Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет комп'ютерної інженерії та управління
(повна назва)

Кафедра електронних обчислювальних машин
(повна назва)

АТЕСТАЦІЙНА РОБОТА

Пояснювальна записка

Рівень вищої освіти другий (магістерський)

Клітинно-автоматне моделювання динаміки реструктурованих мережевих структур

(тема)

Виконав:
студент II курсу, групи СПм-19-1
Гук А.С.
(прізвище, ініціали)

Спеціальність
123 – Комп'ютерна інженерія
(код і повна назва спеціальності)

Тип програми освітньо-професійна
(освітньо-професійна або освітньо-наукова)

Освітня програма
Системне програмування
(повна назва освітньої програми)

Керівник: проф. Міхаль О.П.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри ЕОМ _____ Коваленко А.А.
(підпис) (прізвище, ініціали)

2020 р.

Харківський національний університет радіоелектроніки

Факультет комп'ютерної інженерії та управління

Кафедра електронних обчислювальних машин

Рівень вищої освіти другий (магістерський)

Спеціальність 123 – Комп'ютерна інженерія
(код і повна назва)

Тип програми освітньо-професійна
(освітньо-професійна або освітньо-наукова)

Освітня програма Системне програмування
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
"____" _____ 20___ р.

ЗАВДАННЯ

НА АТЕСТАЦІЙНУ РОБОТУ

студентові Гуку Артему Сергійовичу
(прізвище, ім'я, по батькові)

1. Тема роботи Клітинно-автоматне моделювання динаміки реструктурованих мережевих структур

затверджена наказом по університету від "30" жовтня 2020 р. № 1486 Ст

2. Термін подання студентом роботи до екзаменаційної комісії 14 грудня 2020 р.

3. Вхідні дані до роботи
Клітинні автомати
Реструктурованість
Мережева структура
Python

4. Перелік питань, що потрібно опрацювати в роботі
Аналіз предметної області
Функції сусідства в теорії клітинних автоматів
Моделюваннння реструктурованих мережевих структури

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (слайдів) 13 арк. ф. А4

6. Консультанти розділів роботи (заповнюється за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Аналіз завдання	06.11.2020-14.11.2020	
2	Аналіз науково-технічної літератури	15.11.2020-21.11.2020	
3	Пошук існуючиих моделей	22.11.2020-27.11.2020	
4	Пошук аналогічних моделей комп'ютерних	28.11.2020-05.12.2020	
	мереж		
5	Імітаційне моделювання	06.12.2020-07.12.2020	
6	Оформлення пояснювальної записки	08.12.2020-10.12.2020	
7	Оформлення графічної частини	11.12.2020-13.12.2020	

Дата видачі завдання 3 листопада 2020 р.

Студент _____
(підпис)

Керівник роботи _____ проф. Міхаль О.П.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка атестаційної роботи: 82 с., 20 рис., 3 табл., 4 дод., 7 джерел.

КЛІТИННІ АВТОМАТИ, ДИНАМІКА, РЕСТРУКТУРОВАНІСТЬ, МЕРЕЖЕВА СРУКТУРА, РЕСУРС, ПРОТОКОЛ.

Дана робота присвячена клітинно-автоматному моделюванню динаміки реструктурованих мережевих структур.

Підтримка функціонування мережевої структури в умовах впливу різних зовнішніх факторів передбачає одиночність або множинність, регулярність та унікальність в часі окремих впливів зовнішнього середовища на мережеву структуру. В умовах наявності таких впливів структура функціонально видозмінюється. Мета видозмін – підтримання функціональності структури. Динаміка реструктуризації визначається динамікою впливів зовнішнього середовища. Дані фактори взаємопов'язані і, зрозуміло, специфічні для різних предметних областей. Але загальний характер взаємозв'язків і взаємозалежностей – єдиний і, в цілому, відповідає представленому опису. У зв'язку з цим актуальна проблематика моделювання мережевих структур в аспекті їх реструктурування.

ABSTRACT

Master’s thesis: 82 pages, 20 figures, 3 tables, 4 appendices, 7 sources.

CELLULAR MACHINES, DYNAMICS, RESTRUCTURING, NETWORK STRUCTURE, RESOURCE, PROTOCOL.

This work is devoted to cell-automatic modeling of the dynamics of restructured network structures. Supporting the functioning of the network structure under the influence of various external factors involves loneliness or plurality, regularity and uniqueness in time of individual environmental influences on the network structure. In the presence of such influences, the structure changes functionally. The purpose of the changes is to maintain the functionality of the structure. The dynamics of restructuring is determined by the dynamics of environmental influences. These factors are interrelated and, of course, specific to different subject areas. But the general nature of the relationships and interdependencies – the only one and, in general, corresponds to the presented description. In this regard, the topical issue of modeling network structures in terms of their restructuring.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

КА – клітинні автомати

КАП – клітинно-автоматний підхід

ВСТУП

Актуальність проблеми. В даний час більшість обчислювальних задач вимагає для свого рішення ресурсів, переважаючих можливості однопроцесорних комп'ютерів. З'являється все більше алгоритмів, що використовують паралельні або розподілені обчислення для вирішення завдань. Крім цього, виникають нові завдання, які вимагають паралельної або розподіленої архітектури для свого рішення.

Беручи до уваги ту обставину, що зараз при розробці складних програмно-апаратних комплексів значна частка витрат йде на розробоку програмного забезпечення і ця частка неухильно зростає, завдання створення інструментального засобу для автоматизації проектування програмного забезпечення систем з паралельною і розподіленої обчислювальними архітектурами стає актуальною.

Існує ряд математичних моделей паралельних і розподілених обчислень. Однією з них є клітинні автомати, застосування яких і присвячена ця магістерська робота

Клітинні автомати – це дискретні динамічні системи, поведінку яких може бути повністю описано в термінах локальних залежностей. Ці автомати є моделі паралельних обчислень, які володіють як завгодно великим ступенем паралелізму. Можна вважати, що вони являють собою дискретний еквівалент поняття «поле». Клітинні автомати застосовуються для проведення різних обчислювальних експериментів, так як вони зручні, наприклад, для чисельного рішення диференціальних рівнянь і рівнянь в приватних похідних. Вони також широко використовуються для моделювання поведінки складних систем.

Існує ряд засобів автоматизації проектування програмного забезпечення обчислювальних експериментів з використанням клітинних автоматів таких, як, наприклад, засіб SIMP/STEP (реалізуючи набір функцій

для вирішення завдань фізичного моделювання), CAGE (інструмент для освітніх цілей) або Mirek's Cellebration (MCell) (засіб для візуалізації) і CDL (інструмент, що дозволяє використовувати FPGA системи для виконання обчислювальних експерементів). Однак всі ці засоби мають обмеження, які накладаються на предметні області вирішуваних завдань, що реалізують клітинні автомати або використовують обчислювальні архітектури. Не існує єдиного засобу, придатного, як для освітніх, так і для наукових цілей. Тому для автоматизації проектування програмного забезпечення обчислювальних експериментів на основі клітинних автоматів необхідно розробити універсальний, розширюваний інструментальний засіб, що володіє широким спектром функціональних можливостей, так як існуючі інструменти не дозволяють вирішувати різноманітні задачі з різних предметних областей за допомогою широкого спектра клітинних автоматів. Так, наприклад, далеко не кожен засіб дозволяє моделювати автомати, як з двовимірними, так і з тривимірними гратами. Мала їх частина підтримує використання різноманітних околиця.

Метою атестаційної роботи є дослідження моделей і алгоритмів клітинних автоматів в контексті реструктурованих мережевих структур.

Завдання:

- розглянути базові принципи організації клітинної структури об'єктів живої природи;
- розробити комп'ютерну модель ресурсно-залежних клітинних автоматів з використанням мови Python;
- провести аналіз реструктурованих мережевих структури;
- провести імітаційне моделювання.

1 МОДЕЛІ КЛІТИННИХ АВТОМАТІВ

У цьому розділі представлено загальний опис історії розвитку теорії клітинних автоматів. У ній вводяться основні поняття цієї теорії, а також математичний апарат і формалізм. Доводиться коректність затверджувальних властивостей і вхідних понять. Пропонується математична база для подальших міркувань. Крім того, наводяться й аналізуються вимоги до інструментальних засобів автоматизації проектування програмного забезпечення обчислювальних експериментів з використанням клітинних автоматів. Перераховуються і описуються більше двадцяти існуючих продуктів, розроблених для цих цілей, аргументується потреба в розширенні цього списку, наводяться основні властивості, можливості і переваги використання схожих програмних засобів.

1.1 Введення в історію та ідеологію клітинних автоматів

В кінці 60-х років XX століття клітинні автомати були необхідні для подальшого прогресу фізики, біології, хімії, математики, комп'ютерних наук та інших галузей знань. Вони багато разів винаходили під різними назвами, в безлічі країн, людьми, які працюють в різних галузях науки. В результаті, терміни «ітеративні масиви», «обчислюючі простори», «однорідні структури» і «клітинні автомати» практично є синонімами.

Коли Станіслав Улам (Stanislaw Ulam) почав свої дослідження, він навряд чи міг припустити, що, грунтуючись на його ідеях, видатний американський вчений Джон фон Нейман (John von Neumann) введе поняття «клітинний автомат».

Незважаючи на те, що фон Нейман був математиком і фізиком, ідея прийшла до нього під час побудови пояснення певних об'єктів біології. Він використовував клітинні автомати для створення більш правдоподібних

моделей просторово-протяжних систем. Дійсно, механізми, які були запропоновані їм для отримання самовідновлюючих структур на клітинному автоматі, сильно нагадують відкриті в наступному десятилітті закономірності, які спостерігаються в біологічних системах.

Результати досліджень фон Неймана наведені, зокрема, в фундаментальній праці. Перше її видання з'явилося в 1966 році, в той час як автор помер в 1957 році. Книгу дописував його учень, Артур Беркс (Arthur Burks), що став відомим фахівцем в області клітинних автоматів.

В кінці Другої світової війни, в той час як фон Нейман створював один з перших електронних комп'ютерів, німецький інженер Конрад Цузе (Konrad Zuse) запропонував ряд ідей, які зробили б його одним з основоположників теорії паралельних обчислень, якби відразу стали широко відомими.

Серед іншого Цузе придумав «обчислюючі простори» – клітинні автомати. Особливий його інтерес викликало застосування цих систем до завдань чисельного моделювання в механіці. На жаль, ситуація в світі завадила роботам вченого набути поширення, в той час як за працями фон Неймана стежив увесь науковий світ.

Математики прийшли до клітинних автоматів, розглядаючи ітераційні перетворення просторово-розподілених структур з дискретним набором станів. Так з'явилися «ітераційні масиви». Відразу стали виникати вирішення важливих теоретичних завдань в цій області, наприклад, питань оборотності, обчислюваності, досяжності, тощо. У групі комп'ютерної логіки університету штату Мічиган Джон Холланд (John Holland) вперше застосував клітинні автомати для вирішення завдань адаптації та оптимізації.

На жаль, недостатність спілкування і єдиної термінології вели до значного дублювання робіт. Так важливі характерні особливості клітинних автоматів, доведені Річардсон (Richardson) на двадцяти сторінках в континуальної підстановці в топології канторівських множин, могли бути записані в двох рядках у вигляді слідства попередньої роботи Хедлунда (Hedlund).

Поняття «однорідні структури» надзвичайно близько до «клітинних автоматів» за своїм походженням і позначає їх апаратну реалізацію, в якій ключовою властивістю цих систем є їх гомогенність.

Спектр застосування клітинних автоматів надзвичайно широкий. Ефект вибуху бомби справила стаття провідної рубрики математичних ігор і головоломок журналу «Scientific American» Мартіна Гарднера (Martin Gardner). Він опублікував опис клітинного автомата Джона Хортона Конвея (John Horton Conway) «Життя». Гра «Життя» стала культовою і зробила поняття «клітинний автомат» невід'ємною частиною побутового жаргону людей з технічною освітою.

Клітинний автомат – це якийсь світ, який живе за певними законами. Його еволюція, як життя будь-якої замкнутої системи, безцільна. З плином часу змінюється стан решітки, але система сама по собі не може помітити цей процес, не може робити висновки, зрозуміти, навіщо це потрібно і як довго буде тривати. Розвиток світу набуває сенсу, коли з'являється спостерігач, який дивиться на нього з іншого світу і аналізує процес еволюції. Таким чином, розглянуті моделі служать для вивчення еволюційних процесів.

При вирішенні задач моделювання решітка стає простором для подання деякого середовища або речовини. Нехай на рисунку 1.а схематично зображено якусь речовину. Зони, що володіють однаковим значенням деякого істотного параметра, виділені одним і тим же кольором. Припустимо, що при вирішенні завдання ніякі інші характеристики речовини не істотні.

Для чисельного моделювання необхідно описати речовину для обчислювальної машини, визначити просторовий розподіл значень параметра. Для цього слід провести дискретизацію, замінити реальний набір даних кінцевим числом інформативних значень.

Наведемо приклади дискретизації речовини, зображеної на рисунку 1.а за допомогою квадратної і гексагональної грат. Для цього накладемо їх на

речовину (рис. 1.b і 1.d, відповідно). У кожну клітинку помістимо значення, відповідне тій зоні матеріалу, яку вона охоплює. Якщо це - якийсь числовий параметр, то клітці можна привласнити його середнє значення по всіх точках, охоплених кліткою, значення з її центру або, як в даному прикладі, значення, яким володіє більшість точок (рис. 1.c і 1.e, відповідно). Використовуються й інші способи визначення значення, яке слід помістити в клітку, що відповідає даній області середовища.

Клітинні автомати утворюють загальну парадигму паралельних обчислень подібно до того, як машини Тьюринга утворюють парадигму послідовних обчислень. В результаті вони можуть бути використані для реалізації паралельних алгоритмів, як моделі, що володіють природним паралелізмом.

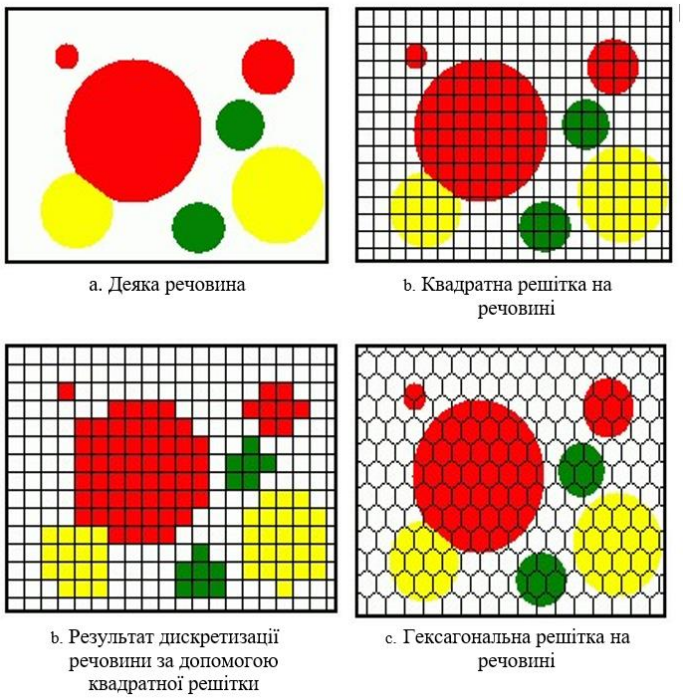

Рисунок 1.1 – Приклад дискретизації елемента простору

Одне з головних відмінностей клітинної системи від інших обчислювальних систем полягає в тому, що у всіх інших присутні дві принципово різні частини:

- архітектурна (керуюча), яка фіксована і активна (тобто виконує деякі операції)
- дані, які змінні і пасивні (тобто самі по собі вони нічого зробити не можуть).

У клітинних автоматів обидві ці складові збираються з принципово ізоморфних, не відрізняючихся один від одного елементів. Таким чином, обчислювальна система може оперувати своєю матеріальною частиною, модифікувати, розширювати себе і будувати собі подібних. Хоча системи цього класу і були придумані Джоном фон Нейманом, така паралельна архітектура отримала назву «не-фон-неймановской», так як «фон-неймановскую» архітектуру він створив раніше.

Дане твердження може здатися суперечливим. Здавалося б, до архітектурної частини логічніше віднести, наприклад, грати і правила автомата. Однак це скоріше апаратна частина. Тут мається на увазі те, що, наприклад, при розгляді автомата, що реалізує гру «Життя», при даних решітці і правилах, змінюючи лише стан клітин, можна реалізувати універсальну обчислювальну систему, що дозволяє проводити будь-які обчислення. За своїми виразними здібностями така модель буде еквівалентна довільним машинам Тьюринга і клітинним автоматам. При цьому клітини, що описують архітектуру моделюється обчислювальної системи, не будуть відрізнятися від клітин, що представляють собою дані.

Клітинний автомат, званий комп'ютером Бенкса, також має властивість обчислювальної універсальності. Використовувати його вкрай неефективно, але з теоретичної точки зору факт його існування - важливий результат.

1.2 Визначення клітинного автомата

Формально клітинний автомат *A* являє собою четвірку об'єктів {G, Z, N, f}, де:

G – дискретний метричний простір, решітка автомата, набір його

клітин. Наявність метрики дозволяє забезпечити те, щоб всі клітини знаходилися на кінцевій відстані від даної, так як ні для якої метрики ніякі дві точки дискретного метричного простору не можуть бути нескінченно віддалені один від одного. Крім того, поняття метрики і координат клітини виявляється надзвичайно корисним при визначенні безлічі клітин околиці N, описуваної далі. Як правило, простір G є одно-, дво- або тривимірним, проте він може мати і велику розмірність.

Z – кінцева безліч можливих станів клітини. Таким чином, стан всієї решітки є елементом множини Z|G|.

N – кінцева безліч, що визначає околицю клітини – тобто |N| клітин, які впливають на новий стан даної. Його можна задавати, наприклад, як безліч відносних зсувів з клітки, необхідних для досягнення навколишніх осередків. Елементи цієї множини будемо називати «сусідами»1 клітини.

f – функція переходів. Обчислювана функція, яка діє $Z \times Z|N| \rightarrow Z$ для автоматів з клітинами з пам'яттю і $Z|N| \rightarrow Z$ для автоматів з клітинами без пам'яті. Відмінність полягає в тому, що в функції переходів клітинних автоматів з клітинами з пам'яттю поточногу стану клітини також передається як параметр.

Крок автомата полягає в застосуванні функції переходів до кожної клітини решітки. Він вважається завершеним тільки після того, як новий стан буде визначено для всіх елементів.

Клітинні автомати в загальному випадку характеризуються такими фундаментальними властивостями:

Решітка однорідна - ніякі дві області не можуть бути відрізнені один від одного по ландшафту. Це забезпечується семантикою простору G, а також єдинством визначення околиці N, безлічі значень Z і функції переходів f.

Взаємодії локальні. Стан клітин околиці впливає на стан даної клітини, що забезпечується семантикою функції f.

Оновлення значень станів всіх клітин решітки відбувається одночасно після обчислення нового стану кожної з них.

Необхідно відзначити, що при вирішенні деяких завдань виникає необхідність відмовитися від виконання цих властивостей «класичних» клітинних автоматів.

Автомати, для яких не виконується третя з них, називаються асинхронними клітинними автоматами.

1.3 Теорема про три двовимірні решітки з правильних багатокутників

На практиці часто використовуються клітинні автомати з двовимірними решітками з правильних багатокутників. Автомати саме з такими решітками будуть переважно обговорюватися в цій роботі. На рисунках 1.2 – 1.4 показані фрагменти трикутної, квадратної і шестикутної (гексагональної) решіток[2] , відповідно.

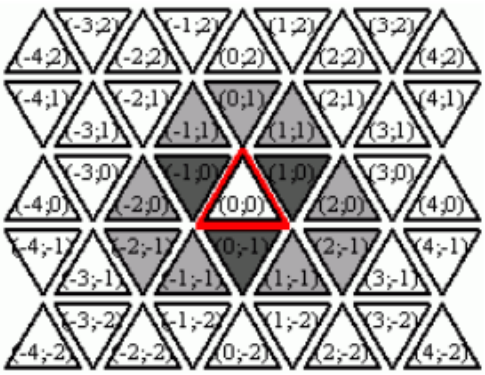

Рисунок 1.2 – Фрагмент трикутної решітки

Терміни «квадратна решітка» , «решітка квадратів» та «решітка з квадратів» (аналогічно для інших багатокутників) будемо вважати синонімами.

У більшості випадків, як околиці клітини використовують підмножина набору її безпосередніх сусідів.

Околиця, що складається з клітинок, що мають спільну сторону з даної, називається околицею фон Неймана або множиною головних сусідів. На рисунках 1.2-1.4 елементи такої околиці для центральної клітинки виділені темно-сірим кольором.

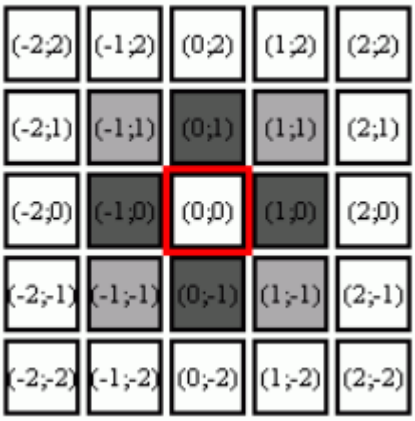

Рисунок 1.3 – Фрагмент квадратної решітки

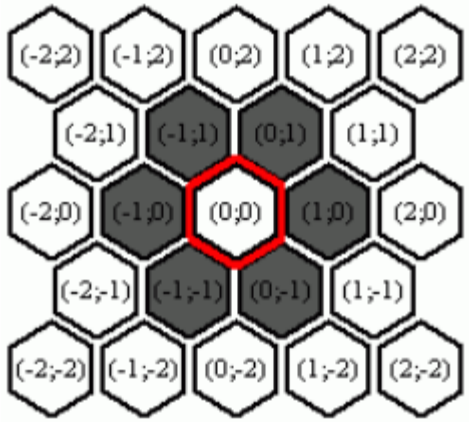

Рисунок 1.4 – Фрагмент шестикутної решітки

Околиця, що складається з клітинок, що мають хоча б загальну вершину з даною, називається околицею Мура або множиною безпосередніх сусідів. На рисунках 1.2-1.4 елементи такої околиці для центральної клітинки виділені світло-сірим або темно-сірим кольорами.

Зрозуміло, що множина безпосередніх сусідів будь-якої клітини містить або дорівнює множині її головних сусідів.

Нехай a – елемент множини G ,клітинка решітки. Множина безпосередніх сусідів клітинки a будемо позначати ,як S0(a),а множину сусідів клітинки a , знайдене відповідно до визначення околиці N - як N (a) (в загальному випадку воно може не збігатися ні з S (a), ні з S0(a)).

Доведемо наступну теорему.

Теорема 1. Про три двумірні решітки з правильних багатокутників.

Не існує іншої решітки з правильних багатокутників, крім трикутної, квадратної і шестикутної.

В даному випадку мається на увазі саме регулярне зіставлення однакових клітинок так, щоб вони мали спільні сторони або вершини.

Доведення:

Сума кутів правильного n-кутника A (n) = π · (n -) 2. Тоді $\frac{A(n)}{n}$ - величина кожного кута n-кутника.

Нехай з правильних n-кутників вдалося скласти грати. Тоді повний кут в 2π радіан утворюють кути цілого числа фігур.

Позначимо це число, як k.

Таким чином, k багатокутників можна скласти так, щоб вони мали загальну вершину. Визначимо k, як функцію від n:

$$2\pi = k\cdot\frac{A(n)}{n} \Leftrightarrow = 2\pi = k\cdot\frac{\pi\cdot(n-2)}{n} \Rightarrow k(n) = \frac{2n}{n-2}$$

Будемо розглядати цю функцію при n ≥ 3, так як трикутник - багатокутник з найменшою кількістю вершин. Візьмемо похідну від k (n) по n:

$$k'(n) = -\frac{n}{(n-2)^2}$$

При n ≥ 3 функція k (n) убуває, так як k’(n) <0. Отже, все можливі значення k менше k (3) = 6. Відзначимо також, що k (n)> 2, так як істинно наступне нерівність:

$$k(n) > 2 \Leftrightarrow \frac{2n}{n-2} > 2 \Leftrightarrow 2n > 2n - 4 \Leftrightarrow 0 > -4$$

Решітку можна побудувати, якщо цілому *n* буде відповідати ціле *k*. З викладеного вище випливає, що можливі лише чотири значення *k* - від трьох до шести. Побудуємо функцію *n (k),* зворотню до k (n), і перевіримо, яким з можливих значень *k* відповідає ціле *n*:

$$k(n) = \frac{2n}{n-2} \Leftrightarrow k\cdot(n-2) = 2n \Leftrightarrow n\cdot(k-2) = 2k \Rightarrow n(k) = \frac{2k}{k-2}$$

Маємо:

k = 3 ⇒ n (k) = 6 - гексагональна решітка.

k = 4 ⇒ n (k) = 4 - квадратна решітка.

$k = 5 \Rightarrow n(k) = \frac{10}{3}$ - не ціле n, ґрати не побудувати.

k = 6 ⇒ n (k) = 3 - трикутні грати.

Таким чином, дійсно існують тільки три перераховані вище двовимірні решітки з правильних багатокутників.

1.4 Резервування ресурсів у програмному забезпеченні реального часу

Введемо поняття кільця для даної клітини решітки. Клітинка є елементом (сусідом) нульового кільця для самої себе. Її сусіди - елементами першого кільця. Далі - рекурсивно: для даної клітинки елементами і-го кільця (кільця і-го порядку) називаються всі клітинки, які є сусідами будь-якої клітини її (і-1) -го кільця, виключаючи такі з них, які самі належать кільцям менших порядків.

Формально, якщо позначити через R (a, i) множину клітинок і-го кільця (I ≥ 0) для клітинки a, то наведене вище визначення можна записати так:

R (a) 0, = {a}

R (a i), = {b| ∃c ∈ R (a, i - 1) : b ∈ N (c), ¬∃j <I : b ∈ R (a, j)} (1.1)

Також має місце наступне співвідношення:

$$|R\ (a\ ,i)| \leq |N|^i,\ i \geq 0 \quad (1.2)$$

Припустимо, що в якості околиці клітинки буде використовуватися безліч її безпосередніх сусідів - для будь-якого елементу а вірна рівність N (a) = S (a). Поняття кільця для всіх трьох розглянутих решіток в цьому випадку ілюструють рисунках 1.5-1.7. Суміжні кільця для центральної клітинки виділені різними кольорами.

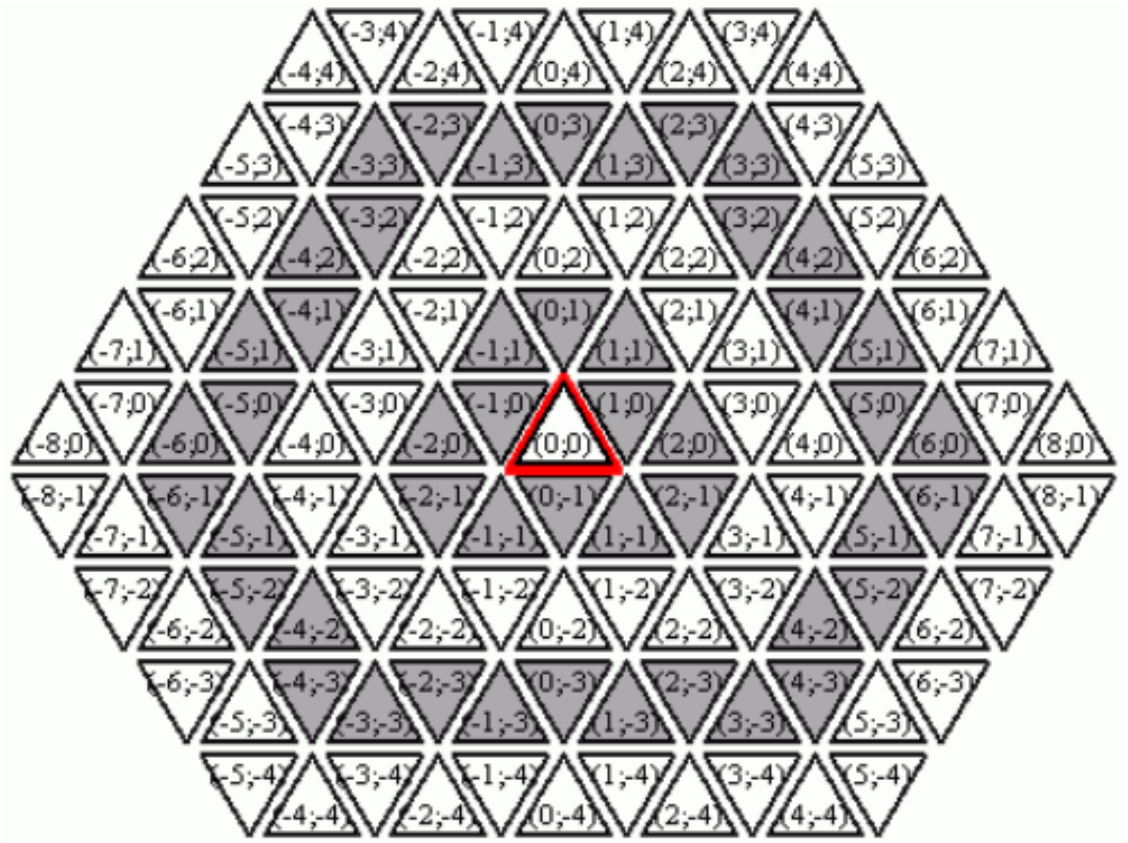

Рисунок 1.5 – Кільця для трикутної решітки

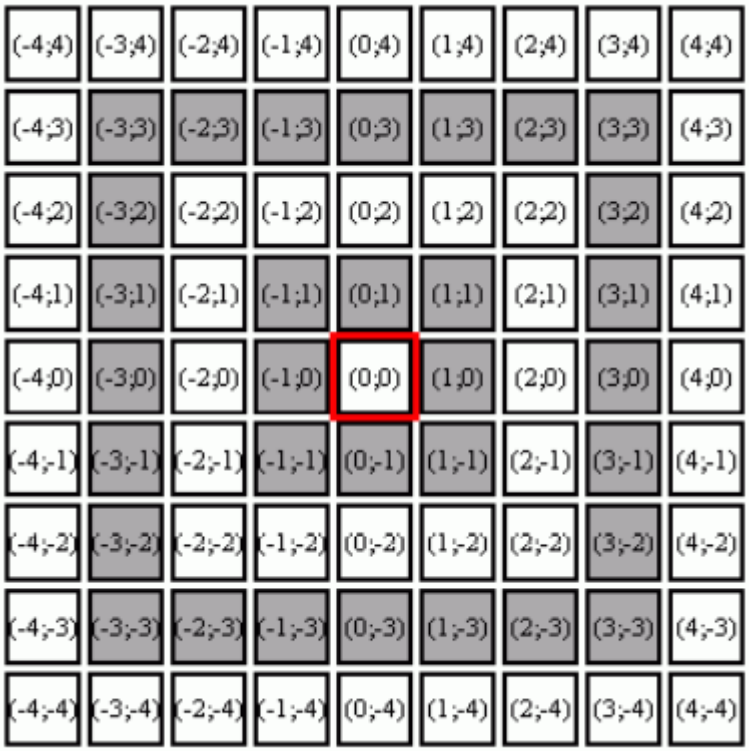

Рисунок 1.6 – Кільця для квадратної решітки

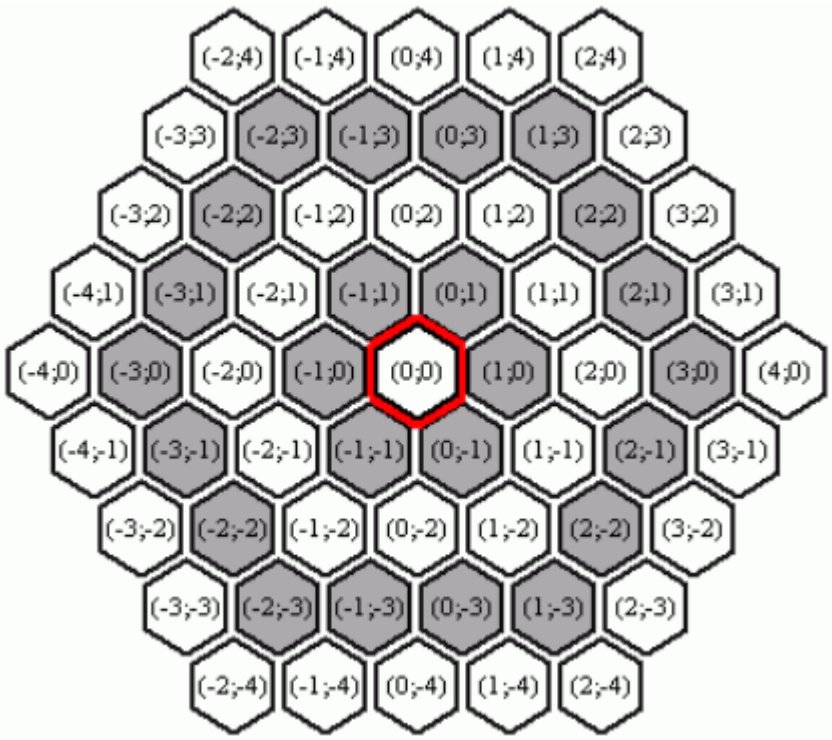

Рисунок 1.7 – Кільця для шестикутної решітки

Далі, якщо для будь-яких двох клітин a й b виконання співвідношення a∈N (b) тягне виконання співвідношення b∈N (a), то будемо називати таку околиця симетричною. Необхідно відзначити, що околиця, що складається з безпосередніх або головних сусідів клітини, є симетричною.

При вирішенні більшості завдань розглядаються симетричні околиці. Далі в міркуваннях будемо вважати околиці симетричними.

Доведемо ряд теорем, які стверджують основну властивість поняття кільця.

Теорема 2. Про існування і єдиність індексу кільця.

Для будь-яких двох клітин a і b, існує одне і єдине невід'ємне число і таке, що b∈R (a, i).

Доведення:

Рисунко 1.7. Кільця для шестикутної решітки

Існування. Нехай не існує такого i, щоб b∈R (a, i). Тоді і для будь-якого з сусідів клітини b, наприклад, клітинки b1, не існує такого індексу i1, щоб b1∈R (a, i1). Для будь-якого з сусідів клітинки b1, наприклад, клітинки b2, також не знайдеться індекс i2, щоб b2∈R (a, i2).

Продовжуючи ці міркування і враховуючи симетричність відносин сусідства, приходимо до висновку, що в цьому випадку решітка буде розділена, як мінімум, на дві підмножини клітинок так, що жодна з клітинок однієї підмножини НЕ буде сусідкою ніякої з клітин іншої підмножини, що суперечить умові однорідності решітки.

Єдиність. Нехай значення індексу не єдине й існують i1 і i2, такі, що b∈R (a, i1) і b∈R (a, i2). Це означає, що:

$$\exists c1 \in R\,(a, i1 - 1): b \in N(c1), \neg\exists j < i1 : b \in R(a, j)$$

$$\exists c2 \in R\,(a, i2 - 1): b \in N(c2)\ \neg\exists j < i2 : b \in R\,(a, j)$$

З першого твердження випливає, що i_2 не може бути більше i_1, а з другого - що i_1 не може бути більше i_2. Таким чином, i_1 і i_2 рівні.

Теорема 3. Про симетрію відносин приналежності і кільцю.

Для будь-яких двох клітин *a* і *b*, твердження $b \in R\ (a,\ i)$ і $a \in R\ (b,\ i)$ еквівалентні.

Доведення:

Доведемо твердження методом математичної індукції по i:

База. Нехай i = 0. Тоді, якщо b∈R (a, 0), то a і b збігаються і, отже, a∈R (b, 0). Аналогічно проводяться міркування в іншу сторону. Істинність твердження для i = 1 випливає з припущення симетричності відносин сусідства.

Перехід. Нехай для i = j-1 і i = j твердження вірне. І, зокрема, a∈R (bj-1, j-1) ⇔ bj-1∈R (a, j-1) і a∈R (bj , j) ⇔ bj ∈R (a, j). Доведемо для i = j + 1, що a∈R (bj + 1, j + 1) ⇔ bj + 1∈R (a, j + 1).

Доведемо, що з твердження a∈R (bj + 1, j + 1) випливає, що bj + 1 ∈ R (a, j + 1). Дійсно, якщо a∈R (bj + 1 ,j + 1), то, за формулою (1) існує клітинка c, така, що a∈N (c), c∈R (bj + 1 , j). Крім того, a∉R (bj+1, k), де k <j + 1 в силу теореми 2.

За індукційному припущенням c∈R (bj + 1, j) ⇔ bj + 1 ∈ R (c, j). За припущенням симетричності околиці a∈N (c) ⇔ c∈N (a). Внаслідок, можна стверджувати, що bj + 1∈ R (a, j + 1).

Доведемо, що із твердження bj+1 ∈R (a, j + 1) випливає, що a∈R (bj + 1, j + 1). Дійсно, якщо bj + 1 ∈R (a, j + 1), то, за формулою (1) існує клітинка c, така, що bj + 1∈N (c), c∈R (a, j). Крім того, bj + 1 ∉R (a, k), де k < j + 1.

За індукційному припущенням c∈R (a, j) ⇔ a∈R (c, j). За припущенням симетричності околиці bj + 1 ∈N (c) ⇔ c∈N (bj + 1). Внаслідок, можна стверджувати, що a∈R (bj + 1, j + 1).

Таким чином, дійсно bj + 1∈R (a, j + 1) ⇔ a∈R (bj+1, j + 1).

1.5 Метрика

Введемо метрику для простору *G*. Це можна зробити безліччю способів. Як приклад виберемо наступний: відстанню *D (a, b)* від клітини *a* до клітинки *b* будемо називати номер кільця клітинки *a* (або *b*), якому належить клітинка *b* (або *a*). Формально, вираз матиме вигляд:

$$D (a, b) = i: a \in R (b\ i),$$

$$D (a, b) = i: b \in R (a\ i), \quad (1.3)$$

Доведемо, що введена таким чином функція є метрикою.

Теорема 4. Про метрику.

Функція D (a, b), визначена формулою (3) є метрикою.

Доведення:

Перевіримо три основні властивості метрики.

D (a, b) невід'ємно визначена, так як за визначенням індекс кільця невід'ємний.

D (a, b) = D (b, a) - вірно, так як $b \in R (a, i) \Leftrightarrow a \in R (b, i)$ за теоремою 3.

Доведемо правило трикутника, тобто той факт, що $D (a, b) \leq D (a, c) + D (c, b)$, методом математичної індукції по D (c, b):

База. Якщо D (c, b) = 0, то клітинки b і c збігаються і D (a, b) = D (a, c) +0.

Перехід. Нехай для всіх клітинок b і ci таких, що D (ci, B) = i, і будь-яких a, вірне твердження $D (a, b) \leq D (a, ci) + D (ci , B)$. Доведемо, що для всіх клітинок b і cc+1 таких, що D (ci+1, b) = i + 1, і будь-яких a, вірне твердження $D (a, b) \leq D (a, ci + 1) + D (ci + 1, b)$.

Виберемо клітинку ci , яка є сусідньою з клітинкою ci + 1. Внаслідок значення D (a, ci + 1) буде перебувати між D (a, ci) - 1 і D (a,ci) +1. Перевіримо нерівність $D (a, b) \leq D (a, ci + 1) + D (ci+1, b)$ для всіх трьох

варіантів, беручи до уваги, що D (a, b) ≤D (a, ci) + i за індукційному припущенням:

D (a, ci + 1) = D (a, ci) -1 ⇒ D (a, b) ≤D (a, ci) -1+ (i + 1) = D (a, ci) + i - вірно.

D (a, ci + 1) = D (a, ci) ⇒ D (a, b) ≤D (a, ci) + (i + 1) - вірно.

D (a, ci + 1) = D (a, ci) +1 ⇒ D (a, b) ≤D (a, ci) +1+ (i + 1) = D (a, ci) + (i + 2) – вірно.

Таким чином, індукційне припущення доведено і функція D (a, b), визначена формулою (3) є метрикою.

Поняття кільця і відстані можуть бути узагальнені і поширені на випадок решіток довільної розмірності.

Необхідно відзначити, що питання про метриці для решітки клітинного автомата не піднімалося іншими авторами. Тим не менш, вона неявно присутня при вирішенні переважної більшості завдань хоча б тому, що околиця клітинок, зазвичай описується в термінах відносних зсувів з даної.

Для однієї і тієї ж метрики, в залежності від завдання, координати можна ввести різними способами. Найпростіший з них – декартові координати, при використанні яких кожна клітина характеризується двома (у випадку двовимірної решітки) цілими числами x і y.

1.6. Теорема про еквівалентність клітинного автомату набору кінцевих автоматів

Той факт, що клітинні автомати є моделлю паралельних обчислень, дозволяє зробити припущення, що вони представляють собою безліч моделей послідовних обчислень, наприклад, кінцевих автоматів.

Введемо визначення кінцевих автоматів Мілі і Мура.

Кінцевий автомат Мілі є п'ятірка об'єктів A = {Z, X, Y, f, g}, де

Z – кінцева множина станів автомата.

X - кінцева множина вхідних впливів на автомат.

Y – кінцева множина вихідних впливів автомата.

f – функція переходів автомата, $Z \times X \rightarrow Z$.

g – функція виходів автомата, $Z \times X \rightarrow Y$.

Кінцевим автоматом Мура є п'ятірка об'єктів A = {Z, X, Y, f, g}, що відрізняється від автомата Мілі лише тим, що функція виходів g діє таким чином: $X \rightarrow Y$.

Поняття «функція» в двох останніх визначеннях може бути замінено поняттям «обчислювана програма».

Доведемо теорему, яка стверджує, що клітинний автомат еквівалентний не більше ніж рахунковому набору однакових кінцевих автоматів Мілі.

Теорема 5. Теорема про еквівалентність набору кінцевих автоматів.

Клітинний автомат A = {G, N, Z, f} може бути реалізований за допомогою безлічі кінцевих автоматів Мілі {Ai}, де Ai = {ZA, XA, YA, fA, gA}, де $1 \leq i \leq G$.

Доведення:

Наведемо формальний конструктивний метод побудови безлічі кінцевих автоматів по клітинному автомату.

Нехай дано клітинний автомат A = {G, N, Z, f}. Розглянемо побудову набору кінцевих автоматів Мілі {Ai}, де Ai = {ZA, XA, YA, fA, gA}, що реалізує функціональність еквівалентну функціональності автомата A.

Кожен кінцевий автомат буде виконувати функції однієї клітини. При цьому в ньому повинні бути дві внутрішні змінні. Назвемо їх «новий стан» (він буде зберігати нове значення змінної стану, обчислене для наступного кроку) і «значення визначено» (логічний прапор, значення якого дорівнює одиниці, якщо «новий стан» вже було обчислено і нулю в іншому випадку), спочатку рівне нулю.

Наведемо п'ять складових компонентів автомата Мілі. Тут і далі в даному доведенні символом «⊥» будемо позначати, так званий, «сигнал поновлення», яким обмінюються автомати для синхронізації.

ZA = Z. Змінна, що зберігає стан кінцевого автомата має ті ж безліч значень, що і для клітини клітинного автомата. Значення змінної «новий стан», а також булевского прапора «значення визначено» не слід включати в стан автомата в цілому, так як обмін ними з сусідніми автоматами не повинен проводитися.

XA = Z | N | ∪ {⊥}. Через вхідні впливу автомат отримує інформацію про стан сусідів. Крім того, до безлічі вхідних впливів необхідно додати сигнал поновлення для забезпечення одноразовості присвоєння клітинам нових значень.

YA = Z∪ {⊥}. Через вихідні впливу автомат повідомляє сусідам свій стан або пересилає сигнал оновлення.

fA повинна реалізовувати наступну функціональність – якщо вхідний вплив на автомат не дорівнює сигналу поновлення, то fi повинна викликати для відповідної клітини функцію переходів автомата f (t) (де t∈XA для клітинного автомата з клітинами без пам'яті і t∈XA × Z для клітинного автомата з клітинами з пам'яттю). Отримане значення слід зберегти у внутрішній змінній «новий стан». Змінній «значення визначено» при цьому має бути присвоєно значення одиниці. Результат, що повертає при цьому функцію fi має дорівнювати номеру поточного стану автомата. Якщо вхідний вплив на автомат рівен сигналу поновлення, то функція fi повинна повернути значення внутрішньої змінної «новий стан», змінивши, тим самим, свій стан.

gA повинна реалізовувати таку функціональність. Якщо вхідний вплив на автомат не дорівнює сигналу поновлення то результат, що повертається функцією має дорівнювати номеру поточного стану автомата. Якщо вхідний вплив на автомат рівен сигналу поновлення, то поведінка визначається станом прапора «значення визначено». Якщо прапорець встановлений (дорівнює одиниці), то функція повинна повернути значення, відповідне сигналу поновлення. Якщо він не встановлений, то повертаємий функцією результат повинен бути рівний номеру поточного стану автомата.

По завершенні кожної ітерації автомати повинні отримати сигнали відновлення, для чого достатньо подати цей сигнал на вхід одного з них.

Необхідно відзначити, що для асинхронних автоматів потреба в сигналах поновлення відпадає. В силу того, що питання перетворення кінцевих автоматів Мілі в кінцеві автомати Мура розглядалися неодноразово, окремо розглядати еквівалентність клітинного автомата набору автоматів Мура не має сенсу.

1.7 Апаратні та програмні реалізації клітинних автоматів

Клітинний автомат - модель, що володіє природним паралелізмом і для того, щоб отримати якісь переваги від її використання, необхідно застосовувати паралельне апаратне або програмне забезпечення.

Використання спеціалізованого обладнання виглядає досить привабливим, тому що при вирішенні задач за допомогою клітинних автоматів потрібен великий обсяг пам'яті для зберігання станів решітки. Однак взаємодія змінних, відповідних клітин, локальна, що може бути вигідно використано при апаратній реалізації.

Крім того, при проведенні експериментів за допомогою клітинних автоматів, необхідно виконувати величезну кількість ітерацій. Припустимо, для отримання задовільних результатів при вирішенні прикладної задачі потрібно зробити близько 1015 кроків. За надзвичайно оптимістичною оцінкою, при моделюванні клітинного автомата на персональному комп'ютері з архітектурою *iX86*, ітерація може зайняти кілька мікросекунд. Тоді експеримент займе тисячоліття. Спеціалізоване апаратне забезпечення дозволить зробити ці показники більш реальними.

Найвідоміший зразок такого обладнання було розроблено групою інформаційної механіки Массачусетського технологічного інституту і називається CAM (Cellular Automata Machine - машина клітинних автоматів).

Досить популярною була шоста версія цього продукту, CAM-6, що є PCI-пристроєм, що підключається до комп'ютера за архітектурою iX86 під управлінням операційної системи PC-DOS. Він використовує відеосистему персонального комп'ютера для візуалізації стану решітки автомата, а також його електроживлення, дискову пам'ять і т.д. В обмін машина надає свої обчислювальні можливості. Симбіоз виходить надзвичайно вигідним. Застосуванню CAM-6 присвячена робота.

Остання, на даний момент, версія цього пристрою, CAM-8, працює в тандемі з комп'ютерами сімейства Sun workstation за допомогою інтерфейсу SBus. На комп'ютері має бути запущено надане з пристроєм програмне забезпечення STEP.

Для програмування цих пристроїв використовується інтерпретована мова Forth, розширена і модифікована для підтримки відповідних примітивів.

Для програмування CAM-8 на низькому рівні розроблена бібліотека StepLib для мови C.

Однак апаратні рішення вимагають капіталовкладень, чіткого усвідомлення потреб розв'язуваної задачі і використовуються, переважно, в спеціалізованих лабораторіях. Інструментальні засоби автоматизації проектування програмного забезпечення обчислювальних експериментів за допомогою клітинних автоматів дозволяють легко перетворити офіс або комп'ютерний клас університету в обчислювальний центр. Співвідношення ціна/якість для них виглядає в даний час найбільш перспективним, так як можливості персональних комп'ютерів невблаганно ростуть.

Це ілюструється тим фактом, що група вчених під керівництвом Томазо Тоффолі (Tommaso Toffoli), яка створила CAM, спочатку працювала над інструментальним засобом для моделювання автоматів. Потім з нього «виросло» згадане вище апаратне рішення. Однак зараз ця група знову працює над програмним засобом моделювання (проект носить ім'я SIMP / STEP і описується в підрозділі 1.9), а розробка CAM припинена.

1.8 Вимоги до засобів автоматизації проектування програмного забезпечення обчислювальних експериментів за допомогою клітинних автоматів

Перерахуємо вимоги до засобів автоматизації проектування програмного забезпечення обчислювальних експериментів за допомогою клітинних автоматів. Наведені нижче характеристики не є обов'язковими, проте, така функціональність дозволить проекту претендувати на те, щоб бути універсальним середовищем для клітинних обчислень. Інструментальний засіб, якщо має можливість, повинен:

- Надавати користувачам дружній, зручний, сучасний інтерфейс з багатим набором можливостей для роботи.
- Наочно візуалізувати стан решітки, забезпечувати зручну навігацію по ній.
- Бути максимально універсальним, що не накладати обмежень на безліч автоматів, які можна використовувати.
- Виконувати ітерації, використовуючи сучасні обчислювальні та комунікаційні технології, швидко і ефективно распараллелювати обчислення автомата за допомогою доступних ресурсів (машин обчислювального кластера, процесорів багатопроцесорної системи і т.д.).

Оптимізувати обчислення, наприклад, вилучати не оновлювані зони решітки з розгляду, використовувати шаблони для порівняння околиць щоб уникнути зайвих трудомістких обчислень і т.п.

Дозволяти зручно і надійно проводити тривалі (можливо, багатомісячні) обчислювальні експерименти. Незважаючи на те, що майже всі обчислювальні ресурси системи будуть зайняті здійсненням ітерацій, програма повинна продовжувати бути інтерактивною і відповідати на запити користувача.

Дозволяти проводити не тільки самі обчислення, але й аналіз результатів обчислень, а саме, підтримувати побудову різноманітних

графіків, демонструвати зміни різних параметрів модельованої системи і т.п.

1.9 Огляд існуючих засобів автоматизації проектування програмного забезпечення обчислювальних експериментів за допомогою клітинних автоматів

У цьому розділі наводиться список з 21 найбільш популярних і вдалих засобів автоматизації проектування програмного забезпечення обчислювальних експериментів за допомогою клітинних автоматів. Для кожного продукту вказується наступна інформація:

- назва – імена авторів;
- умови розповсюдження;
- вимоги до апаратного забезпечення, а також операційної системи;
- короткий коментар.

Далі наводиться список продуктів в алфавітному порядку.

1. CAGE. Автори: Д. Трунфіо (D. Trunfio) та інші.

Поширюється вільно, проте офіційний реліз ще не випущений, продукт не готовий.

Робоча станція під управлінням операційної системи сімейства Windows або Windows NT.

«Cellular Automata General Environment» - один з найновіших продуктів. Програма спочатку орієнтована на використання в освітніх цілях. Не підтримує розподілених або паралельних обчислень. Володіє надзвичайно багатим інтерфейсом. Однак очевидно, спектр його можливостей настільки широкий, що його доводиться визнати складним для розуміння. Ця обставина в поєднанні з тим, що документація до проекту доступна тільки на італійській мові, створює істотну перешкоду для його застосування. Функції переходів задаються С-подібною мовою і компілюються. Однією з переваг даного засобу є те, що він дозволяє використовувати нерегулярні решітки.

2. CAM Simulator. Автори: З. Белзо (Z. Belso) и М. Варджиас (M. Vargyas).

Поширюється вільно.

Робоча станція під управлінням операційної системи MS-DOS.

Програма представляє собою симулятор пристрою CAM-6 (Cellular Automata Machine), від якого програма успадкувала ряд істотних обмежень (використання тільки квадратної решітки та околиці, що складається лише з безпосередніх сусідів і т.п.).

3. CAMEL. Автори: Д. Талія (D. Talia) и Дж. Спеззано (G. Spezzano).

Вільно не поширюється, можна отримати тільки версію для однопроцесорної машини, що позбавляє сенсу застосування цього засобу для реальних завдань.

Робоча станція під управлінням операційної системи UNIX / Linux або обчислювальний кластер, який використовує бібліотеку MPICH, що реалізує стандарт MPI.

За іронією долі, цей проект має практично ту ж назву, що і проект, що описується в даній роботі (амперсант в назві останнього з'явився саме в зв'язку з існуванням цього проекту). Однак, в даному випадку слово «CAMEL» означає «Cellular Automata environMent for systEms modeLing». Для опису автоматів, визначающих решітку, завдання функції переходів, околиці та іншого використовується спеціально розроблена авторами мова CARPET (CellulAR Programming EnvironmenT). Програма з цієї мови транслюється в програму на мові С з використанням бібліотеки MPICH і виконується середовищем CAMEL. Підтримуються тільки квадратні і не більше ніж тривимірні решітки. Проект досить старий. Інтерфейс реалізований за допомогою бібліотеки Motif.

4. CANL. Автори: Р. Наполітано (R. Napolitano), К. ді Наполі (C. Di Napoli).

Вільно не поширюється.

Платформа Cray YMP або Sun workstation.

Програма підтримує автомати з двовимірними квадратними гратами і декартовими метриками. В околиці можуть бути присутніми тільки безпосередні сусіди.

5. CAPow. Автор: Р. Ракер (R. Rucker).

Поширюється вільно.

Робоча станція під управлінням з операційної системи сімейства Windows або Windows NT.

Проект з дуже багатим інструментарієм для візуалізації результатів. Програма підтримує автомати з одно- і двовимірними гратами і декартовими метриками. Правила для автомата описуються на мові C ++ з використанням призначеної для цього бібліотеки. Зібрані в динамічну бібліотеку правила виконуються середовищем. Однак засіб не підтримує розподілених або паралельних обчислень.

6. CASim. Автор: С. Колер (S. Kohler).

Поширюється вільно.

Робоча станція під управлінням операційної системи на основі ОС UNIX / Linux з встановленою бібліотекою X11 / Motif.

Програма дозволяє моделювати автомати з двовимірними гратами і декартовими метриками.

7. CAT/CARP. Автори: И. Ясеняк (I. Jaceniak), С. Фоке (S. Focke), У. Лінк (U. Link).

Поширюється вільно.

Робоча станція під управлінням операційної системи сімейства Windows, Windows NT або OS / 2. Програма підтримує автомати з двовимірними квадратними гратами і декартовими метриками.

8. CD. Автор: К. Хошбергер (C. Hochberger).

Поширюється вільно.

Робоча станція під управлінням операційної системи сімейства UNIX / Linux, а також обчислювальний кластер або платформа FPGA сімейства CEPRA.

Програма підтримує автомати з двовимірними квадратними гратами і декартовими метриками. Правила задаються на мові С і компілюються

9. CDM/SLANG. Автор: Х. Зібург (H. Sieburg).

Поширюється вільно.

Робоча станція Apple Macintosh.

Програма підтримує автомати з двовимірними гратами і декартовими метриками.

10. CelLab. Автори: Р. Ракер (R. Rucker) и Дж. Уолкер (J. Walker).

Поширюється вільно.

Робоча станція під управлінням операційної системи сімейства Windows.

Правила пишуться на мові С і компілюються.

11. CELLAS/FUNDEF. Автор: Т. Легенді (T. Legendi).

Свободно не поширюється.

Робоча станція під управлінням операційної системи MS-DOS.

Програма підтримує автомати з двовимірними гратами і декартовими метриками.

12. Cellsim. Автори: К. Ленгтон (C. Langton) и Д. Хібелер (D. Hiebeler).

Поширюється вільно.

Робоча станція під управлінням операційної системи сімейства UNIX / Linux або станція Connection Machine CM-2.

Програма підтримує автомати з одно- і двовимірними квадратними гратами і декартовими метриками, а також околиці радіусом не більше трьох. Програма розроблялася в Університеті Сантафе (США), одному з лідерів в області дослідження клітинних автоматів.

13. Cellular/Cellang. Автор: Дж. Дана Еккарт (J. Dana Eckart).

Поширюється вільно.

Робоча станція Sun workstation або SGI з операційною системою сімейства UNIX / Linux. Також підтримується робота під управління операційної системи сімейства MS-DOS.

Програма підтримує автомати з довільними квадратними, кубічними, гіперкубічними гратами і декартовими метриками.

14. CEPROL. Автор: F. Зойтер (F. Seutter).

Умови поширення невідомі

Робоча станція під управлінням операційної системи сімейства UNIX / Linux.

Програма підтримує автомати з двовимірними гратами і декартовими метриками. Правила пишуться на мові Pascal і компілюються.

15. DDLab. Автор: К. Ленгтон (A. Wuensche).

Поширюється вільно.

Робоча станція Sun workstation, Apple Macintosh або станція під управлінням операційної системи на основі ОС MS-DOS.

Вельми багатий за своїми можливостями проект, який розроблявся в Університеті Сантафе.

Не підтримує розподілених або паралельних обчислень.

16. HICAL. Автори: А. Бекерс (A. Beckers), Т. Уорш (T. Worsch) та інші.

Поширюється вільно.

Робоча станція під управлінням операційної системи сімейства UNIX / Linux з системою X11.

Інструментальний засіб з досить широкими можливостями. Підтримує автомати з одно- і двовимірними квадратними гратами і декартовими метриками. Дозволяє використовувати різні локальні правила, а також реалізовувати межавтоматні взаємодії і використовувати структуровані значення станів клітин. Зручний інструментарій для спостереження за експериментами.

17. LCAU. Автор: Х. В. Макінтош (H. V. Macintosh).

Поширюється вільно.

Робоча станція під управлінням операційної системи MS-DOS або станція NeXTSTEP.

Програма надає якісно нові і важливі засоби дослідження клітинних автоматів. Зокрема, з її допомогою можна знаходити цикли в зміні стану решітки, будувати діаграми де Брюна, і визначати, з певною ймовірністю, попередній стан решітки. Однак продукт не підтримує розподілені або паралельні обчислення.

18. Mirek's Cellebration (MCell). Автор: М. Вийтовиц (M. Wyjtowicz).

Поширюється вільно.

Робоча станція під управлінням операційної системи сімейства Windows.

Незважаючи на значний термін з моменту релізу (остання версія датована 2001 роком) це - одна з найбільш популярних програм розглянутого класу. Хоча програма не підтримує засобів для паралельних або розподільних обчислень і дозволяє використовувати тільки квадратні решітки, багатство набору прикладів робить її ідеальною для візуалізації еволюції станів. Однак засоби визначення функції переходів незручні.

19. SCARLET. Автори: М. Кутриб (M. Kutrib) та інші.

Умови розповсюдження невідомі.

Робоча станція під управлінням операційної системи Sun Solaris 2.x.

Програма підтримує автомати з довільними гратами і декартовими метриками. В якості станів клітин можуть бути використані цілі числа і рядки.

20. SIMP/STEP. Автори: Т. Тоффолі (T. Toffoli), Т. Бах (T. Bach).

Поширюється вільно з вихідним кодом.

Робоча станція під управлінням операційної системи сімейства UNIX / Linux.

Даний продукт розробляється під керівництвом одного з авторів апаратної платформи для моделювання клітинних автоматів, пристрою CAM (Cellular Automata Machine), який очолює зараз «лабораторію програмованої матерії» в Бостонському університеті. Програма дозволяє описувати, форматувати, генерувати, візуалізувати і аналізувати експерименти на

клітинних автоматах. Підтримує тільки існуючі картезіанські метрики. Функції переходів описуються за допомогою інтерпретованої мови Python.

21. WinCA. Автори: Б. Фіш (B. Fisch) и Д. Гріффіт (D. Griffeath).

Поширюється вільно.

Робоча станція під управлінням операційної системи сімейства Windows або Windows NT.

Програма підтримує автомати з двовимірними гратами і декартовими метриками. Не підтримує розподілені або паралельні обчислення.

На рисунку 1.8 знаками «+» і «-» показано підтримує той чи інший продукт кожну з семи вимог до інструментальних засобів обговорюваного класу, перерахованих в даному підрозділі.

Таким чином, жоден з перерахованих вище інструментальних засобів автоматизації проектування програмного забезпечення обчислювальних експериментів з використанням клітинних автоматів не тільки не відповідає всім вимогам, але навіть не підтримує більше трьох з них. Особливо погано стоїть справа з такими властивостями, як універсальність (третя властивість), підтримка технологій паралельних і розподілених обчислень (четверта властивість), оптимізація (п'ята властивість) і аналіз (сьома властивість), без виконання яких важко уявити собі інструментальний засіб для наукових обчислень.

1.10. Компонентна модель декомпозиції клітинних автоматів

Компонент, який реалізує правила автомата, декларує також список аналізованих параметрів, що характеризують роботу системи в цілому. Середовище повинно дозволяти спостерігати зміну їх значень в динаміці, будувати графіки, файли звітів і т. д. Таким чином, на аналіз експерименту не буде витрачати додатковий обчислювальний час.

Для аналізу таких параметрів повинен бути передбачений п'ятий тип компонентів – аналізатор (analyzer).

Назва	1	2	3	4	5	6	7
CAGE	+	+	–	–	–	+	–
CAM Simulator	–	+	–	–	–	+	–
CAMEL	–	–	+	+	–	+	–
CANL	+	–	–	+	–	+	–
CAPow	+	+	–	–	–	+	–
CASim	–	–	–	–	–	+	–
CAT/CARP	–	–	–	–	–	+	–
CDL	–	–	–	+	–	+	–
CDM/SLANG	–	–	–	–	–	+	–
CelLab	–	–	–	+	–	+	–
CELLAS/FUNDEF	–	–	–	–	–	+	–
Cellsim	+	+	–	–	–	+	–
Cellular/Cellang	+	–	+	–	–	+	–
CEPROL	–	–	–	–	–	+	–
DDLab	+	+	–	–	–	+	–
HICAL	+	–	–	–	–	+	+
LCAU	–	+	–	–	–	+	+
Mirek's Cellebration	+	+	–	–	–	+	–
SCARLET	+	+	–	–	–	+	–
SIMP/STEP	–	–	–	+	+	+	–
WinCA	+	+	–	–	–	–	–

Рисунок 1.8 – Наявність тих чи інших властивостей у обговорюваних інструментальних засобах

Реалізація метрики у вигляді окремого компонента дозволяє використовувати нестандартні системи координат, як, наприклад, узагальнені координати [Error! Bookmark not defined].

1.11 Висновки до розділу 1

Введено термін «клітинний автомат», а також супутні поняття і математичний апарат.

Сформульовано вимоги до інструментальних засобів автоматизації проектування програмного забезпечення обчислювальних експериментів з використанням клітинних автоматів.

Показано, що жодне з розглянутих засобів зазначеного класу не задовольняє навіть більш ніж трьом з семи сформульованих вимог.

Запропонована компонентна модель декомпозиції клітинних автоматів для зручності їх реалізації.

Введено основні поняття і описані базові властивості розробленого інструментального засобу, що задовольняє всім вимогам.

2 ФУНКЦІЇ СУСІДСТВА ПРИ МОДЕЛЮВАННІ КЛІТИННИХ АВТОМАТІВ

У цьому розділі розробляється метод введення узагальнених координат для решіток клітинних автоматів, що дозволяє забезпечити універсальний підхід до зберігання інформації про стани клітин для різних решіток. Узагальнена координата - невід'ємне ціле число, однозначно вказує на клітку решітки клітинного автомата довільної розмірності.

Пропоновані координати мають ряд важливих властивостей. Введення узагальнених координат фактично являє собою нумерацію невід'ємними цілими числами елементів багатовимірного дискретного простору. Метод ілюструється набором прикладів для двовимірних клітинних автоматів, для яких запроваджуються:

- спіральні узагальнені координати для решіток з трикутників;
- спіральні узагальнені координати для решіток з квадратів;
- спіральні узагальнені координати для решіток з шестикутників;
- координати для решіток з трикутників, що базуються на спіральних узагальнених координатах для решіток з шестикутників;
- узагальнені координати для решіток з квадратів, засновані на кривих Пеано (Peano).

Можливість введення цих та інших різновидів координат підтримується розробленим в магістерській роботі інструментальним засобом. Обговорювані в цьому розділі приклади введення узагальнених координат були випробувані при приведенні обчислювальних експериментів.

2.1 Основні вимоги та особливості моделі резервування ресурсів

На рисунках 1.1 – 1.7 в кожній клітині вказана пара декартових координат. Ідея узагальнених координат полягає в тому, щоб пронумерувати

всі клітини решітки невід'ємними цілими числами так, щоб для кожного номера існувала клітина, і для кожної клітини існував номер, з точністю до обмеженості розмірів решітки. Ці номери будуть називатися узагальненими координатами клітин.

Одній з клітин присвоюється номер нуль, вона приймається за початок координат. Номери інших клітин характеризують їх положення щодо неї.

У загальному випадку введення узагальнених координат полягає в побудові взаємооднозначного відображення з безлічі невід'ємних цілих чисел у множину цілих чисел в ступеня k. У цьому розділі будуть розглянуті різні відображення для k = 2, однак запропоновані методи можуть бути застосовані і для решіток більшої розмірності.

Для того щоб при здійсненні ітерацій не довелося перетворювати узагальнені координати в декартові, і назад, необхідний математичний апарат, що дозволяє для кожної клітини знаходити всіх її сусідів, не переходячи в іншу систему координат.

В силу локальності взаємодій для цього достатньо забезпечити перебування лише безпосередніх сусідів. У разі більш складної околиці, якщо вона описана в термінах відносних зсувів, її елементи можна буде знайти за допомогою рекурсії.

Подальше обговорення присвячено тому, як забезпечити перебування безпосередніх сусідів при використанні певних способів введення узагальнених координат для всіх трьох можливих двовимірних решіток з правильних багатокутників. Першою буде розглянута квадратна решітка, яка володіє найпростішим кільцем і найбільш часто використовувана на практиці. Далі будуть розглянуті шестикутна і трикутна решітки. Потім буде запропонований ще один метод введення узагальнених координат для трикутної решітки, що забезпечує більш ефективне знаходження безпосередніх сусідів клітини. У висновку буде запропоновано ввести координати на решітці з квадратів згідно кривої Пеано.

Той факт, що відповідно до запропонованого в попередньому розділі методом компонентної декомпозиції автоматів, в розглянутому інструментальному засобі метрика виділяється в окремий, незалежний компонент, дозволяє використовувати нестандартні системи координат в середовищі CAME & L. Обговорювані далі приклади введення узагальнених координат були реалізовані в ній.

2.2 Спіральні узагальнені координати для квадратної решітки

Введемо так звані «спіральні» узагальнені координати для квадратної решітки. Дамо деякій, клітці координату нуль. Далі пронумеруємо елементи її першого кільця за годинниковою стрілкою, починаючи з лівого-верхнього кута. Потім пронумеруємо також клітини другого кільця, третього і т.д. Результат показаний на рис. 8.

Суміжні кільця виділені білим або світло-сірим кольорами, проте поверх цього виділення темно-сірим кольором показані кути кілець.

Порівняємо кожному з чотирьох кутів індекс від нуля до трьох за годинниковою стрілкою, починаючи з лівого-верхнього кута. Індекси також наведені на малюнку.

Лінія, що з'єднує клітини решітки в порядку, визначеному цією нумерацією, являє собою спіраль. Тому дані узагальнені координати і названі «спіральними». Перш ніж розглянути метод знаходження безпосередніх сусідів будь-якої клітини при цьому способі введення координат необхідно визначити кілька допоміжних функцій.

Перша з них - довжина сторони n-го кільця a_n, кількість клітин від одного кута до наступного, включаючи самі кути. Так як $a_0 = 1$, а довжина сторони кожного наступного кільця більше, ніж попереднього на дві клітини, то отримаємо

$$an = 2n + 1 \qquad (2.1)$$

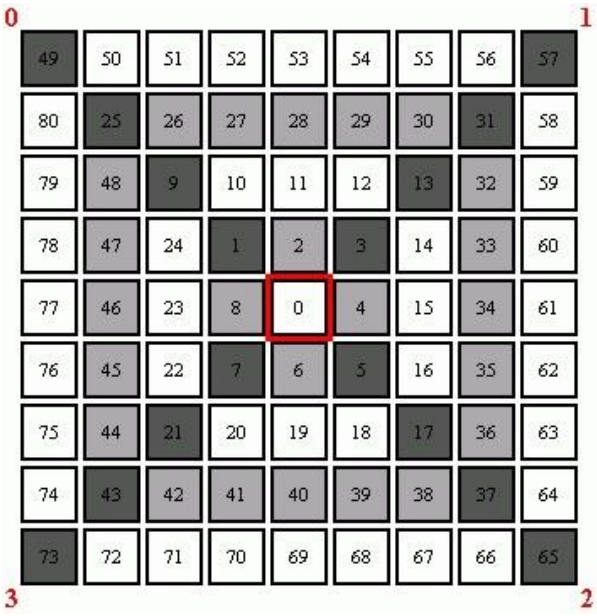

Рисунок 2.1 – Спіральні узагальнені координати для квадратної решітки

Друга функція, Cn, m - координата кута з індексом m для n-го кільця, де m приймає значення від нуля до трьох. Ясно що

$$Cn,0 = a_{n-1}^2 = (2n-1)2 . \quad (2.2)$$

Координати інших трьох кутів можуть бути знайдені додатками величини a_n-1 до координати попереднього. В результаті маємо:

$$C_{n,m} = a_{n-1}^2 + m \cdot (a_n - 1) = (2n-1)^2 + 2m \cdot n. \quad (2.3)$$

Необхідно відзначити, що формула (7), як багато інших виведені далі вирази, не працює при n = 0. Однак, це не суттєво, так як єдина клітина нульового кільця (початок координат) буде розглядатися окремо.

Третьою необхідною величиною є функція n(a) - номер кільця клітини нуль, якому належить клітина a. Вираз для неї виходить, як рішення рівняння a = $C_{n(a),0}$ щодо n(a), в цілих числах. З формули (6) маємо:

$$n(a) = \lceil \sqrt{a+1} \rceil div 2 \qquad (2.4)$$

Переходячи до метода визначення безпосередніх сусідів, необхідно ввести їх індексацію: з лівого-верхнього кута за годинниковою стрілкою, спочатку - по головним сусідам, потім - по іншим. Результат наведено на рисунку 2.1.

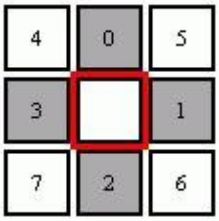

Рисунок 2.2 – Індекси безпосередніх сусідів для квадратної решітки

Через $N_m(a)$ будемо позначати сусіда клітини a з індексом m. Досить забезпечити знаходження головних сусідів, так як інші можуть бути визначені з співвідношень (2.5):

$$\begin{aligned} &N\,4\,(a) = N0\,(N1(a)) = N1(N0\,(a)) \\ &N\,5\,(a) = N1(N2(a)) = N2(N1\,(a)) \\ &N\,6\,(a) = N2(N3(a)) = N3(N2\,(a)) \\ &N7\,(a) = N3\,(N0\,(a)) = N0\,(N3\,(a)) \end{aligned} \qquad (2.5)$$

Обчислення координат сусідів клітини починається з визначення номера кільця, якому вона належить, за формулою (8). Для даної клітини a цей номер n=n(a). Далі потрібно розглянути дев'ять варіантів розташування клітини всередині кільця. Необхідно відзначити, що діапазон значень координат на n-му кільці - від $C_{n,0}$ до $C_{n+1,0}$-1.

У таблиці 2 наводяться формули для знаходження чотирьох головних сусідів клітини (сусідів з індексами від нуля до трьох) в залежності від їх розташування. У стовпці «розташування» вказується конкретна координата клітини, або інтервал, в який вона може потрапити в даному кільці.

Таблиця 2.1 – Функції визначення головних сусідів для спіральних узагальнених координат на квадратній решітці

Розміщення	Індекс головного сусіда			
	0	1	2	3
$C_{n,0}$	$C_{n+1,0}+1$	$a+1$	$C_{n+1,0}-1$	$C_{n+2,0}-1$
$(C_{n,0}; C_{n,1})$	$C_{n+1,0}+1+a-C_{n,0}$	$a+1$	$C_{n-1,0}-1+a-C_{n,0}$	$a-1$
$C_{n,1}$	$C_{n+1,1}-1$	$C_{n+1,1}+1$	$a+1$	$a-1$
$(C_{n,1}; C_{n,2})$	$a-1$	$C_{n+1,1}+1+a-C_{n,1}$	$a+1$	$C_{n-1,1}-1+a-C_{n,1}$
$C_{n,2}$	$a-1$	$C_{n+1,2}-1$	$C_{n+1,2}+1$	$a+1$
$(C_{n,2}; C_{n,3})$	$C_{n-1,2}-1+a-C_{n,2}$	$a-1$	$C_{n+1,2}+1+a-C_{n,2}$	$a+1$
$C_{n,3}$	$a+1$	$a-1$	$C_{n+1,3}-1$	$C_{n+1,3}+1$
$(C_{n,3}; C_{n+1,0}-1)$	$a+1$	$C_{n-1,3}-1+a-C_{n,3}$	$a-1$	$C_{n+1,3}+1+a-C_{n,3}$
$C_{n+1,0}-1$	$C_{n,0}$	$C_{n-1,0}$	$a-1$	$C_{n+2,0}-2$

Окремого розгляду потребує клітина нуль, так як вона є всіма чотирма кутами нульового кільця і її «розташування» визначити неможливо. Тому повинні бути особливо визначені вісім випадків - чотири сусіда нуля і нуль, як сусід чотирьох клітин: N_0 (0) = 2, N_1 (0) = 4, N_2 (0) = 6, N_3 (0) = 8, N_2 (2) = 0, N_3 (4) = 0, N_0 (6) = 0, N_1 (8) = 0.

Використання формул (9) призведе до рекурсивного виклику функцій при реалізації запропонованого методу, однак, якщо не скористатися ними, то число можливих «розташувань» клітин істотно збільшиться, так як для кожного кільця необхідно буде окремо розглядати не тільки кутові клітини, але і суміжні з ними.

2.3 Спіральні узагальнені координати для шестикутної решітки

До шестикутної решітки застосуємо такий же підхід, що й до квадратної. Деякій клітці присвоюється координата нуль. Далі, починаючи з лівого верхнього кута, за годинниковою стрілкою послідовно нумеруються клітини її кілець. Результат показаний на рисунок 2.3.

Суміжні кільця виділені білим або світло-сірим кольорами. Темно-сірі кольором показані кути кілець. Кожному з шести кутів зіставлений індекс від нуля до п'яти за годинниковою стрілкою, починаючи з лівого-верхнього кута.

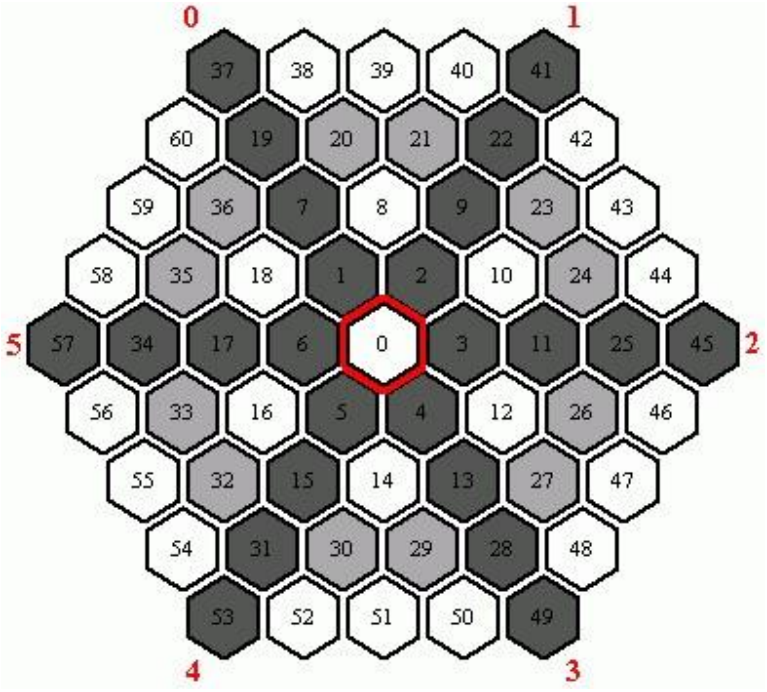

Рисунок 2.3 – Спіральні узагальнені координати для шестикутної решітки

Необхідно визначити для шестикутної решітки ті ж допоміжні функції, які були введені вище для квадратної решітки.

Перша функція – довжина сторони n-го кільця an. Так як $a_0 = 1$, а довжина сторони кожного наступного кільця на одну клітку більше, ніж попереднього, то

$$a_n = n + 1. \quad (2.6)$$

Друга функція, $C_{n,\,m}$ - координата кута з індексом m для n-го кільця, де m приймає значення від нуля до п'яти. З формули (10) випливає, що загальна довжина кожного наступного кільця на шість клітин більше, ніж попереднього. Беручи до уваги той факт, що довжина першого кільця - шість клітин, отримаємо:

$$C_{n,0} = 6E(n-1)+1, \quad (2.7)$$

де

$$E(x) = 1 + 2 + \ldots + (x-1) + x = \sum_{i=1}^{x} i = \frac{x^2+x}{2}. \quad (2.8)$$

Іншими словами, *E (x)* - сума *x* перших натуральних чисел.

Решта кутів можуть бути знайдені додаванням величини a_{n-1} до координати попереднього. В результаті вираз для кута матиме вигляд:

$$C_{n,m} = 6E(n-1) + 1 + m\cdot(a_n - 1) = 3(n^2 - n) + 1 + m\cdot n. \quad (2.9)$$

Третьою функцією є функція n(a).

Вираз для неї виходить, як рішення рівняння [a/6] = E(n (a)) в цілих числах щодо n (a). В результаті маємо:

$$n(a) = \left[\frac{\sqrt{1+8\left[\frac{a}{6}\right]} - 1}{2} \right]. \quad (2.10)$$

Переходячи до методу визначення безпосередніх сусідів, введемо їх індексацію з лівого-верхнього кута за годинниковою стрілкою. Результат наведено на рисунку 2.4.

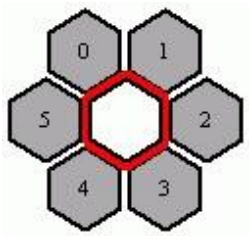

Рисунок 2.5 – Індекси безпосередніх сусідів для шестикутної решітки

Як і в попередньому випадку, через Nm(a) будемо позначати сусіда клітини a з індексом m. У шестикутної решітці все безпосередні сусіди - головні і щоб уникнути невиправданого використання рекурсивних функцій наведемо формули для знаходження кожного з них.

Обчислення координат сусідів клітини починається з визначення номера кільця, якому вона належить, за формулою (2.10). Для даної клітини a цей номер n = n (a). Далі необхідно розглянути тринадцять варіантів розташування клітини всередині кільця.

Як і для квадратної решітки, окремого розгляду вимагає клітина нуль, так як вона є всіма шістьма кутами нульового кільця і її «розташування» визначити неможливо. В силу цього повинні бути особливо визначені дванадцять випадків - шість сусідів нуля і нуль, як сусід шести клітин: N_0 (0) = 1, N_1 (0) = 2, N_2 (0) = 3, N_3 (0) = 4, N_4 (0) = 5, N_5 (0) = 6, N_3 (1) = 0, N_4 (2) = 0, N_5 (3) = 0, N_0 (4) = 0, N_1 (5) = 0, N_2 (6) = 0.

У таблиці 2.2 наводяться формули для знаходження головних сусідів клітини в залежності від її розташування.

Таблиця 2.2 – Функції визначення головних сусідів для спіральних узагальнених координат на шестикутної решітки

Розміщення	Індекс головного сусіда					
	0	*1*	*2*	*3*	*4*	*5*
$C_{n,0}$	$C_{n+1,0}$	$C_{n+1,0}+1$	$a+1$	$C_{n-1,0}$	$C_{n+1,0}-1$	$C_{n+2,0}-1$
$(C_{n,0}; C_{n,1})$	$C_{n+1,0}+a-C_{n,0}$	$C_{n+1,0}+a-C_{n,0}+1$	$a+1$	$C_{n-1,0}+a-C_{n,0}$	$C_{n-1,0}+a-C_{n,0}-1$	$a-1$
$C_{n,1}$	$C_{n+1,1}-1$	$C_{n+1,1}$	$C_{n+1,1}+1$	$a+1$	$C_{n-1,1}$	$a-1$
$(C_{n,1}; C_{n,2})$	$a-1$	$C_{n+1,}1+a-C_{n,1}$	$C_{n+1,1}+a-C_{n,1}+1$	$a+1$	$C_{n-1,1}+a-C_{n,1}$	$C_{n-1,1}+a-C_{n,1}-1$
$C_{n,2}$	$a-1$	$C_{n+1,2}-1$	$C_{n+1,2}$	$C_{n+1,2}+1$	$a+1$	$C_{n-1,2}$
$(C_{n,2}; C_{n,3})$	$C_{n-1,2}+a-C_{n,2}-1$	$a-1$	$C_{n+1,2}+a-C_{n,2}$	$C_{n+1,2}+a-C_{n,2}+1$	$a+1$	$C_{n-1,2}+a-C_{n,2}$
$C_{n,3}$	$C_{n-1,3}$	$a-1$	$C_{n+1,3}-1$	$C_{n+1,3}$	$C_{n+1,3}+1$	$a+1$
$(C_{n,3}; C_{n,4})$	$C_{n-1,}3+a-C_{n,3}$	$C_{n-1,3}+a-C_{n,3}-1$	$a-1$	$C_{n+1,3}+a-C_{n,3}$	$C_{n+1,3}+a-C_{n,3}+1$	$a+1$
$C_{n,4}$	$a+1$	$C_{n-1,4}$	$a-1$	$C_{n+1,4}-1$	$C_{n+1,4}$	$C_{n+1,4}+1$
$(C_{n,4}; C_{n,5})$	$a+1$	$C_{n-1,4}+a-C_{n,4}$	$C_{n-1,4}+a-C_{n,4}-1$	$a-1$	$C_{n+1,4}+a-C_{n,4}$	$C_{n+1,4}+a-C_{n,4}+1$
$C_{n,5}$	$C_{n+1,5}+1$	$a+1$	$C_{n-1,5}$	$a-1$	$C_{n+1,5}-1$	$C_{n+1,5}$
$(C_{n,5}; C_{n+1,0}-1)$	$C_{n+1,5}+a-C_{n,5}+1$	$a+1$	$C_{n-1,5}+a-C_{n,5}$	$C_{n-1,5}+a-C_{n,5}-1$	$a-1$	$C_{n+1,5}+a-C_{n,5}$
$C_{n+1,0}-1$	$C_{n+2,0}-1$	$C_{n,0}$	$C_{n-1,0}$	$C_{n,0}-1$	$a-1$	$C_{n+2,0}-2$

2.4 Спіральні узагальнені координати для трикутної решітки

Для трикутної решітки також можна ввести спіральні узагальнені координати. Деякій клітці присвоюється координата нуль. Далі, починаючи з лівого-верхнього кута, послідовно за годинниковою стрілкою нумеруються

клітини її кілець. Результат показаний на рисунку 2.5.

Суміжні кільця виділені білим або світло-сірим кольорами. Темно-сірим кольором показані кути кілець. Порівняємо кожен з шести кутів індекс від нуля до п'яти, за годинниковою стрілкою, починаючи з лівого-верхнього кута.

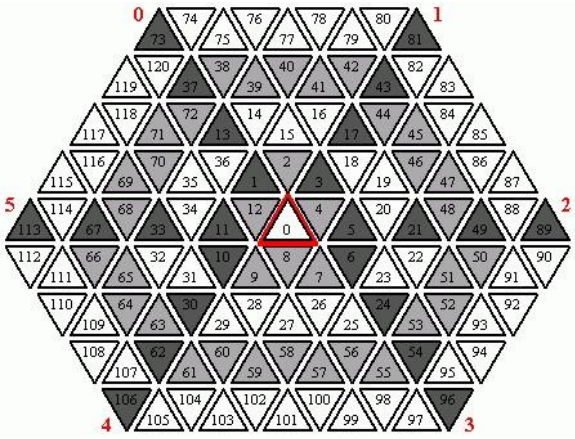

Рисунок 2.5 – Спіральні узагальнені координати для трикутної решітки

Необхідно визначити для трикутної решітки допоміжні функції, описані вище.

Довжина сторони n-го кільця an для решітки трикутників не може бути введена так просто, як для розглянутих вище решіток. У кожного кільця шість сторін і довжини деяких з них різняться. В силу цього необхідно задати довжину ребра n-го кільця, як $a_{n,\ m}$, де m - індекс кута з якого «виходить» сторона при обході за годинниковою стрілкою. Беручи до уваги довжини ребер першого кільця, і з огляду на те, що вони збільшуються на дві клітини при переході до кожного наступного кільця, отримаємо:

$$\begin{cases} a1,0 = 3 \\ a1,1 = 3 \\ a1,2 = 2 \\ a1,3 = 5 \\ a1,4 = 2 \\ a1,5 = 3 \end{cases} \Rightarrow \begin{cases} an,0 = 2n+1 \\ an,1 = 2n+1 \\ an,2 = 2n \\ an,3 = 2n+3 \\ an,4 = 2n \\ an,5 = 2n+1 \end{cases} \qquad (2.11)$$

Останні формули не вірні для нульового кільця, однак це не настільки важливо, адже, як неодноразово зазначалося вище, клітина нуль завжди розглядається окремо, як особливий випадок.

З формул (16) випливає, що загальна довжина кожного наступного кільця більше в порівнянні з попереднім на дванадцять клітин. Беручи до уваги той факт, що довжина першого кільця – дванадцять клітин, отримаємо такий вираз для обчислення $C_{n,0}$, координати нульового кута n-го кільця.

$$C_{n,0} = 12E(n-1) + 1, \qquad (2.11)$$

де E(x) визначається за формулою (2.11).

Координати інших кутів Cn,m можуть бути знайдені послідовними додатками величини an,m-1-1 до координати Cn,m-1-го кута. В результаті маємо такий вираз:

$$C_{n,m} = 12E(n-1) + 1 + \sum_{i=0}^{m-1}(a_{n,i} - 1) = 6(n^2 - n) + 1 + 2m \cdot n + \begin{cases} -1, \text{при } m = 3 \\ 1, \text{при } m = 4 \\ 0, \text{інакше} \end{cases} \qquad (2.12)$$

Третьою необхідною функцією є функція n(a). Вираз для неї виходить, як рішення рівняння [12/a] = E (n(a)) в цілих числах щодо n(a). В результаті, маємо:

$$n(a) = \left[\frac{\sqrt{1 + 8\,[a/12]} - 1}{2} \right] \qquad (2.13)$$

У трикутної решітці зустрічаються клітини двох можливих типів: орієнтовані «вгору» (▲) і «вниз» (▼). При використанні викладеного методу введення узагальнених координат, всі клітини з непарними номерами і нульова клітина будуть орієнтовані «вгору», а інші – «вниз».

Переходячи до методу визначення безпосередніх сусідів, введемо їх індексацію так, щоб вона була зручна для обох варіантів орієнтації. Для клітин, орієнтованих «вгору», пронумеруємо безпосередніх сусідів з лівого-верхнього кута за годинниковою стрілкою спочатку - по головним сусідам, потім - по іншим. Індексація для клітин, орієнтованих «вниз», виходить з цієї розворотом. Результат наведено на рисунку 2.6.

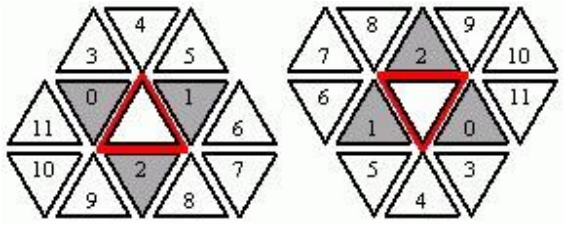

Рисунок 2.6 – Індекси безпосередніх сусідів для трикутної решітки

Знову через Nm (a) будемо позначати сусіда клітини a з індексом m. Досить забезпечити знаходження головних сусідів, так як інші можуть бути знайдені з співвідношень (20):

$$N_3(x) = N_2(N_0(x))$$
$$N_4(x) = N_1(N_3(x)) = N_0(N_5(x)) = N_1(N_2(N_0(x))) = N_0(N_2(N_1(x)))$$
$$N_5(x) = N_2(N_1(x))$$
$$N_6(x) = N_0(N_1(x))$$
$$N_7(x) = N_2(N_6(x)) = N_1(N_8(x)) = N_2(N_0(N_1(x))) = N_1(N_0(N_2(x))) \quad (2.14)$$
$$N_8(x) = N_0(N_2(x))$$
$$N_9(x) = N_1(N_2(x))$$
$$N_{10}(x) = N_0(N_9(x)) = N_2(N_{11}(x)) = N_0(N_1(N_2(x))) = N_2(N_1(N_0(x)))$$
$$N_{11}(x) = N_1(N_0(x))$$

Обчислення координат сусідів клітини починається з визначення номера кільця, якому вона належить, за формулою (19), а також її орієнтації.

Для цього а номер n = n (a). Далі необхідно розглянути тринадцять варіантів розташування клітини всередині кільця (з урахуванням окремого розгляду різних орієнтацій при одному і тому ж розташуванні - дев'ятнадцять варіантів).

Необхідно відзначити, що орієнтації всіх кутів, а також останніх клітин в кільці відомі заздалегідь.

У таблиці 2.3 наводяться формули для знаходження безпосередніх сусідів клітини в залежності від її розташування і орієнтації (▲/▼). Для кутів орієнтація відома заздалегідь.

Таблиця 2.3 – Функції визначення головних сусідів для спіральних узагальнених координат на трикутній решітці

Розміщення	▲/▼	Індекс головного сусіда		
		0	1	2
$C_{n,0}$	▲	$C_{n+2,0}-1$	$a+1$	$C_{n+1,0}-1$
$(C_{n,0}. C_{n,1})$	▲	$a-1$	$a+1$	$C_{n-1,0}+a-C_{n,0}-1$
	▼	$a+1$	$a-1$	$C_{n+1,0}+a-C_{n,0}+1$
$C_{n,1}$	▲	$a-1$	$C_{n+1,1}+1$	$a+1$
$(C_{n,1}. C_{n,2})$	▲	$a-1$	$C_{n+1,1}+a-C_{n,1}+1$	$a+1$
	▼	$a+1$	$C_{n-1,1}+a-C_{n,1}-1$	$a-1$
$C_{n,2}$	▲	$a-1$	$C_{n+1,2}-1$	$a+1$
$(C_{n,2}. C_{n,3})$	▲	$C_{n-1,2}+a-C_{n,2}-1$	$a-1$	$a+1$
	▼	$C_{n+1,2}+a-C_{n,2}+1$	$a+1$	$a-1$
$C_{n,3}$	▼	$C_{n+1,3}-1$	$a+1$	$a-1$
$(C_{n,3}. C_{n,4})$	▲	$a+1$	$a-1$	$C_{n+1,3}+a-C_{n,3}+1$
	▼	$a-1$	$a+1$	$C_{n-1,3}+a-C_{n,3}-1$

Продовження таблиці 2.2

$C_{n,4}$	▼	$a–1$	$C_{n+1,4}+1$	$a+1$
$(C_{n,4}. C_{n,5})$	▲	$a+1$	$C_{n–1,4}+a–C_{n,4}–1$	$a–1$
	▼	$a–1$	$C_{n+1,4}+a–C_{n,4}+1$	$a+1$
$C_{n,5}$	▲	$C_{n+1,5}+1$	$a+1$	$a–1$
$(C_{n,5}. Cn+1,0–1)$	▲	$C_{n+1,5}+a–C_{n,5}+1$	$a+1$	$a–1$
	▼	$C_{n–1,5}+a–C_{n,5}–1$	$a–1$	$a+1$
$Cn+1,0–1$	▼	$C_{n–1,0}$	$a–1$	$C_{n,0}$

Як і в описаних вище методах, окремого розгляду вимагає клітина нуль, так як вона є всіма шістьма кутами нульового кільця і її розташування визначити неможливо. В силу цього повинні бути особливо визначено шість випадків - три сусіда нуля і нуль, як сусід трьох клітин: N_0 (0) = 12, N_1 (0) = 4, N_2 (0) = 8, N_0 (12) = 0, N_1 (4) = 0, N_2 (8) = 0.

Використання формул (20) призводить до рекурсивних викликів функцій при реалізації запропонованого методу. На відміну від квадратної решітки, де були використані аналогічні властивості, описувані формулами (15), для трикутної решітки глибина рекурсії сягатиме трьох при знаходженні безпосередніх сусідів з індексами 4, 7 і 10. Однак, якщо не користуватися рекурсивними формулами (20), то число «розташувань» клітин, які необхідно буде розглядати, збільшиться значно істотніше, ніж для квадратної решітки.

У зв'язку з тим, що тільки чверть безпосередніх сусідів клітини може бути знайдена без використання рекурсії, даний метод введення координат є вельми неефективним, якщо потрібно знаходити всі дванадцять сусідів для кожної клітки великої решітки на кожній ітерації.

3 ІМІТАЦІЙНЕ МОДЕЛЮВАННЯ РЕСТРУКТУРОВАНИХ МЕРЕЖЕВИХ СТРУКТУР З ВИКОРИСТАННЯМ КЛІТИННИХ АВТОМАТІВ

Реструктуризація - важливий аспект вивчення поведінки мережевих структур як при їх розробці, так і при (в процесі) їх практичної експлуатації, включаючи модернізацію і реорганізацію. Мережеві структури, незалежно від конкретної прикладної області призначення або використання, припускають ту або іншу ступінь автономності окремих вузлів і зв'язків, а також різну ступінь доступності для їх регламентного обслуговування. У зв'язку з цим актуальним є питання забезпечення (супроводу, інформаційної та технічної підтримки) змін топології мережевих структур:

- введення в дію або вихід з ладу окремих вузлів,
- зміна ресурсо-забезпеченості і функціональності окремих вузлів,
- міжвузлова передача ресурсів (ресурсоподдержка),
- міжвузлове перенесення функціональності (передача функцій),
- утворення нових зв'язків між вузлами,
- руйнування колишніх (раніше функціонували) зв'язків,
- зміна інформаційної навантаженості зв'язків,
- зміни характеру і призначення (цілей використання) зв'язків.
- загальне розподілення інформаційних процесів в мережевій структурі і ін.

Підтримка функціонування мережевої структури в умовах впливу різних зовнішніх факторів, що впливають передбачає одиночність (одноразово) або множинність (повторюваність або згруповані), регулярність або унікальність в часі окремих впливів зовнішнього середовища на мережеву структуру. В умовах наявності таких впливів структура функціонально видозмінюється (варіюється по функціональності). Мета видозмін - підтримання функціональності структури. Динаміка видозмін (динаміка реструктуризації) - визначається динамікою впливів зовнішнього

середовища. Дані речі (фактори і їх прояви) взаємопов'язані і, зрозуміло, специфічні для різних предметних областей. Але загальний характер взаємозв'язків і взаємозалежностей - єдиний і в цілому відповідає представленому якісному опису.

У зв'язку з цим актуальна проблематика моделювання мережевих структур в аспекті їх реструктурування. Як зазначалося, характер реструктурування багато в чому визначається особливостями предметної області. Але є інваріанти. До числа незмінних (незмінно цікавих) аспектів моделювання відноситься, зокрема, відтворення і вивчення динаміки процесів, що відбуваються. Динаміка, зокрема, проявляється як структурні зміни, які визначають зміни функціонального плану. Як наслідок, реалізуються зміни (різко залежать від конкретики предметної області) по перерозподілі ресурсів і можливостей мережевої структури із забезпечення максимальної підтримки її функціональності.

В рамках викладеної парадигми, для модельної реалізації обрана процедура підтримки функціонування регулярної мережевої структури передачею елементам, близьким до краху, резервних ресурсів сусідніх з ними елементів. Доцільність та допустимість такого режиму функціонування - в максимальному підтримці працездатності системи. Система, як ціле, – повільно деградує, але максимально підтримує свою функціональність.

3.1 Блок-схема програми моделі поведінки

Блок-схема програми моделі вивчення динаміки поведінки (розвитку, підтримки, деградації) мережевої структури представлена на рисунку 3.2. Підбір функціональності параметрів і відповідне алгоритмічне втілення в ній такі, що реалізується (може бути простежено) саме зазначений режим з максимальним уповільненням деградації системи. Метою моделювання може бути, зокрема, знаходження критеріїв і вказівку найбільш доцільного моменту проведення регламентного обслуговування або часткової

функціональної реконструкції системи. Залежно від конкретної предметної області, цільові орієнтації можуть дещо різнитися, але в цілому вивчення поведінки системи в динаміці – найбільш доцільний шлях забезпечення оптимального обслуговування (супроводження, підтримки) мережевої структури.

Старт алгоритму роботи моделі супроводжується відкриттям протоколу. Даний момент вимагає пояснень. Було визнано недоцільним зайвий раз ускладнювати програмну частину, замикаючи роботу всієї моделі на фінальну генерацію результатів. Існують програми, спеціалізовані і оптимізовані під обробку числових результатів. Обробка даних (безвідносно до конкретного типу та призначенню) - окрема просунута і істотно розвинена програмно-інформаційна галузь. У ній розвиваються, програмно реалізуються і вдосконалюються (з урахуванням конкретних практичних запитів) різні підходи, пов'язані зі статистичною обробкою, аппроксимацией, інтерполяцією і екстраполяцією даних. А також розвиваються засоби графічного відображення результатів. Дублювання досягнень цієї окремої інформаційної галузі стало б марною суттєвої розтратою сил з малими шансами на успіх отримання навіть порівнянних (не те що перевершують) результатів. Тому в процесі розробки моделі було визнано розумним зосередити зусилля на розрахунковій частині моделі, з висновком результатів (числових даних) в проміжний текстовий протокол. Подальша обробка даних і графічна репрезентація - проводиться з текстового протоколу в додатково обраному спеціалізованому засобі відображення.

Були використані електронні таблиці (Excel) на яких і реалізувалася графічна частина проекту. Реалізація введення даних - так само без спеціалізованого інтерфейсу. Різноманітність (багатопараметрічність) реалізації відтворюється системно-мережевий частини моделі передбачає безпосередню програмну модифицируемость моделі. Поточні відомі елементи алгоритмічних рішень не вичерпують всіх можливих варіантів. Модель являє собою відкриту систему. Як випливає з попереднього викладу,

структурно-мережева предметна область описана в досить загальному (узагальненому) вербальному плані, і навіть без функціональної формалізації. Це означає, що припустимо широкий інтерпретаційний контекст. При цьому контекст є відкритим, тобто модифікаційним і доповнюються практично в будь-якій частині. У зв'язку з цим, при поточній експлуатації та модифікації моделі, потрібно безпосередній доступ до програмного коду в окремих вузлах і блоках моделі. Тому не доцільний так само і окремий програмно реалірванний спеціалізований інтерфейс введення вихідних параметрів. Цей інтерфейс довелося б щоразу переробляти при функціональній модифікації моделі. Введені параметри, а також число параметрів, - є частиною функціоналу моделі.

Зокрема, це стосується і зміни структури мережі. Реструктуризація мережевої структури проводиться в даній моделі за допомогою зміни (часткової заміни) фрагмента логіки програми. Видозмінюється той елемент моделі, який визначає характер сусідства елементів мережевої структури. Досліджено 3 варіанти сусідства (4, 6 і 8 сусідів), які схематично можуть бути зображені в такий спосіб:

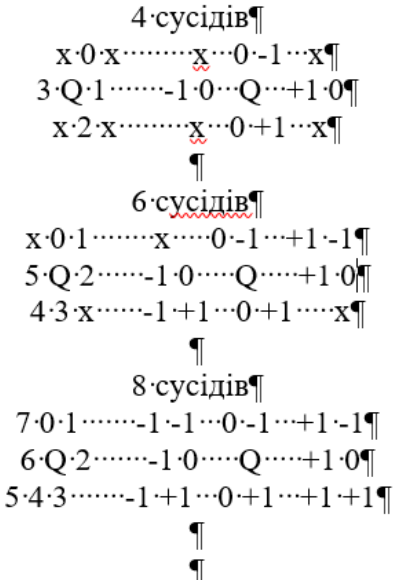

Рисунок 3.1 – Варіанти сусідства

Представлені схеми відображають логіку роботи при створенні відповідного блоку програми. Тут для кожної з груп сусідства показані:

- ліва група – порядок нумерації (за годинниковою стрілкою) в алгоритмі розгляду оточення елемента (0-1-2-3 для варіанта 4 сусідів, 0-1-2-3-4-5 для варіанту 6 сусідів і т.д.) ;

- права група – збільшення індексів, відповідні зміни координат на поле навколо розглянутого елемента (наприклад, певий верхній елемент - -1-1, а правий нижній - + 1 + 1).

Чи не розглянутий елемент (який не потрапляє в сусідство) позначений «х». Даний варіант організації виявився зручним в програмній реалізації моделі.

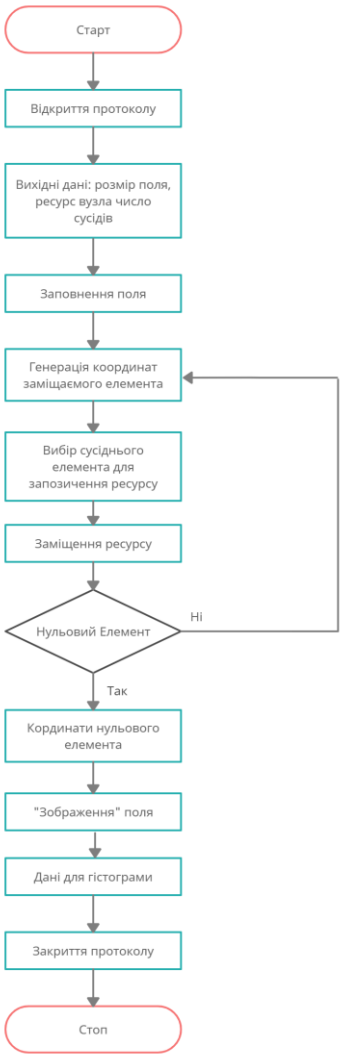

Рисунок 3.2 – Блок-схема модели

Покроковий опис розробленого алгоритму наступний:

Крок 1. Початок. Старт роботи моделі. Передбачається введеня даних.

Крок 2. Відкриття протоколу. Програма відкриває протокол з прописаним в тексті (в коді програми) ім'ям на дописування. Це означає, що колишні записи в протоколі (результати попередніх запусків програми) зберігаються, а нові результати дописують «в хвіст» тексту протоколу.

Крок 3. Введення даних (розмір поля, ресурс вузла мережі, число сусідніх вузлів). Дані, що стосуються параметрів моделі, стають прийнятими до виконання).

Крок 4. Формування і початкове заповнення поля. Поле (масив поля) формується і заповнюється однаковими вихідними числами - значеннями ресурсу елемента поля (вузла мережевої структури)

Крок 5. Псевдовипадкова генерація координат чергового зміненного елемента. Генерується пара псевдовипадкових чисел, які масштабуються як координати одного з елементів поля. Цей елемент буде модифікований з використанням ресурсу одного зі своїх сусідів. Це буде інтерпретовано в моделі як видалення (втрата) цього елемента і заміщення його іншим, частково використовує ресурс одного з сусідніх (оточуючих) елементів.

Крок 6. Псевдовипадковий вибір сусіднього елемента для запозичення його ресурсу. Буде обраний той сусідній елемент, який «поділиться» своїм ресурсом з заміщаються елементом.

Крок 7. Реалізація заміщення ресурсу. Здійснення дій по заміщенню, згідно значень, обраним на Кроків 5 і 6.

Крок 8. Перевірка наявності нульового елемента. Після виконання Кроку 7 може виявитися так, що заміщається елемент цілком задіяв (використовував) ресурс свого сусіда. При цьому в полі з'являється елемент з нульовим ресурсом. Тому здійснюється повний перебір елементів поля з метою пошуку нульового елемента. Якщо нульовий елемент НЕ виявлено, це означає, що функціональність мережевої структури зберігається. Перехід на Крок 5 для обробки чергового заміщає елемента. Якщо нульовий елемент

виявлений, це означає, що мережева структура втрачає функціональність. Програма виходить з циклу обробки заміщаються елементів і йде на завершення - Крок 9.

Крок 9. Висновок координат нульового елемента. Координати нульового елемента заносяться в потокол.

Крок 10. Висновок «зображення» поля. В протокол заноситься поточний стан матриці поля. Елементи матриці (все крім одного - нульового) відображають свій залишок невикористаного ресурсу.

Крок 11. Формування даних для побудови гістограми. Підраховується число елементів кожного із значень ресурсу. Лінійка цих чисел є гістограма розподілу залишкового ресурсу мережевої структури. Даний набір чисел - основне досягнення даного запуску програми, - основний результат моделювання.

Крок 12. Закриття протоколу. Файл протоколу закривається і стає доступним для оператора.

Крок 13. Стоп. Завершення роботи моделі. Програма зависає у вічному циклі в очікуванні натискання будь-якої клавіші комп'ютера. Це зроблено для того, щоб оператор побачив на екрані повідомлення про завершення роботи програми і продовжив дослідження. Зокрема, можливо, оператор введе нові дані і перезапустити модель.

3.2 Програмна реалізація

У Додатку Б представлено лістинг програми моделі мережевої структури.

Програма забезпечена коментарями, які застосовувалися (використовувалися) при написанні тексту. Тому, в принципі, текст програми достатній для розуміння. З огляду на наявність вище блок-схеми і покроковий опис, додаткових пояснень, мабуть, не потрібно.

Програма реалізована на мові Python, версія 2.7. Подальші версії моделі, можливо слід перевести на Python 3 тільки в разі переходу до багатовимірним мережевим структурам. Іншим аргументом обґрунтування доцільності може з'явитися більш висока продуктивність і наявність спеціалізованих графічних бібліотек.

Додатки 2, 3 і 4 ілюструють типові результати, отримані для коефіцієнтів сусідства 4, 6 і 8, відповідно. У зазначених трьох варіантах сусідства представлені однакові значення вихідних параметрів:

- ресурс – 10 і 50,
- розмір поля – 20x20,
- число реалізацій – 15.

Зазначена однаковість дозволяє зіставляти результати.

Для ресурсу 10 представлена таблиця з 15 рядків – даних для побудови гістограм розподілу кількості елементів із заданим ресурсів при досягненні системою стану нульового ресурсу в одному з елементів. Для ресурсу 10 таблиця має, природно, 11 стовпців. Аналогічна таблиця для ресурсу 50 – містить 51 стовпець. Подібний розмір - не цілком зручний для приміщення на сторінку (так щоб забезпечувалася і повнота, і оглядовість). А враховуючи, що таблиця має чисто демонстраційний характер, поміщати її - було визнано просто недоцільним.

Обидві таблиці ілюструються графіками – тривимірними збірками гістограм в лінійному (зліва) і логарифмічному (праворуч) форматі по вертикалі. Два верхніх графіка відповідають ресурсу 10, два нижніх - ресурсу 50. Лінійний масштаб графіків дозволяє якісно оцінити їх загальну ширину, «розкид хвостів», а також побачити їх колоколообразний характер (нормальний закон розподілу). Графіки з логарифмічним масштабом - більш наочні в ілюстрації такого неформалізованного якісного показника, як «загальна маса» вузлів мережевої структури, що залишилися в функціональному стані після виходу з ладу (обнулення ресурсу) одного елемента. Можливо, в ході подальших досліджень з'ясується

інформативність такого показника. Звісно ж, що залишкова «загальна маса», як показник працездатності, може бути вивчена для різних значень числа «нульових» елементів, і як характеристика - може бути корисною при прийнятті рішень (наприклад, експертних або нечітких) щодо підтримки працездатності мережевої структури.

Гістограми (функції розподілу) для значення ресурсу 50 мають «в'ялий» горбистий вид, що пояснюється малим (для такого ресурсу) розміром поля.

Гістограми ресурсу 50 в порівнянні з гістограмами ресурсу 10 - істотно зміщені від площини («стінки»), що відповідає вихідного стану ресурсності. Пояснення – велике значення початкового ресурсу. «Загальна маса» ресурсу при появі первогонуля – істотно знижена.

Суттєвої якісної відмінності у вигляді гістограм для 4, 6 і 8 сусідів – не спостерігається. Мабуть, такі відмінності все-таки існують, але для їх виявлення потрібні великі статистичні вибірки та більш розвинена статистична обробка. Все це може стати предметом подальших досліджень.

У Додатку 2 представлено лістинг програми моделі мережевої структури.

Програма забезпечена коментарями, які застосовувалися (використовувалися) при написанні тексту. Тому, в принципі, текст програми достатній для розуміння. З огляду на наявність вище блок-схеми і покроковий опис, додаткових пояснень, мабуть, не потрібно.

Програма реалізована на мові Python, версія 2.7. Подальші версії моделі, можливо слід перевести на Python 3 тільки в разі переходу до багатовимірним мережевим структурам. Іншим аргументом обґрунтування доцільності може з'явитися більш висока продуктивність і наявність спеціалізованих графічних бібліотек.

Додатки 2, 3 і 4 ілюструють типові результати, отримані для коефіцієнтів сусідства 4, 6 і 8, відповідно. У зазначених трьох варіантах сусідства представлені однакові значення вихідних параметрів:

- ресурс - 10 і 50,
- розмір поля – 20х20,
- число реалізацій – 15.

Зазначена однаковість дозволяє зіставляти результати.

Для ресурсу 10 представлена таблиця з 15 рядків – даних для побудови гістограм розподілу кількості елементів із заданим ресурсів при досягненні системою стану нульового ресурсу в одному з елементів. Для ресурсу 10 таблиця має, природно, 11 стовпців. Аналогічна таблиця для ресурсу 50 – містить 51 стовпець. Подібний розмір – не цілком зручний для приміщення на сторінку (так щоб забезпечувалася і повнота, і оглядовість). А враховуючи, що таблиця має чисто демонстраційний характер, поміщати її – було визнано просто недоцільним.

Попередньо оброблені протоколи і графіки коефіцієнт сусідства 4
Ресурс: 10 і 50. Поле: 20. Кількість реалізацій: 15

1	6	7	16	30	45	78	73	78	48	18
1	1	7	12	20	43	72	82	88	52	22
1	2	3	9	23	48	76	91	80	46	21
1	0	2	8	7	33	71	91	96	70	21
1	2	4	11	16	34	59	97	93	66	17
1	1	3	11	32	52	63	91	86	43	17
1	2	5	8	16	47	59	92	86	67	17
1	0	1	3	11	25	58	84	93	84	40
1	0	2	13	27	56	80	82	81	47	11
1	1	4	17	26	47	75	98	70	49	12
1	0	12	16	28	47	72	87	77	46	14
1	3	2	18	18	40	65	79	98	56	20
1	5	5	12	29	49	94	92	60	38	15
1	2	4	21	29	50	69	81	80	52	11
1	0	0	9	23	31	72	98	96	54	16

Рисунок 3.3 – Попередньо оброблені протоколи і графіки коефіцієнту сусідства 4

Обидві таблиці ілюструються графіками – тривимірними збірками гістограм в лінійному (зліва) і логарифмічному (праворуч) форматі по вертикалі. Два верхніх графіка відповідають ресурсу 10, два нижніх – ресурсу 50. Лінійний масштаб графіків дозволяє якісно оцінити їх загальну ширину, «розкид хвостів», а також побачити їх колоколообразний характер (нормальний закон розподілу). Графіки з логарифмічним масштабом – більш наочні в ілюстрації такого неформализованного якісного показника, як «загальна маса» вузлів мережевої структури, що залишилися в функціональному стані після виходу з ладу (обнулення ресурсу) одного елемента. Можливо, в ході подальших досліджень з'ясується інформативність такого показника. Звісно ж, що залишкова «загальна маса», як показник працездатності, може бути вивчена для різних значень числа «нульових» елементів, і як характеристика – може бути корисною при прийнятті рішень (наприклад, експертних або нечітких) щодо підтримки працездатності мережевої структури.

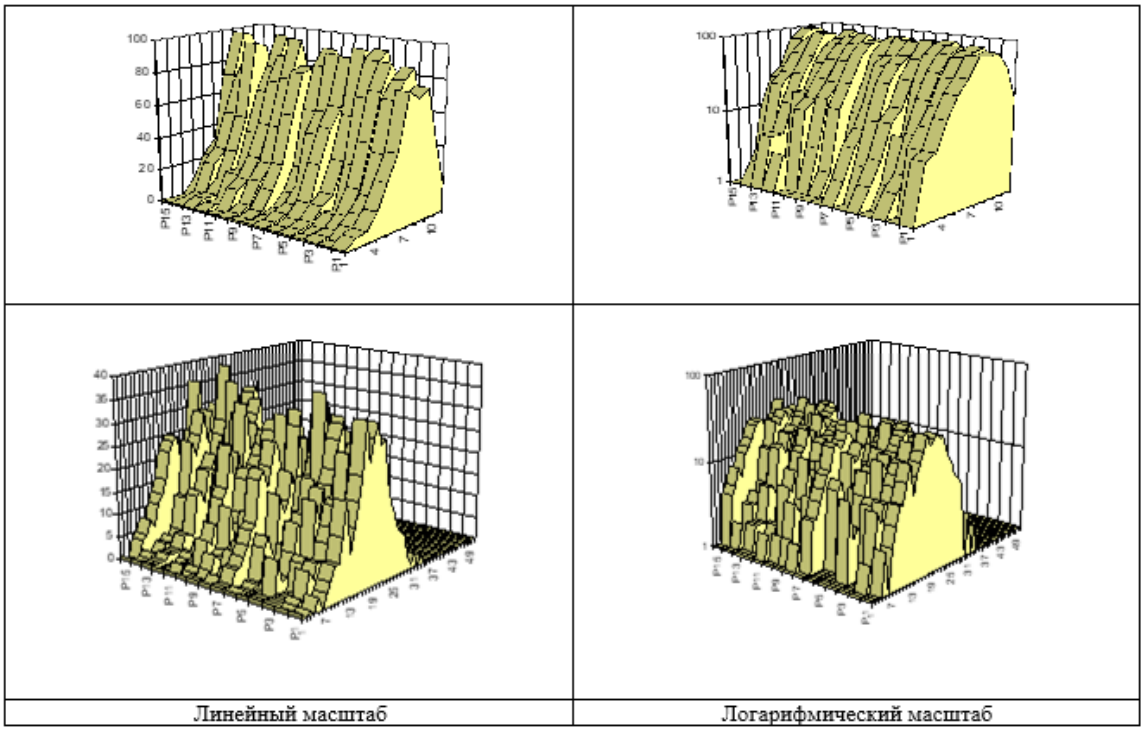

Рисунок 3.3 – Результати моделювання

Гістограми (функції розподілу) для значення ресурсу 50 мають «в'ялий» горбистий вид, що пояснюється малим (для такого ресурсу) розміром поля.

Гістограми ресурсу 50 в порівнянні з гістограмами ресурсу 10 – істотно зміщені від площини («стінки»), що відповідає вихідного стану ресурсності. Пояснення – велике значення початкового ресурсу. «Загальна маса» ресурсу при появі первогонуля – істотно знижена.

Суттєвої якісної відмінності у вигляді гістограм для 4, 6 і 8 сусідів – не спостерігається. Мабуть, такі відмінності все-таки існують, але для їх виявлення потрібні великі статистичні вибірки та більш розвинена статистична обробка. Все це може стати предметом подальших досліджень.

ВИСНОВКИ

Розглянути базові принципи організації клітинної структури об'єктів живої природи. Розроблена комп'ютерна модель ресурсно-залежних клітинних автоматів з використанням мови Python. Проведено аналіз реструктурованих мережевих структур. Провести імітаційне моделювання.

З огляду на сказане вище, найближчими наступними напрямками досліджень по розробленої моделі можуть бути:

- варіювання вихідного ресурсу і розміру поля,
- варіювання конфігурацій сусідства (охоплення випадків 2, 3, 5 і 7 сусідів),
- вивчення анізотропності поля за вибором сусідів,
- введення нелінійних генераторів випадкових числі,
- запровадження нерівномірності (зокрема, знову ж анізотропності) з вироблення ресурсу в залежності від положення елемента в поле.

Може бути так само цікавим в інтерпретаційному плані введення багатовимірності чи багатозв'язного поля. Але при цьому можуть бути затребувані значні обчислювальні потужності.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Аристов А. О. Теория квазиклеточных сетей: научная монография – М: МИСиС, 2014. –188 с.

2. Ванаг В. К. Исследование пространственно распределённых динамических систем методами вероятностного клеточного автомата (рус.) // Успехи физических наук. Обзоры актуальных проблем. : журнал. – май 1999. – Т. 169, № 5. – С. 481-505.

3. Фон Нейман Дж. Теория самовоспроизводящихся автоматов. М.: Мир, 1971.

4. Тоффоли Т., Марголус Н.Машины клеточных автоматов. М.: Мир, 1991. 280 с.

5. Бандман О.Л. Клеточно-автоматные модели пространственной динамики // Системная информатика.– 2005.– Вып. 10.– С. 57–113.

6. Бандман О.Л. Параллельная реализация клеточно-автоматных алгоритмов моделирования пространственной динамики // Сиб. журн. вычисл. математики. / РАН. Сиб. отд-ние.– Новосибирск, 2007.– Т. 10, No 4. – С. 335–348

7. Міхаль О.П., Гук А.С., Скрипник М.С. Клітинно-автоматне моделювання динаміки мережевих структур // Проблеми інформатизації. Тези доповідей восьмої міжнародної науково-технічної конференції / Черкаси-Харків-Баку-Бельсько-Бяла, 2020 – Т1.– С.43