	3 3 3 3 3 3 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4
a_4	1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2

Після виконаного спрощення системи даних, програма генерує змістовні підмаски та знаходить ентропії, за допомогою яких згодом виконується визначення ступеня недетермінованості, відповідно до алгоритму програми «Entropies for content masks» для всіх можливих вирішуючих решіток та заносить усі результати до вихідного файлу. На основі запропонованого прикладу, який застосовує спрощення другого роду відповідно таблиці 4.3 можна побудувати графік математичної моделі оптимальних систем, який представлений на рис. 4.17.

Проте, у даній роботі спрощення за другим критерієм було реалізоване тільки на основі вирішуючих форм без можливості використання об'єднаної вирішуючої решітки, що можна реалізувати у майбутньому.

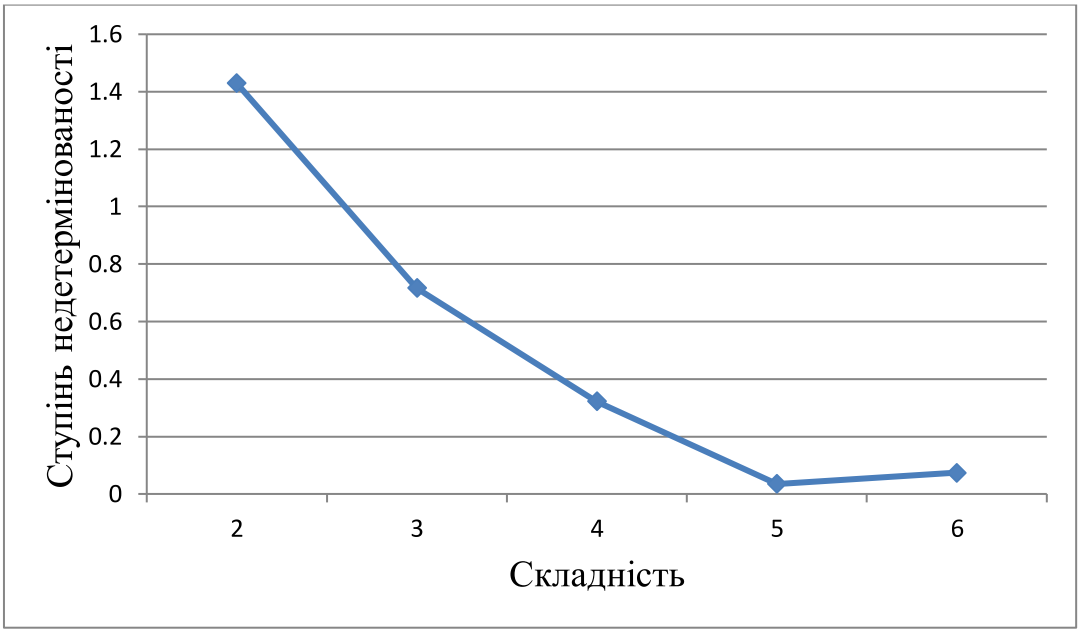

Рисунок 4.17 – Визначення оптимальних систем з поведінкою для множини спрощеної множини станів V_4

Отже, у практичній частині було розглянуто автоматизацію та алгоритмізацію побудови та дослідження математичної моделі складної системи на основі емпіричних даних, а саме спрощення цієї системи на основі двох критеріїв, теоретична база яких розглянута в 3 розділі роботи. На основі поставленої цілі та мети було розроблено програмні продукти, які дозволяють побудувати породжуючу систему з поведінкою, визначити її ентропії, побудувати змістовні підмаски, визначити ступінь недермінованості та виконати спрощення отриманих систем даних, як за замовчуванням, так і з можливістю вибору користувачем відповідних спрощень.

ВИСНОВКИ

Кваліфікаційна робота присвячена алгоритмізації процесу математичного моделювання складних систем та розв’язанні проблеми обчислювальної складності цих систем.

Мета роботи визначалась, як розв’язання проблеми обчислювальної складності, відповідно до якої було побудовано математичну модель складного об’єкту за допомогою методології системного аналізу та виконано спрощення отриманої системи на основі двох критеріїв спрощення.

Для досягненої поставленої мети в роботі здійснено наступне:

– надано характеристику етапам проведення дослідження на основі емпіричних даних;

– формалізовано побудову та дослідження системи на об’єкті, та її подальшого розбиття на системи з поведінкою різних типів;

– визначено види нечітких мір та на основі ймовірнісної міри було введено Шенноновську ентропію, як ступінь недетермінованості для аналізу оптимальності систем різних типів;

– визначено підходи до розв’язання проблеми обчислювальної складності на основі критеріїв спрощення;

– розроблено програмні продукти, що дозволяють автоматизувати та алгоритмізувати побудову та спрощення визначених систем.

У ході проведеного дослідження було надано основні поняття, які застосовуються у системному аналізу для дослідження математичної моделі системи на об’єкті з її подальшим спрощенням. Так, у першому розділі було охарактеризовано основні етапи класичного емпіричного дослідження, за допомогою якого можна задати та досліджувати систему на об’єкті.

У другому розділі висвітлено проблематику побудови різних систем з поведінкою, описано відомості щодо побудови та дослідження розглянутих систем, визначено види нечітких мір та введено критерій оптимальності систем з поведінкою, який оснований на ступені недетермінованості.

Третій розділ роботи було присвячено розв'язку проблеми обчислювальної складності систем з поведінкою за допомогою спрощення двох родів та виділенню методологічних відмінностей між ними.

Крім того, в даній роботі було проілюстровано процес алгоритмізації та автоматизації введеної термінології. Для цього розроблено програмні продукти: «Elementary system with mask», «Directed generating system with behavior», «Content masks for elementary system with mask», «Entropies for content masks», «Entropies of content masks and 1 simplication», «Entropies for content masks and 2 simplication», які дозволяють автоматизувати побудову, дослідження та спрощення математичної моделі визначеного об'єкту. Так, перший програмний продукт «Elementary system with mask» дозволяє використати для початкової системи маску (правила) із заданими параметрами та визначити вибіркові змінні із функцією поведінки. Наступна програма «Directed generating system with behavior» алгоритмізує побудову систем з поведінкою направленого типу та виділяє відповідні функції поведінки для направленої та породжуючої систем. Третій «Content masks for elementary system with mask» і четвертий «Entropies for content masks» програмні продукти пов'язані, оскільки третя програма «Content masks» дозволяє побудувати необхідні змістовні підмаски, які використовуються у програмі «Entropies for content masks» для безпосереднього визначення Шенноновської ентропії для нейтральної системи з поведінкою, для якої виділяється ступінь недетермінованості систем. Програми «Entropies of content masks and 1 simplication», «Entropies for content masks and 2 simplication» також було побудовано на основі програмного продукту «Entropies for content masks» із додаванням можливості спрощення системи даних за допомогою першого і другого роду спрощень відповідно. Крім того, в останніх двох програмах було додано можливість користувачу задати необхідне спрощення як самому, так і використати варіант за замовчуванням.

ПЕРЕЛІК ПОСИЛАНЬ

1. Волкова В. Н. Теория систем и системный анализ: учебник для академического бакалавриата. 2-е изд., перераб. и доп. Москва : Юрайт, 2016. 462 с.
2. Берталанфи Л. Фон. История и статус общей теории систем. Системные исследования: ежегодник. Москва : Наука, 1973. 173 с.
3. Директор С, Рорар Д. Введение в теорию систем. Москва : Мир, 1974. 286 с.
4. Моисеев Н.Н. Математические задачи системного анализа. Москва : Наука, 1981. 488 с.
5. Перегудов Ф.И., Тарасенко Ф.П. Введение в системный анализ: учеб. пособие. Москва : Высшая школа, 1989. 367 с.
6. Берталанфи Л. Фон. Общая теория систем: критический обзор. Исследования по общей теории систем. Москва : Прогресс, 1969. 125 с.
7. Анфилатов В.С. Системный анализ в управлении : учеб. пособие. Москва : Финансы и статистика, 2002. 368 с.
8. Мессарович М., Такахара Я. Общая теория систем : математические основы. Москва : Мир, 1978. 312 с.
9. Садовский В. Н. Основания общей теории систем. Москва : Наука, 1974. 280 с.
10. Klir G. J., Elias D. Architecture of Systems Problem Solving. New York : Plenum Press, 1985. 354 p.
11. Мессарович М. Теория иерархических многоуровневых систем. Москва : Мир, 1973. 344 с.
12. Ackoff R.L. The art of Problem Solving. New York: Wiley-Interscience, 1978. 214 p.
13. Bertalauffy L. General System theory: Foundations, Development, Applications. New York : George Braziller, 1968. 289 p.

14. Mesaronic M. D. Views on General Systems Theory. New York : Wiley, 1964. 196 p.

15. Cavallo R. E., Klir G. J. Reconstruction of possibilistic behavior systems. Fuzzy Sets and Systems. New York : Wiley, 1982. 197 p.

16. Klir G. J. Identification of generative structures in empirical data. New York : Wiley, 1976. 104 p.

17. Кофман А. Введение в теорию нечетких множеств. Москва : Радио и связь, 1982. 432 с.

18. Wang Zh., Klir G. J., Fuzzy Measure Theory, Plenum Press, New York, 1991. 120 p.

19. Яблонский С. В. Введение в дискретную математику. Москва : Наука, 1986. 354 с.

ДОДАТОК А

Код програмного продукту: «Elementary system with mask»

```
var i,j,t,e, o,k,p,l,i1,j1,i2,j2,i3,j3:integer;
    m:array[1..10] of integer; //for mask
    m1:array[1..5,1..170] of integer; //for input system
    m2:array[1..170] of integer; //for determination of lines quantity
    m3:array[1..170,1..14] of integer; //for output system
    f1,f2:text;
    label g;
  begin
    j:=0; writeln('Write the quantity of W:'); while ((j<1)OR(j>170)) do readln(j);
    i:=0; writeln('Write the quantity of V:'); while ((i<1)OR(i>170)) do readln(i);
    e:=0; writeln('Write the quantity of numbers in mask:');
    while ((e<1)OR(e>10)) do readln(e);
    writeln('Write your mask:');
    p:=1;
    //check up the mask
    while p<=e do
      begin
      for k:=p to e do
        begin
         readln(m[k]);
         if m[k]=0 then t:=k-1;
          if (k>1) then
            begin
              if (m[k]<=m[k-1])OR((m[k]-m[k-1])<>1) then
                begin
                  writeln('You have a mistake in your mask: wrong number!');
                end
              else p:=p+1;
            end
          else p:=p+1;
        end;
      end;
    assign (f1, 'E:\input.txt'); reset(f1);
    //take the input system
    for k:=1 to i do
      for p:=1 to j do
        read(f1, m1[k,p]);
    close(f1);
    //rewrite output files
    assign(f1, 'E:\output1_1.txt'); rewrite(f1); p:=0;
    assign(f2, 'E:\output1_2.txt'); rewrite(f2);
    for k:=1 to e*i do write(f2,'S[',k,'] ');
    write(f2,'  | f[B](c)'); writeln(f2);
    i2:=1; j2:=1;
    writeln(f1,'Your system:');
    write(f1,'v\W '); for k:=1 to j do write(f1,' ',k);
    writeln(f1); write(f1,'V1   ');i1:=1;
    //write input system in file
    for k:=1 to i do
      begin
```

```
      for p:=1 to j do if (p<9) then write(f1,m1[k,p],' ')
                   else if (p<100) then write(f1,m1[k,p],'  ')
                     else write(f1,m1[k,p],'   ');
      writeln(f1); i1:=i1+1;
      if i1<i+1 then write(f1,'V',i1,'   ');
    end;
  p:=0;
  for i1:=1 to j do m2[i1]:=1;
  for k:=1 to j-e+1 do
    begin
      for j1:=1 to e do
        begin
          for i1:=1 to i do
            begin
              //writeng the output S[k]
              writeln(f1,'S[',i*p+i1,',',t+k,']=V[',i1,',',j1+k-1,']=',m1[i1,j1+k-1],'; ');
              //determination of otput system
              m3[i2,j2]:=m1[i1,j1+k-1]; j2:=j2+1;
            end;
          p:=p+1;
        end;
      p:=0; l:=0;
      if i2>1 then
        begin
          for i3:=1 to i2-1 do
            begin
              for j3:=1 to e*i do if m3[i2,j3]=m3[i3,j3] then l:=l+1;
              //determination the ending of the line
              if l=e*i then
              begin
                m2[i3]:=m2[i3]+1; goto g;
              end;
            l:=0;
            end;
        end;
      i2:=i2+1; o:=o+1;
g:      j2:=1; writeln(f1);
    end;
  //writing the output system
  for i1:=1 to o do
    begin
      write(f2,'  ');
      for j1:=1 to e*i do if j1<9 then write(f2,m3[i1,j1],'    ')
          else
            //writing output elements
            write(f2,m3[i1,j1],'  ');
      //writing probability of the line
      writeln(f2,'| ',m2[i1]/(j-2*(e-1)):1:5);
    end;
  close(f1); close(f2);
end.
```

ДОДАТОК Б

Код програмного продукту: «Directed generating system with behavior»

```
var i,j,t,e,  n,o,o1,o2,k,p,l,l1,l2,i1,j1,i2,j2,i3,j3,i4,j4,i5,j5,sum:integer;
    m:array[1..10] of integer;
    m0:array[0..170] of integer;
    m1:array[1..4,1..170] of integer;
    m2:array[1..170] of integer;
    m3:array[1..170,1..21] of integer;
    m4:array[1..170] of integer;
    m5:array[1..170,1..21] of integer;
    m6:array[1..170,1..21] of integer;
    m7:array[1..170] of integer;
    f1,f2:text;
    label g1,g2,g3,g4,g5;
  begin
    j:=0; writeln('Write the quantity of W:'); while ((j<1)OR(j>170)) do readln(j);
    i:=0; writeln('Write the quantity of V:'); while ((i<1)OR(i>170)) do readln(i);
    e:=0; writeln('Write the quantity of numbers in mask:');
    while ((e<1)OR(e>10)) do readln(e);
    writeln('Please, write the values in determinant:');
    for i1:=1 to i do
      begin
        m0[i1]:=-1;
        while ((m0[i1]<0)OR(m0[i1]>1)) do readln(m0[i1]);
      end;
    i1:=0;
    writeln('Write your mask:');
    p:=1;
    //check up the mask
    p:=1;
    while p<=e do
      begin
      for k:=p to e do
        begin
         readln(m[k]);
         if m[k]=0 then t:=k-1;
          if (k>1) then
            begin
              if (m[k]<=m[k-1])OR((m[k]-m[k-1])<>1) then
                begin
                  writeln('You have a mistake in your mask: wrong number!');
                end
              else p:=p+1;
            end
          else p:=p+1;
        end;
      end;
    l:=0;
    assign (f1, 'E:\input.txt'); reset(f1);
    for k:=1 to i do
      for p:=1 to j do
        read(f1, m1[k,p]);
```

```
close(f1);
assign(f1, 'E:\output2_1.txt'); rewrite(f1); p:=0;
assign(f2, 'E:\output2_2.txt'); rewrite(f2);
i2:=1; j2:=0; i4:=1; j4:=0; i5:=1; j5:=0;
writeln(f1,'Your system:');
write(f1,'v\W '); for k:=1 to j do write(f1,' ',k);
writeln(f1); write(f1,'V1   ');i1:=1;
for k:=1 to i do
  begin
    for p:=1 to j do if (p<9) then write(f1,m1[k,p],' ') else write(f1,m1[k,p],'  ');
    writeln(f1); i1:=i1+1; if i1<i+1 then write(f1,'V',i1,'   ');
  end;
p:=0; j1:=1;
for i1:=1 to j do begin m2[i1]:=1; m4[i1]:=1; m7[i1]:=1;end;
for i1:=1 to e*i do
  begin
    n:=i1 mod i; if n=0 then n:=i;
    writeln(f1,'lyamba(',n,';',m[j1],')=',i1);
    if ((i1 mod i) = 0) then begin j1:=j1+1; end;
  end;
j1:=1;
for i1:=1 to e*i do
  begin
    n:=i1 mod i;  if(n=0)then n:=i;
    if (m0[n]=0) then
      begin
        writeln(f1,'lyamba_e(',n,';',m[j1],')=',i1);
        if ((i1 mod i) = 0) then begin j1:=j1+1; end;
      end
    else
      begin
        if e=1 then writeln(f1,'lyamba_g(',n,';',m[j1],')=',i1)
        else
          if i1<e*i-i+1 then writeln(f1,'lyamba_g_^(',n,';',m[j1],')=',i1)
          else writeln(f1,'lyamba_g(',n,';',m[j1],')=',i1);
      end;
    if ((i1 mod i) = 0) then begin j1:=j1+1; end;
  end;
for k:=1 to j-e+1 do
  begin
    for j1:=1 to e do
      begin
        for i1:=1 to i do
          begin
            if i1=1 then
              if m0[i1]=0 then writeln(f1,'S[k,w]_e:')
              else
                begin
                  if j1<e then writeln(f1,'S[k,w]_g_^:')
                  else writeln(f1,'S[k,w]_g:')
                end
            else
              begin
                if m0[i1]<>m0[i1-1] then
                  if m0[i1]=0 then writeln(f1,'S[k,w]_e:')
                  else
                    begin
                      if j1<e then writeln(f1,'S[k,w]_g_^:')
                      else writeln(f1,'S[k,w]_g:')
                    end;
              end;
            writeln(f1,'S[',i*p+i1,',',t+k,']=V[',i1,',',j1+k-1,']=',m1[i1,j1+k-1],'; ');
            if m0[i1]=0 then
```

```
          begin
            j2:=j2+1;
            m3[i2,j2]:=m1[i1,j1+k-1];
          end
        else
          begin
            if j1<e then
              begin
                j4:=j4+1;
                m5[i4,j4]:=m1[i1,j1+k-1];
              end
            else
              begin
                j5:=j5+1;
                m6[i5,j5]:=m1[i1,j1+k-1];
              end;
          end;
      end;
    p:=p+1;
  end;
p:=0; l:=0; l1:=0; l2:=0;
if i2>1 then
  begin
    for i3:=1 to i2-1 do
      begin
        for j3:=1 to j2 do
          begin
            if m3[i2,j3]=m3[i3,j3] then l:=l+1;
          end;
        if l=j2 then
        begin
          m2[i3]:=m2[i3]+1; goto g2;
        end;
        l:=0;
      end;
  end;
g2:  if i4>1 then
  begin
    for i3:=1 to i4-1 do
      begin
        for j3:=1 to j4 do if m5[i4,j3]=m5[i3,j3] then l1:=l1+1;
        if l1=j4 then
        begin
          m4[i3]:=m4[i3]+1; goto g4;
        end;
      l1:=0;
      end;
  end;
g4:  if i5>1 then
  begin
    for i3:=1 to i5-1 do
      begin
        for j3:=1 to j5 do if m6[i5,j3]=m6[i3,j3] then l2:=l2+1;
        if l2=j5 then
        begin
          m7[i3]:=m7[i3]+1; goto g5;
        end;
      l2:=0;
      end;
  end;
g5:
if (l=j2)AND(l1<>j4)AND(l2<>j5) then begin i4:=i4+1; i5:=i5+1; o1:=o1+1; o2:=o2+1;end;
if (l=j2)AND(l1<>j4)AND(l2=j5) then begin i4:=i4+1; o1:=o1+1;end;
```

```
        if (l=j2)AND(l1=j4)AND(l2<>j5) then begin i5:=i5+1; o2:=o2+1;end;

        if (l<>j2)AND(l1=j4)AND(l2<>j5) then begin i2:=i2+1; o:=o+1; i5:=i5+1; o2:=o2+1; end;
        if (l<>j2)AND(l1=j4)AND(l2=j5) then begin i2:=i2+1; o:=o+1; end;
        if (l<>j2)AND(l1<>j4)AND(l2=j5) then begin i2:=i2+1; o:=o+1; i4:=i4+1;o1:=o1+1; end;

        if (l<>j2)AND(l1<>j4)AND(l2<>j5) then begin i2:=i2+1; i4:=i4+1; i5:=i5+1; o:=o+1; o1:=o1+1; o2:=o2+1;
end;

        if k<j-e+1 then begin j2:=0; j4:=0; j5:=0;end;
        writeln(f1);
      end;
    write(f2,'  ');
    for k:=1 to e*i do begin n:=k mod i; if n=0 then n:=i; if (m0[n]=0) then write(f2,'S[',k,'] '); end;
    write(f2,'  | f[B](e)'); writeln(f2);
    write(f2,'e = ');
    p:=0;
    sum:=0;
    for i1:=1 to o do sum:=sum+m2[i1];
    for i1:=1 to o do
      begin
          if i1>1 then write(f2,'     ');
          for j1:=1 to j2 do
            if j1<9 then write(f2,m3[i1,j1],'    ')
            else
              if j1<j2 then write(f2,m3[i1,j1],'     ')
              else
                write(f2,m3[i1,j1]);
          writeln(f2,'| ',m2[i1]/sum:1:5);
      end;
    writeln(f2);
    write(f2,'  ');
    for k:=1 to e*i do
      begin
        n:=k mod i;
        if n=0 then n:=i;
        if (m0[n]=1)AND(k<e*i-i+1) then write(f2,'S[',k,'] ');
      end;
    write(f2,'   | f[B](g_^)'); writeln(f2);
    write(f2,'g^= ');
    sum:=0;
    for i1:=1 to o1 do sum:=sum+m4[i1];
    for i1:=1 to o1 do
      begin
          if i1>1 then write(f2,'     ');
          for j1:=1 to j4 do
            if j1<9 then write(f2,m5[i1,j1],'    ')
            else
              if j1<j4 then write(f2,m5[i1,j1],'     ')
                else write(f2,m5[i1,j1]);
          writeln(f2,' | ',m4[i1]/sum:1:5);
      end;
    writeln(f2);
    write(f2,'  ');
    for k:=1 to e*i do
      begin
        n:=k mod i;
        if n=0 then n:=i;
        if (m0[n]=1)AND(k>e*i-i) then write(f2,'S[',k,'] ');
      end;
    write(f2,'   | f[B](g)'); writeln(f2);
    write(f2,'g = ');
    sum:=0;
```

```
  for i1:=1 to o2 do sum:=sum+m7[i1];
  for i1:=1 to o2 do
    begin
       if i1>1 then write(f2,'    ');
       for j1:=1 to j5 do
         if j1<9 then write(f2,m6[i1,j1],'    ')
         else
           if j1<j5 then write(f2,m6[i1,j1],'     ')
             else write(f2,m6[i1,j1]);
       writeln(f2,' | ',(m7[i1]/sum):1:5);
    end;
  close(f1); close(f2);
end.
```

ДОДАТОК В

Код програмного продукту: «Content masks for elementary system with mask»

```
var j,i,e,t,i2,j2,o,k,k1,k2,k4,k5,k6,k7,k8,k9,k10,k11,k12,count,count1,sum:integer;
   bool:byte; k3:real;
m:array[1..10] of integer; //для маски
a: array[1..20] of char;  //число 0/1 для коду Грея
m1:array[1..21,1..170] of integer; //початкова система
m2:array[1..170] of integer; //частоти
m3:array[1..170,1..21] of integer; //елементи використаної маски
f1,f2:text; //файлові змінні
begin
   //задання к-сті рядків/стовпчиків/чисел у масці
   j:=0; writeln('Write the quantity of W:'); while ((j<1)OR(j>170)) do readln(j);
   i:=0; writeln('Write the quantity of V:'); while ((i<1)OR(i>10)) do readln(i);
   e:=0; writeln('Write the quantity of numbers in mask:');
   while ((e<1)OR(e>5)) do readln(e);
   writeln('Write your mask:');
   k1:=1;
   //задання чисел маски та їх перевірка
   while k1<=e do
   begin
     for k:=k1 to e do
       begin
        readln(m[k]);
        if m[k]=0 then t:=k-1;
         if (k>1) then
           begin
             if (m[k]<=m[k-1])OR((m[k]-m[k-1])<>1) then
               begin
                 writeln('You have a mistake in your mask: wrong number!');
               end
             else k1:=k1+1;
           end
         else k1:=k1+1;
       end;
   end;
   k1:=0;k2:=0;
   for k:=1 to e do if m[k]=0 then k1:=k1+1;
   //відкриття файлу з початковою системою
   assign (f1, 'E:\input.txt'); reset(f1);
   for k:=1 to i do
     for k1:=1 to j do
       read(f1, m1[k,k1]);
   k:=0;k1:=0;
   close(f1);
   assign(f2, 'E:\l5_rez.txt'); rewrite(f2);
   for k:=1 to e*i do a[k]:='0'; //запис всіх чисел Грея,як "0"
   for k:=1 to j do m2[k]:=1; //задання всіх частот,як "1"
   k:=0;
   REPEAT
```

```
for k4:=1 to i do if a[k4]='1' then k5:=k5+1; //підрахунок к-сті "1"
if (k5>0) then
  begin
    for k4:=1 to i do
      begin
        for k6:=0 to e-1 do
          begin
            //підрахунок к-сті "1" у коді Грея
            if (a[i*k6+k4]='1') then k7:=k7+1;
          end;
        if (k7>0) then k8:=k8+1;
        k7:=0;
      end;
    //перевірка умови знахоження хоча б однієї "1" у стовпчику
    //(не закресленого елемента)
    if k8=i then
      begin
        for k6:=1 to j do m2[k6]:=1;
        count:=0;count1:=0;i2:=1;j2:=1;
        k2:=k2+1;
        writeln(f2,'Your mask № ',k2,':');//запис фрази у файл
        for k6:=1 to e do write(f2,' -');
        writeln(f2);
        //запис у файл "гарного" виду маски
        for k6:=i downto 1 do
          begin
            for k9:=e-1 downto 0 do if a[i*k9+k6]='0' then write(f2,'|X')
                                    else write(f2,'| ');
            write(f2,'|');
            writeln(f2);
            for k9:=1 to e do write(f2,' -');
            writeln(f2);
          end;
        //запис у файл назви стовпчиків таблиць
        for k6:=i*e downto 1 do if a[k6]='1' then
        begin write(f2,' S[',e*i-k6+1,']');count:=count+1;end;
        write(f2,' | f[B](c)'); writeln(f2);
        for k6:=1 to j-e+1 do
          begin
            for k9:=1 to e do
              for k10:=1 to i do
                //перевірка на "0" або "1" у послідовності a
                if a[e*i-(k9*i-i-1+k10)]='1' then
                  begin
                    //запис вибіркового числа у вихідну с-му(при "1")
                    m3[i2,j2]:=m1[k10,k9+k6-1];
                    j2:=j2+1; //підрахунок к-сті записаних чисел
                  end;
            //блок перевірки на спідпадання рядків для підрахунку частоти
            if i2>1 then
              begin
                for k9:=1 to i2-1 do
                  begin
                    for k10:=1 to count do if m3[i2,k10]=m3[k9,k10]
                                           then k12:=k12+1;
                    if k12=count then
                      begin
                        //враховуємо співпадання рядків
                        bool:=1;
                        m2[k9]:=m2[k9]+1;
                      end;
                    k12:=0;
                  end;
```

```
          end;
        if bool=0 then   //якщо рядки не співпали
          begin
            i2:=i2+1;
            count1:=count1+1;
            j2:=1;
          end
        else j2:=1;  //враховуємо співпадання рядків для частоти
        bool:=0;
      end;
    sum:=0;
    //сума всіх частот(для ймовірності)
    for k9:=1 to count1 do
      begin
        sum:=sum+m2[k9];
      end;
    for k9:=1 to count1 do
      begin
        for k10:=1 to count do
          begin
            //запис елементів вихідної с-ми у файл
            write(f2,'   ',m3[k9,k10],' ');
          end;
        //запис ймовірності кожного рядка системи
        writeln(f2,' | ',m2[k9]/sum:1:5);
      end;
  end;
  k8:=0;
 end;
k5:=0;
k:=k+1;
k1:=1;
k3:=k;
//зміна чисел у коді Грея
while k3-2*trunc(k3/2)=0 do
  begin
    k3:=k3/2;
    inc(k1);
  end;
if k1<=i*e then
  case a[k1] of
    '0': a[k1]:='1';
    '1': a[k1]:='0'
  end
UNTIL k1>e*i;
close(f2);
end.
```

ДОДАТОК Г

Код програмного продукту: «Entropies for content masks»

```
const x=1; //к-сть одиниць у визначнику входу
var j,i,e,t,i2,j2,o,k,k1,k2,k4,k5,k6,k7,k8,k9,k10,k11,k12,count,count1,sum:integer;
bool:byte; k3:real;
m:array[1..10] of integer; //для маски
a: array[1..20] of char;  //число 0/1 для коду Грея
m1:array[1..21,1..170] of integer; //початкова система
m2:array[1..170] of integer; //частоти
m3:array[1..170,1..21] of integer; //елементи використаної маски
H,H1,H2:real; //ентропії
lev:array[1..5] of real; //запис найменших ентропій для нейтральної с-ми
f1,f2:text; //файлові змінні
begin
    //задання к-сті рядків/стовпчиків/чисел у масці
    j:=0; writeln('Write the quantity of W:'); while ((j<1)OR(j>170)) do readln(j);
    i:=0; writeln('Write the quantity of V:'); while ((i<1)OR(i>10)) do readln(i);
    e:=0; writeln('Write the quantity of numbers in mask:');
    while ((e<1)OR(e>5)) do readln(e);
    writeln('Write your mask:');
    k1:=1;
    //задання чисел маски та їх перевірка
    while k1<=e do
    begin
      for k:=k1 to e do
        begin
         readln(m[k]);
         if m[k]=0 then t:=k-1;
          if (k>1) then
            begin
              if (m[k]<=m[k-1])OR((m[k]-m[k-1])<>1) then
                begin
                  writeln('You have a mistake in your mask: wrong number!');
                end
              else k1:=k1+1;
            end
          else k1:=k1+1;
        end;
    end;
    k1:=0;k2:=0;
    for k:=1 to e do if m[k]=0 then k1:=k1+1;
    //відкриття файлу з початковою системою
    assign (f1, 'E:\input.txt'); reset(f1);
    for k:=1 to i do
      for k1:=1 to j do
        read(f1, m1[k,k1]);
    k:=0;k1:=0;
    close(f1);
    assign(f2, 'E:\l6_rez.txt'); rewrite(f2);
    for k:=1 to i+1 do lev[k]:=1000;
    for k:=1 to e*i do a[k]:='0'; //запис всіх чисел Грея,як "0"
    for k:=1 to j do m2[k]:=1; //задання всіх частот,як "1"
```

```
k:=0;
REPEAT
  for k4:=1 to i do if a[k4]='1' then k5:=k5+1;  //підрахунок к-сті "1"
  if (k5>0) then
    begin
      for k4:=1 to i do
        begin
          for k6:=0 to e-1 do
            begin
              //підрахунок к-сті "1" у коді Грея відносно маски
              if (a[i*k6+k4]='1') then k7:=k7+1;
            end;
          if (k7>0) then k8:=k8+1;
          k7:=0;
        end;
      //перевірка умови знахоження хоча б однієї "1" у стовпчику
      //(не закресленого елемента)
      if k8=i then
        begin
            for k6:=1 to j do m2[k6]:=1;
            count:=0;count1:=0;i2:=1;j2:=1;H:=0;
            k2:=k2+1;
            writeln(f2,'Your mask № ',k2,':'); //запис фрази у файл
            for k6:=1 to e do write(f2,' -');
            writeln(f2);
            //запис у файл "гарного" виду маски
            for k6:=i downto 1 do
              begin
                for k9:=e-1 downto 0 do if a[i*k9+k6]='0' then write(f2,'|X') else write(f2,'| ');
                write(f2,'|');
                writeln(f2);
                for k9:=1 to e do write(f2,' -');
                writeln(f2);
              end;
            //запис у файл назви стовпчиків таблиць
            for k6:=i*e downto 1 do if a[k6]='1' then begin write(f2,' S[',e*i-k6+1,']');count:=count+1;end;
            write(f2,' | f[B](c)'); writeln(f2);
            for k6:=1 to j-e+1 do
              begin
                for k9:=1 to e do
                  for k10:=1 to i do
                    //перевірка на "0" або "1" у послідовності a
                    if a[e*i-(k9*i-i-1+k10)]='1' then
                      begin
                        //запис вибіркового числа у вихідну с-му(при "1")
                        m3[i2,j2]:=m1[k10,k9+k6-1];
                        j2:=j2+1; //підрахунок к-сті записаних чисел
                      end;
                //блок перевірки на спідпадання рядків для підрахунку частоти
                if i2>1 then
                  begin
                    for k9:=1 to i2-1 do
                      begin
                        for k10:=1 to count do if m3[i2,k10]=m3[k9,k10] then k12:=k12+1;
                        if k12=count then
                          begin
                            //враховуємо співпадання рядків
                            bool:=1;
                            m2[k9]:=m2[k9]+1;
                          end;
                        k12:=0;
                      end;
                  end;
```

```
      if bool=0 then  //якщо рядки не співпали
        begin
          i2:=i2+1;
          count1:=count1+1;
          j2:=1;
        end
      else j2:=1; //враховуємо співпадання рядків для частоти
      bool:=0;
    end;
  for k9:=1 to count1 do
    begin
      for k10:=1 to count do
        begin
          //запис елементів вихідної с-ми у файл
          write(f2,'   ',m3[k9,k10],' ');
        end;
      //запис ймовірності кожного рядка системи
      writeln(f2,' | ',m2[k9]/(j-e+1):1:5);
      //підрахунок Шеннонoвської ентропії
      H:=H+(m2[k9]/j)*ln(m2[k9]/(j-e+1))/ln(2);
    end;
//врахування знаку "-" у Шеннонoвській ентропії
H:=-H;
writeln(f2,'H=',H:1:5);
//початок формування нейтральної с-ми (G)
for k6:=1 to j do m2[k6]:=1;
count:=0;count1:=0;i2:=1;j2:=1;H1:=0;
//той самий підрахунок,що й для загальної с-ми(тільки лівої частини
for k6:=i*e downto i+1 do if a[k6]='1' then begin count:=count+1;end;
for k6:=1 to j-e+1 do
  begin
    for k9:=1 to e do
      for k10:=1 to i do
        if a[e*i-(k9*i-i-1+k10)]='1' then
          begin
            m3[i2,j2]:=m1[k10,k9+k6-1];
            j2:=j2+1;
          end;
    if i2>1 then
      begin
        for k9:=1 to i2-1 do
          begin
            for k10:=1 to count do if m3[i2,k10]=m3[k9,k10] then k12:=k12+1;
            if k12=count then
              begin
                bool:=1;
                m2[k9]:=m2[k9]+1;
              end;
            k12:=0;
          end;
      end;
    if bool=0 then
      begin
        i2:=i2+1;
        count1:=count1+1;
        j2:=1;
      end
    else j2:=1;
    bool:=0;
  end;
//підрахунок ентропії для нейтральної с-ми
if count>0 then
for k9:=1 to count1 do
```

```
            begin
              H1:=H1+(m2[k9]/j)*ln(m2[k9]/(j-e+1))/ln(2);
            end;
          H1:=-H1;
          writeln(f2,'H(G`)=',H1:1:5);
          //підрахунок ентропії для нейтральної с-ми
          H1:=H-H1;
          writeln(f2,'H(G|G`)=',H1:1:5);
          k10:=0;
          for k9:=e*i downto i do if a[k9]='0' then inc(k10);
          //перевірка ентропії на мінімальність
          if H1<lev[i*e-i-k10+1] then lev[i*e-i-k10+1]:=H1;
          for k6:=1 to j do m2[k6]:=1;
          count:=0;count1:=0;i2:=1;j2:=1;H2:=0;
        end;
      k8:=0;
    end;
  k5:=0;
  k:=k+1;
  k1:=1;
  k3:=k;
  //зміна чисел у коді Грея
  while k3-2*trunc(k3/2)=0 do
    begin
      k3:=k3/2;
      inc(k1);
    end;
  if k1<=i*e then
    case a[k1] of
      '0': a[k1]:='1';
      '1': a[k1]:='0'
    end
 UNTIL k1>e*i;
 writeln(f2,lev);
 close(f2);
end.
```

ДОДАТОК Д

Код програмного продукту: «Entropies of content masks and 1 simplication»

```
const x=1; //к-сть одиниць у визначнику входу
var j,i,e,t,i2,j2,o,k,k1,k2,k4,k5,k6,k7,k8,k9,k10,k11,k12,count,count1,sum:integer;
bool:byte; k3:real;
//for 1 simplication
b: array[1..10] of char;
p1, p2: real;
i1,p, p3, p4,p5,p6,p7, del1: integer;
m4:array[1..21,1..170] of integer;
del:array[1..10] of integer;
f3,f4,f5:string;
//
m:array[1..10] of integer; //для маски
a: array[1..20] of char;  //число 0/1 для коду Грея
m1:array[1..21,1..170] of integer; //початкова система
m2:array[1..170] of integer; //частоти
m3:array[1..170,1..21] of integer; //елементи використаної маски
H,H1,H2:real; //ентропії
lev:array[1..5] of real; //запис найменших ентропій для нейтральної с-ми
f1,f2:text; //файлові змінні
begin
    //задання к-сті рядків/стовпчиків/чисел у масці
    j:=0; writeln('Write the quantity of W:'); while ((j<1)OR(j>170)) do readln(j);
    i:=0; writeln('Write the quantity of V:'); while ((i<1)OR(i>10)) do readln(i);
    e:=0; writeln('Write the quantity of numbers in mask:');
    while ((e<1)OR(e>5)) do readln(e);
    writeln('Write your mask:');
    k1:=1;
    //задання чисел маски та їх перевірка
    while k1<=e do
    begin
      for k:=k1 to e do
        begin
         readln(m[k]);
         if m[k]=0 then t:=k-1;
          if (k>1) then
            begin
              if (m[k]<=m[k-1])OR((m[k]-m[k-1])<>1) then
                begin
                  writeln('You have a mistake in your mask: wrong number!');
                end
              else k1:=k1+1;
            end
          else k1:=k1+1;
        end;
    end;
    k1:=0;k2:=0;
    for k := 1 to e do if m[k]=0 then k1:=k1+1;
    //for 1 simplication
    i1 := i;
    for k:= 1 to i1 do del[k]:=1;
```

```
writeln('Do you want to use simplication of 1 degree?');
readln(f3);
if f3='yes' then
  begin
    writeln('Do you want to remove them by yourself?');
    readln(f4);
    if f4='yes' then
      begin
        for k:=1 to i1 do
          begin
            writeln('Do you want to remove ',k,' condition?');
            readln(f5);
            if f5='yes' then del[k]:=0;
          end;
      end
    else writeln('Okay, than the program will use all removements, which possible');
  end;
//

//відкриття файлу з початковою системою
assign (f1, 'E:\input.txt'); reset(f1);
 for k:=1 to i do
   begin
     for k1:=1 to j do
       begin
         read(f1, m1[k,k1]);
         m4[k,k1] := m1[k,k1];
         //write(m1[k,k1]);
       end;
     //writeln;
   end;
k:=0;k1:=0;
close(f1);
assign(f2, 'E:\Entropies+1_simplication+whole_view.txt'); rewrite(f2);


if (f3='yes')AND(f4<>'yes') then
  begin
    for p:=1 to i do b[p]:='0';
    p1:=0;
    REPEAT
      //1 simplification
      p4:=0;
      for p3:=1 to i1 do
        if b[p3]='1' then inc(p4);
      if (p4>0)AND(p4<i1) then
        begin
          for p5:=1 to i1 do
            for p6:=1 to j do
              m1[p5,p6] := m4[p5,p6];
          writeln(f2,'----------------------------------------------------------------------');
          writeln(f2,'Your conditions:');
          for p := 1 to i1 do
            if b[p]='1' then write(f2,'V',p,' ');
          writeln(f2);
          writeln(f2,'----------------------------------------------------------------------');
          p6:=0;
          for p:=1 to i1 do
            if b[p]='1' then
              begin
                inc(p6);
                for p5:=1 to j do m1[p6,p5]:=m4[p,p5];
              end;
```

```
i := p4;
{for p5:=1 to i do
  begin
    for p6 := 1 to j do write(m1[p5,p6]);
    writeln(f2);
  end; }
//Entropies of content masks
  for k:=1 to 5 do lev[k]:=0;
  for k:=1 to i+1 do lev[k]:=1000;
  for k:=1 to e*i do a[k]:='0'; //запис всіх чисел Грея,як "0"
  for k:=1 to j do m2[k]:=1; //задання всіх частот,як "1"
  k := 0;
  k2 := 0;
  REPEAT
    for k4:=1 to i do if a[k4]='1' then k5:=k5+1;  //підрахунок к-сті "1"
    if (k5>0) then
      begin
        for k4:=1 to i do
          begin
            for k6:=0 to e-1 do
              begin
                //підрахунок к-сті "1" у коді Грея відносно маски
                if (a[i*k6+k4]='1') then k7:=k7+1;
              end;
            if (k7>0) then k8:=k8+1;
            k7:=0;
          end;
        //перевірка умови знахоження хоча б однієї "1" у стовпчику
        //(не закресленого елемента)
        if k8=i then
          begin
              for k6:=1 to j do m2[k6]:=1;
              count:=0;count1:=0;i2:=1;j2:=1;H:=0;
              k2:=k2+1;
              writeln(f2,'Your mask № ',k2,':'); //запис фрази у файл
              for k6:=1 to e do write(f2,' -');
              writeln(f2);
              //запис у файл "гарного" виду маски
              for k6:=i downto 1 do
                begin
                  for k9:=e-1 downto 0 do if a[i*k9+k6]='0' then write(f2,'|X') else write(f2,'| ');
                  write(f2,'|');
                  writeln(f2);
                  for k9:=1 to e do write(f2,' -');
                  writeln(f2);
                end;
              //запис у файл назви стовпчиків таблиць
              for k6:=i*e downto 1 do if a[k6]='1' then begin write(f2,' S[',e*i-k6+1,']');count:=count+1;end;
              write(f2,' | f[B](c)'); writeln(f2);
              for k6:=1 to j-e+1 do
                begin
                  for k9:=1 to e do
                    for k10:=1 to i do
                      //перевірка на "0" або "1" у послідовності a
                      if a[e*i-(k9*i-i-1+k10)]='1' then
                        begin
                          //запис вибіркового числа у вихідну с-му(при "1")
                          m3[i2,j2]:=m1[k10,k9+k6-1];
                          j2:=j2+1; //підрахунок к-сті записаних чисел
                        end;
                  //блок перевірки на спідпадання рядків для підрахунку частоти
                  if i2>1 then
                    begin
```

```
        for k9:=1 to i2-1 do
          begin
            for k10:=1 to count do if m3[i2,k10]=m3[k9,k10] then k12:=k12+1;
            if k12=count then
              begin
                //враховуємо співпадання рядків
                bool:=1;
                m2[k9]:=m2[k9]+1;
              end;
            k12:=0;
          end;
      end;
    if bool=0 then  //якщо рядки не співпали
      begin
        i2:=i2+1;
        count1:=count1+1;
        j2:=1;
      end
    else j2:=1; //враховуємо співпадання рядків для частоти
    bool:=0;
  end;
 for k9:=1 to count1 do
  begin
    for k10:=1 to count do
      begin
        //запис елементів вихідної с-ми у файл
        write(f2,'   ',m3[k9,k10],' ');
      end;
    //запис ймовірності кожного рядка системи
    writeln(f2,' | ',m2[k9]/(j-e+1):1:5);
    //підрахунок Шенноновської ентропії
    H:=H+(m2[k9]/j)*ln(m2[k9]/(j-e+1))/ln(2);
  end;
//врахування знаку "-" у Шенноновській ентропії
H:=-H;
writeln(f2,'H=',H:1:5);
//початок формування нейтральної с-ми (G)
for k6:=1 to j do m2[k6]:=1;
count:=0;count1:=0;i2:=1;j2:=1;H1:=0;
//той самий підрахунок,що й для загальної с-ми(тільки лівої частини
for k6:=i*e downto i+1 do if a[k6]='1' then begin count:=count+1;end;
for k6:=1 to j-e+1 do
 begin
  for k9:=1 to e do
    for k10:=1 to i do
      if a[e*i-(k9*i-i-1+k10)]='1' then
        begin
          m3[i2,j2]:=m1[k10,k9+k6-1];
          j2:=j2+1;
        end;
  if i2>1 then
    begin
      for k9:=1 to i2-1 do
        begin
          for k10:=1 to count do if m3[i2,k10]=m3[k9,k10] then k12:=k12+1;
          if k12=count then
            begin
              bool:=1;
              m2[k9]:=m2[k9]+1;
            end;
          k12:=0;
        end;
    end;
```

```
              if bool=0 then
                begin
                  i2:=i2+1;
                  count1:=count1+1;
                  j2:=1;
                end
              else j2:=1;
              bool:=0;
            end;
          //підрахунок ентропії для нейтральної с-ми
          if count>0 then
          for k9:=1 to count1 do
            begin
              H1:=H1+(m2[k9]/j)*ln(m2[k9]/(j-e+1))/ln(2);
            end;
          H1:=-H1;
          writeln(f2,'H(G`)=',H1:1:5);
          //підрахунок ентропії для нейтральної с-ми
          H1:=H-H1;
          writeln(f2,'H(G|G`)=',H1:1:5);
          k10:=0;
          for k9:=e*i downto i do if a[k9]='0' then inc(k10);
          //перевірка ентропії на мінімальність
          if H1<lev[i*e-i-k10+1] then lev[i*e-i-k10+1]:=H1;
          for k6:=1 to j do m2[k6]:=1;
          count:=0;count1:=0;i2:=1;j2:=1;H2:=0;
        end;
      k8:=0;
    end;
  k5:=0;
  k:=k+1;
  k1:=1;
  k3:=k;
  //зміна чисел у коді Грея
  while k3-2*trunc(k3/2)=0 do
    begin
      k3:=k3/2;
      inc(k1);
    end;
  if k1<=i*e then
    case a[k1] of
      '0': a[k1]:='1';
      '1': a[k1]:='0'
    end
 UNTIL k1>e*i;
 for k9 := 1 to i+1 do write(f2,lev[k9],' ');
 writeln(f2);
end;
p1:=p1+1;
p:=1;
p2:=p1;
while p2-2*trunc(p2/2)=0 do
  begin
    p2 := p2/2;
    inc(p);
  end;
if p<=i1 then case b[p] of
    '0': b[p]:='1';
    '1': b[p]:='0' end
UNTIL p>i1;
end;
//1 simplication, for user choise
if (f3='yes')AND(f4='yes') then
```

```
begin
 p4:=0;
 writeln(f2,'Your conditions:');
 for p := 1 to i1 do
   if del[p]=1 then
     begin
       write(f2,'V',p,' ');
       inc(p4);
     end;
     writeln(f2);
 p6:=0;
 for p:=1 to i1 do
   if del[p]=1 then
     begin
       inc(p6);
       for p5:=1 to j do m1[p6,p5]:=m4[p,p5];
     end;
 i := p4;
end;
////
////entropies

       for k:=1 to i+1 do lev[k]:=1000;
       for k:=1 to e*i do a[k]:='0'; //запис всіх чисел Грея,як "0"
       for k:=1 to j do m2[k]:=1; //задання всіх частот,як "1"
       k := 0;
       k2 := 0;
       REPEAT
         for k4:=1 to i do if a[k4]='1' then k5:=k5+1;  //підрахунок к-сті "1"
         if (k5>0) then
           begin
             for k4:=1 to i do
               begin
                 for k6:=0 to e-1 do
                   begin
                     //підрахунок к-сті "1" у коді Грея відносно маски
                     if (a[i*k6+k4]='1') then k7:=k7+1;
                   end;
                 if (k7>0) then k8:=k8+1;
                 k7:=0;
               end;
             //перевірка умови знахоження хоча б однієї "1" у стовпчику
             //(не закресленого елемента)
             if k8=i then
               begin
                   for k6:=1 to j do m2[k6]:=1;
                   count:=0;count1:=0;i2:=1;j2:=1;H:=0;
                   k2:=k2+1;
                   writeln(f2,'Your mask № ',k2,':'); //запис фрази у файл
                   for k6:=1 to e do write(f2,' -');
                   writeln(f2);
                   //запис у файл "гарного" виду маски
                   for k6:=i downto 1 do
                     begin
                       for k9:=e-1 downto 0 do if a[i*k9+k6]='0' then write(f2,'|X') else write(f2,'| ');
                       write(f2,'|');
                       writeln(f2);
                       for k9:=1 to e do write(f2,' -');
                       writeln(f2);
                     end;
                   //запис у файл назви стовпчиків таблиць
                   for k6:=i*e downto 1 do if a[k6]='1' then begin write(f2,' S[',e*i-k6+1,']');count:=count+1;end;
                   write(f2,' | f[B](c)'); writeln(f2);
```

```
  for k6:=1 to j-e+1 do
    begin
      for k9:=1 to e do
        for k10:=1 to i do
          //перевірка на "0" або "1" у послідовності a
          if a[e*i-(k9*i-i-1+k10)]='1' then
            begin
              //запис вибіркового числа у вихідну с-му(при "1")
              m3[i2,j2]:=m1[k10,k9+k6-1];
              j2:=j2+1; //підрахунок к-сті записаних чисел
            end;
      //блок перевірки на спідпадання рядків для підрахунку частоти
      if i2>1 then
        begin
          for k9:=1 to i2-1 do
            begin
              for k10:=1 to count do if m3[i2,k10]=m3[k9,k10] then k12:=k12+1;
              if k12=count then
                begin
                  //враховуємо співпадання рядків
                  bool:=1;
                  m2[k9]:=m2[k9]+1;
                end;
              k12:=0;
            end;
        end;
      if bool=0 then  //якщо рядки не співпали
        begin
          i2:=i2+1;
          count1:=count1+1;
          j2:=1;
        end
      else j2:=1; //враховуємо співпадання рядків для частоти
      bool:=0;
    end;
  for k9:=1 to count1 do
    begin
      for k10:=1 to count do
        begin
          //запис елементів вихідної с-ми у файл
          write(f2,'   ',m3[k9,k10],' ');
        end;
      //запис ймовірності кожного рядка системи
      writeln(f2,' | ',m2[k9]/(j-e+1):1:5);
      //підрахунок Шенноновської ентропії
      H:=H+(m2[k9]/j)*ln(m2[k9]/(j-e+1))/ln(2);
    end;
//врахування знаку "-" у Шенноновській ентропії
H:=-H;
writeln(f2,'H=',H:1:5);
//початок формування нейтральної с-ми (G)
for k6:=1 to j do m2[k6]:=1;
count:=0;count1:=0;i2:=1;j2:=1;H1:=0;
//той самий підрахунок,що й для загальної с-ми(тільки лівої частини
for k6:=i*e downto i+1 do if a[k6]='1' then begin count:=count+1;end;
for k6:=1 to j-e+1 do
  begin
    for k9:=1 to e do
      for k10:=1 to i do
        if a[e*i-(k9*i-i-1+k10)]='1' then
          begin
            m3[i2,j2]:=m1[k10,k9+k6-1];
            j2:=j2+1;
```

```
                  end;
               if i2>1 then
                 begin
                   for k9:=1 to i2-1 do
                     begin
                       for k10:=1 to count do if m3[i2,k10]=m3[k9,k10] then k12:=k12+1;
                       if k12=count then
                         begin
                           bool:=1;
                           m2[k9]:=m2[k9]+1;
                         end;
                       k12:=0;
                     end;
                 end;
               if bool=0 then
                 begin
                   i2:=i2+1;
                   count1:=count1+1;
                   j2:=1;
                 end
               else j2:=1;
               bool:=0;
             end;
           //підрахунок ентропії для нейтральної с-ми
           if count>0 then
           for k9:=1 to count1 do
             begin
               H1:=H1+(m2[k9]/j)*ln(m2[k9]/(j-e+1))/ln(2);
             end;
           H1:=-H1;
           writeln(f2,'H(G`)=',H1:1:5);
           //підрахунок ентропії для нейтральної с-ми
           H1:=H-H1;
           writeln(f2,'H(G|G`)=',H1:1:5);
           k10:=0;
           for k9:=e*i downto i do if a[k9]='0' then inc(k10);
           //перевірка ентропії на мінімальність
           if H1<lev[i*e-i-k10+1] then lev[i*e-i-k10+1]:=H1;
           for k6:=1 to j do m2[k6]:=1;
           count:=0;count1:=0;i2:=1;j2:=1;H2:=0;
         end;
       k8:=0;
     end;
   k5:=0;
   k:=k+1;
   k1:=1;
   k3:=k;
   //зміна чисел у коді Грея
   while k3-2*trunc(k3/2)=0 do
     begin
       k3:=k3/2;
       inc(k1);
     end;
   if k1<=i*e then
     case a[k1] of
       '0': a[k1]:='1';
       '1': a[k1]:='0'
     end
  UNTIL k1>e*i;
  for k9 := 1 to i+1 do write(f2,lev[k9],' ');
  writeln(f2);
 close(f2);
end.
```

ДОДАТОК Е

Код програмного продукту: «Entropies for content masks and 2 simplication»

```
const x=1; //к-сть одиниць у визначнику входу
var
   j,i,e,t,i2,j2,o,k,k1,k2,k4,k5,k6,k7,k8,k9,k10,k11,k12,count,count1,sum:integer;
   bool:byte; k3:real;
   m:array[1..10] of integer; //для маски
   a: array[1..20] of char;  //число 0/1 для коду Грея
   m1:array[1..21,1..170] of integer; //початкова система
   m2:array[1..170] of integer; //частоти
   m3:array[1..170,1..21] of integer; //елементи використаної маски
   H,H1,H2:real; //ентропії
   lev:array[1..5] of real; //запис найменших ентропій для нейтральної с-ми
   f1,f2:text; //файлові змінні

   //for 2 simpication
   b1: array[1..10] of char;
   b2: array[1..10] of char;
   b3: array[1..510,1..10] of char;
   z,z1,z3,z4,z5,z6,kol,kol1,all,n1,kol2,n: integer;
   z2:real;
   flag:byte;
   m5:array[1..21,1..170] of integer;
   f7,f8:string;
begin
   //задання к-сті рядків/стовпчиків/чисел у масці
   j:=0; writeln('Write the quantity of W:'); while ((j<1)OR(j>170)) do readln(j);
   i:=0; writeln('Write the quantity of V:'); while ((i<1)OR(i>10)) do readln(i);
   e:=0; writeln('Write the quantity of numbers in mask:');
   while ((e<1)OR(e>5)) do readln(e);
   writeln('Write your mask:');
   k1:=1;
   //задання чисел маски та їх перевірка
   while k1<=e do
   begin
     for k:=k1 to e do
       begin
        readln(m[k]);
        if m[k]=0 then t:=k-1;
         if (k>1) then
           begin
             if (m[k]<=m[k-1])OR((m[k]-m[k-1])<>1) then
               begin
                 writeln('You have a mistake in your mask: wrong number!');
               end
             else k1:=k1+1;
           end
         else k1:=k1+1;
       end;
   end;
```

```
k1:=0;k2:=0;
for k:=1 to e do if m[k]=0 then k1:=k1+1;
//відкриття файлу з початковою системою
assign (f1, 'E:\input.txt'); reset(f1);
 for k:=1 to i do
   begin
     for k1:=1 to j do
       begin
         read(f1, m1[k,k1]);
         //write(m1[k,k1]);
         m5[k,k1]:=m1[k,k1];
       end;
     //writeln;
   end;
close(f1);
k:=0;k1:=0;
assign(f2, 'E:\Entropies+2_simplication+whole_view.txt'); rewrite(f2);
writeln('Do you want to use simplication of 2 degree?');
  readln(f7);
  if f7='yes' then
    begin
      writeln('Do you want to simpify them by yourself?');
      readln(f8);
      if f8='yes' then
        begin
          writeln('Which condition do you want to simpify?');
          readln(n1);
        end
      else writeln('Okay, than the program will use all simplification, which possible.');
    end;


for z6:=1 to i do
  begin
    if (f7='yes')AND(f8<>'yes') then n1 := z6;
    if ((f7='yes')AND(f8='yes')AND(z6=1))OR((f7='yes')AND(f8<>'yes')) then
      begin
        //writeln('YESSSSSSSSS');
        writeln(f2,'We simplify ',n1 ,' row by 2 simplication');
        n := m5[n1,1];
        for z := 1 to n do b1[z]:='0';
        z1 := 0;
        all := 0;
        for k1 := 2 to j do
          begin
            if m1[n1,k1] > n then n:=m1[n1,k1];
          end;
        //writeln(n);
        REPEAT
          kol := 0;
          kol1 := 1;
          for z3:=1 to n do
            if b1[z3]='1' then
              begin
                inc(kol);
                if z3<n then
                  if b1[z3+1]='1' then inc(kol1);
              end;
          if (kol = kol1)AND(kol>1)AND(kol<n) then
            begin
              inc(all);
              for z:=1 to n do b3[all,z]:=b1[z];
            end;
```

```
  kol := 0;
  kol1 := 0;
  for z3 := 1 to n do b2[z3]:=b1[z3];
  for z3 := 1 to n do
    if b2[z3]='1' then b2[z3]:='2'
      else b2[z3] := '1';
  for z3:=2 to n do
    if (b2[z3]=b2[z3-1])AND(b2[z3]<>b2[z3+1]) then inc(kol);
  if (b2[1]='2')AND(kol>1) then
        begin
          inc(all);
          for z:=1 to n do b3[all,z]:=b2[z];
        end;
  inc(z1);
  z:=1;
  z2:=z1;
  while z2-2*trunc(z2/2)=0 do    begin
  z2:=z2/2;
  inc(z)                         end;
  if z<=n then
    case b1[z] of
      '0': b1[z]:='1';
      '1': b1[z] := '0' end;
UNTIL z>n;


kol := 0;
kol1:=0;
for z:=1 to all do
  begin
    for z3:=1 to n do
      begin
        if z3=1 then
          if (b3[z,z3]=b3[z,z3+1])AND(b3[z,z3]<>'0') then write(f2, '(V',z3,' ')
          else write(f2, 'V',z3,' ')
        else if (z3>1) AND (z3<n) then
          begin
            if (b3[z,z3]=b3[z,z3-1]) AND (b3[z,z3]<>b3[z,z3+1]) AND(b3[z,z3]<>'0')
              then write(f2, 'V',z3,') ')
            else if (b3[z,z3]<>b3[z,z3-1]) AND (b3[z,z3]=b3[z,z3+1]) AND(b3[z,z3]<>'0')
              then write(f2, '(V',z3,' ')
            else if (b3[z,z3]<>b3[z,z3-1]) AND (b3[z,z3]<>b3[z,z3-1]) AND(b3[z,z3]<>'0')
              then write(f2, 'V',z3,' ')
            else if (b3[z,z3]=b3[z,z3-1]) AND (b3[z,z3]=b3[z,z3-1]) AND(b3[z,z3]<>'0')
              then write(f2, 'V',z3,' ')
            else if (b3[z,z3]='0') then write(f2, 'V',z3,' ')
          end
        else if (z3=n) then
          if (b3[z,z3]=b3[z,z3-1]) AND (b3[z,z3]<>'0')
            then write(f2, 'V',z3,')')
          else if (b3[z,z3]<>b3[z,z3-1]) AND (b3[z,z3]<>'0')
            then write(f2, 'V',z3)
          else if (b3[z,z3]='0')
            then write(f2, 'V',z3);

      end;
    writeln(f2);
    for z3:=1 to i do
      begin
        for z4:=1 to j do
          begin
            m1[z3,z4]:=m5[z3,z4];
            write(f2,m5[z3,z4]);
```

```
      end;
    writeln(f2);
  end;
kol := 1;
kol1:=2;
for z3:= 2 to n do
  begin
    if (b3[z,z3]<>b3[z,z3-1])OR(b3[z,z3]='0') then
      begin
        inc(kol); inc(kol1);
      end;
    if (z3<n)AND(b3[z,z3]=b3[z,z3-1])AND(b3[z,z3]=b3[z,z3+1])AND(b3[z,z3]<>'0') then inc(kol1);
    if (z3<n) then
      if (b3[z,z3]=b3[z,z3-1])AND(b3[z,z3]<>b3[z,z3+1])AND(b3[z,z3]<>'0') then
        begin
          if kol1<=(n/2) then kol2 := trunc((kol+kol1)/2)+1
          else kol2 := trunc((kol+kol1)/2);
          writeln(f2,'{V',kol,' - V',kol1,'} = {',kol,' - ',kol1,'} -> {',kol2,'}');
          for z4:=1 to i do
              for z5:=1 to j do
                  if (m1[z4,z5]>=kol)AND(m1[z4,z5]<=kol1) then
                    m1[z4,z5]:=kol2;
          kol := kol1;
          inc(kol1);
        end;
    if (z3=n) then
      if (b3[z,z3]=b3[z,z3-1])AND(b3[z,z3]<>'0') then
        begin
          if kol1<=(n/2) then kol2 := trunc((kol+kol1)/2)+1
          else kol2 := trunc((kol+kol1)/2);
          writeln(f2,'{V',kol,' - V',kol1,'} = {',kol,' - ',kol1,'} -> {',kol2,'}');
          for z4:=1 to i do
              for z5:=1 to j do
                  if (m1[z4,z5]>=kol)AND(m1[z4,z5]<=kol1) then
                    m1[z4,z5]:=kol2;
          kol := kol1;
        end;
  end;
for z3:=1 to i do
  begin
    for z4:=1 to j do
      begin
        write(f2,m1[z3,z4]);
      end;
    writeln(f2);
  end;

//Entropies

for k:=1 to i+1 do lev[k]:=1000;
for k:=1 to e*i do a[k]:='0'; //запис всіх чисел Грея,як "0"
for k:=1 to j do m2[k]:=1; //задання всіх частот,як "1"
k:=0;k2:=0;
REPEAT
  for k4:=1 to i do if a[k4]='1' then k5:=k5+1;  //підрахунок к-сті "1"
  if (k5>0) then
    begin
      for k4:=1 to i do
        begin
          for k6:=0 to e-1 do
            begin
              //підрахунок к-сті "1" у коді Грея відносно маски
              if (a[i*k6+k4]='1') then k7:=k7+1;
```

```
      end;
     if (k7>0) then k8:=k8+1;
     k7:=0;
   end;
 //перевірка умови знаходження хоча б однієї "1" у стовпчику
 //(не закресленого елемента)
 if k8=i then
   begin
      for k6:=1 to j do m2[k6]:=1;
      count:=0;count1:=0;i2:=1;j2:=1;H:=0;
      k2:=k2+1;
      writeln(f2,'Your mask № ',k2,':'); //запис фрази у файл
      for k6:=1 to e do write(f2,' -');
      writeln(f2);
      //запис у файл "гарного" виду маски
      for k6:=i downto 1 do
        begin
          for k9:=e-1 downto 0 do if a[i*k9+k6]='0' then write(f2,'|X') else write(f2,'| ');
          write(f2,'|');
          writeln(f2);
          for k9:=1 to e do write(f2,' -');
          writeln(f2);
        end;
      //запис у файл назви стовпчиків таблиць
      for k6:=i*e downto 1 do if a[k6]='1' then begin write(f2,' S[',e*i-k6+1,']');count:=count+1;end;
      write(f2,' | f[B](c)'); writeln(f2);
      for k6:=1 to j-e+1 do
        begin
          for k9:=1 to e do
            for k10:=1 to i do
              //перевірка на "0" або "1" у послідовності a
              if a[e*i-(k9*i-i-1+k10)]='1' then
                begin
                  //запис вибіркового числа у вихідну с-му(при "1")
                  m3[i2,j2]:=m1[k10,k9+k6-1];
                  j2:=j2+1; //підрахунок к-сті записаних чисел
                end;
          //блок перевірки на спідпадання рядків для підрахунку частоти
          if i2>1 then
            begin
              for k9:=1 to i2-1 do
                begin
                  for k10:=1 to count do if m3[i2,k10]=m3[k9,k10] then k12:=k12+1;
                  if k12=count then
                    begin
                      //враховуємо співпадання рядків
                      bool:=1;
                      m2[k9]:=m2[k9]+1;
                    end;
                  k12:=0;
                end;
            end;
          if bool=0 then  //якщо рядки не співпали
            begin
              i2:=i2+1;
              count1:=count1+1;
              j2:=1;
            end
          else j2:=1; //враховуємо співпадання рядків для частоти
          bool:=0;
        end;
      for k9:=1 to count1 do
        begin
```

```
      for k10:=1 to count do
        begin
          //запис елементів вихідної с-ми у файл
          write(f2,'   ',m3[k9,k10],' ');
        end;
      //запис ймовірності кожного рядка системи
      writeln(f2,' | ',m2[k9]/(j-e+1):1:5);
      //підрахунок Шенноновської ентропії
      H:=H+(m2[k9]/j)*ln(m2[k9]/(j-e+1))/ln(2);
    end;
//врахування знаку "-" у Шенноновській ентропії
H:=-H;
writeln(f2,'H=',H:1:5);
//початок формування нейтральної с-ми (G)
for k6:=1 to j do m2[k6]:=1;
count:=0;count1:=0;i2:=1;j2:=1;H1:=0;
//той самий підрахунок,що й для загальної с-ми(тільки лівої частини
for k6:=i*e downto i+1 do if a[k6]='1' then begin count:=count+1;end;
for k6:=1 to j-e+1 do
  begin
    for k9:=1 to e do
      for k10:=1 to i do
        if a[e*i-(k9*i-i-1+k10)]='1' then
          begin
            m3[i2,j2]:=m1[k10,k9+k6-1];
            j2:=j2+1;
          end;
    if i2>1 then
      begin
        for k9:=1 to i2-1 do
          begin
            for k10:=1 to count do if m3[i2,k10]=m3[k9,k10] then k12:=k12+1;
            if k12=count then
              begin
                bool:=1;
                m2[k9]:=m2[k9]+1;
              end;
            k12:=0;
          end;
      end;
    if bool=0 then
      begin
        i2:=i2+1;
        count1:=count1+1;
        j2:=1;
      end
    else j2:=1;
    bool:=0;
  end;
//підрахунок ентропії для нейтральної с-ми
if count>0 then
for k9:=1 to count1 do
  begin
    H1:=H1+(m2[k9]/j)*ln(m2[k9]/(j-e+1))/ln(2);
  end;
H1:=-H1;
writeln(f2,'H(G`)=',H1:1:5);
//підрахунок ентропії для нейтральної с-ми
H1:=H-H1;
writeln(f2,'H(G|G`)=',H1:1:5);
k10:=0;
for k9:=e*i downto i do if a[k9]='0' then inc(k10);
//перевірка ентропії на мінімальність
```

```
                if H1<lev[i*e-i-k10+1] then lev[i*e-i-k10+1]:=H1;
                for k6:=1 to j do m2[k6]:=1;
                count:=0;count1:=0;i2:=1;j2:=1;H2:=0;
              end;
            k8:=0;
          end;
        k5:=0;
        k:=k+1;
        k1:=1;
        k3:=k;
        //зміна чисел у коді Грея
        while k3-2*trunc(k3/2)=0 do
          begin
            k3:=k3/2;
            inc(k1);
          end;
        if k1<=i*e then
          case a[k1] of
            '0': a[k1]:='1';
            '1': a[k1]:='0'
          end
      UNTIL k1>e*i;
      writeln(f2,lev);
      for z3 := 1 to j do write(f2,'-');
        writeln(f2);
    end;

   writeln(f2,all);
  end;
 end;
close(f2);
writeln('Job`s done!');
end.
```