Міністерство освіти і науки України
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»
Кафедра мультимедійних систем факультету інформатики

Використання механізмів клітинних автоматів для моделювання складних систем

Текстова частина до курсової роботи
за спеціальністю «Інженерія програмного забезпечення» – 121

Керівник курсової роботи
доцент Жежерун О.П.
_____
(підпис)
“____” _____ 2022 р.

Виконав студент
Войцеховський Є.О.
“____” _____ 2022 р.

Київ 2022

Міністерство освіти і науки України
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»
Кафедра мультимедійних систем факультету інформатики

ЗАТВЕРДЖУЮ
Завідувач кафедри мультимедійних систем,
доцент, кандидат наук – Жежерун О.П.
(підпис)
„____”_____2021 р.

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ
на курсову роботу
студенту Войцеховському Євгенію Олександровичу
факультету інформатики 1 курсу магістерської програми
ТЕМА: **Використання механізмів клітинних автоматів для моделювання складних систем**

Зміст ТЧ до курсової роботи:

1. Календарний план
2. Вступ
3. Огляд теоретичної частини
4. Висновки
5. Список використаних джерел

Дата видачі „___” _____ 2021 р. Керівник _____
(підпис)
Завдання отримав _____
(підпис)

Тема: **Використання механізмів клітинних автоматів для моделювання складних систем**

№ п/п	Назва етапу курсової роботи	Термін виконання етапу	Примітка
1.	Отримання завдання на курсову роботу.	22.10.2021	
2.	Огляд літератури за темою роботи.	15.01.2022	
3.	Проведення досліджень	03.02.2022	
3.	Опис результатів дослідження	21.03.2022	
4.	Аналіз отриманих результатів з керівником	18.04.2022	
6.	Корегування роботи	25.05.2022	
7.	Створення презентації та написання доповіді.	01.06.2022	
8.	Остаточне оформлення пояснювальної роботи та слайдів.	09.06.2022	
9.	Захист курсової роботи	14.06.2022	

Студент _____
Керівник _____
“_____”_____

ЗМІСТ

АНОТАЦІЯ

Ця робота досліджує базові характеристики класичного клітинного автомата (КА) та як ці механізми впливають на поведінку моделей, що побудовані на основі КА.

У першому розділі розглянуті характеристики решітки КА, та як моделі змінюють параметри решітки для власних потреб.

У другому розділі розглянуті характеристики агентів КА, та як моделі використовують, додають або змінюють ці характеристики.

ВСТУП

В останні роки поняття «клітинний автомат» зустрічається в багатьох галузях людської діяльності, і клітинні автомати знайшли своє використання в якості надійного засобу побудувати модель певних процесів, що базуються на виконанні простих правил. Це стало можливим завдяки розвитку обчислювальної техніки, що дозволило обраховувати стани тисяч клітинних автоматів за долі секунди. Подібний прорив не пройшов безслідно, і люди почали цікавитися тими речами, які вони не могли обрахувати вручну. Тому розвиток клітинних автоматів став можливий лише завдяки розвитку техніки та працям багатьох дослідників, одним з яких був Джон фон Нейман.

Вчені з усього світу не припиняють своє дослідження можливостей клітинних автоматів та намагаються максимізувати їх потенціал як обчислювальної моделі. Ця робота ставить за свою мету проаналізувати параметри клітинних автоматів та дослідити, як інші дослідники використовують ці можливості у своїх моделях.

Предмет дослідження цієї курсової – моделі обчислення в інформатиці.

Об’єкт дослідження цієї курсової – параметри клітинних автоматів.

Текстова частина цієї курсової роботи складається із вступу, загальних понять, двох розділів, висновків та списку використаних джерел.

ЗАГАЛЬНІ ПОНЯТТЯ

Клітинний автомат (КА) – це дискретна модель, що на основі простих правил може генерувати складну поведінку. Це дозволяє використовувати його в якості засобу для побудови складних математичних моделей з різною поведінкою, які використовуються в багатьох сферах.

Класичний КА складається з n-мірної решітки (поле), набору агентів, набору правил та певного початкового стану. Зміна будь-якого з цих елементів приводить до радикальних змін властивостей певного КА. Ці зміни дають можливість створити різні типи КА зі своєю унікальною поведінкою.

Після запуску КА отримує стани певної кількості інших автоматів та свій власний стан, використовує правила переходу та змінює відповідно свій стан. Такий процес виконує кожен (або якась кількість) КА на полі, після чого ця послідовність кроків повторюється. Така послідовна зміна станів поля та агентів може набути характерних властивостей та поведінки певного реального явища або бути використана для обрахунку певних обчислень.

Хоч КА базуються на простих правилах та елементах, але їх широке використання стало можливе з розвитком електронно-обчислювальної техніки, бо зі зростанням кількості клітинних автоматів та розміру поля швидко зростає кількість обчислень, які необхідно зробити. Звісно, ці обрахунки може зробити й людина, але для моделей з тисяч КА ці обчислення потребують надто велику кількість часу, щоб досягти потрібного ефекту або результату.

Саме тому великі КА показують себе найкраще на обчислювальних машинах. Оскільки КА обраховується на деякому одноманітному полі, це відкриває можливість паралельного обчислення різних областей цього поля одразу на кількох машинах. Не всі КА можливо обчислити швидше методами розпаралелюванням (КА невеликого розміру на кількох машинах будуть обчислюватися навіть довше, бо час обрахунків невеликих КА значно менший, ніж час на спілкування між машинами та об’єднання результатів їх роботи).

КА може бути заданий чотирма способами:

- *Графічно.* Для цього способу потрібно намалювати стан КА та його сусідів та поставити у відповідність цьому малюнку стан, у який перейде КА на наступному кроці. Це зручний спосіб для графічного представлення та розуміння людиною, але для машини цей спосіб незручний.

- *Через умови.* Цей спосіб дозволяє обрати одразу кілька автоматів, які відповідають певним умовам. Цей спосіб можна використати при задаванні правил для відомого КА «Гра життя», що було зроблено в роботі [12]. Користувач задає певні правила, а обчислювальний пристрій знаходить КА, які відповідають цим правилам.

- *Код Вольфрама.* Це відома система називання КА, яка дозволяє однозначно визначити певний КА. Щоб присвоїти КА певний код, потрібно знайти всі його комбінації власного стану та сусідів, відсортувати за зменшенням, поставити у відповідність певній комбінації стан, у який перейде КА, та перевести цей список в число (найчастіше переводять у десяткову систему). Це зручний спосіб запису правил КА для машин.

- *Логічний вираз*. Робота [8] дає таблицю, в якій кожному правилу для одновимірного двоколірного автомата відповідає певний логічний вираз, що складається з мінімальної кількості логічних операторів. Наприклад, правило 16 за кодом Вольфрама буде представлено у вигляді наступного правила: $p \wedge (\neg q) \wedge (\neg r)$.

РОЗДІЛ 1. РЕШІТКА

1.1 Розмірність

Перший параметр решітки, що впливає на роботу КА, це її розмірність. Решітка – це n-мірний простір, у якому працює певний КА. Найчастіше використовується одновимірна та двовимірна решітка, хоч деколи може використовуватися і тривимірна решітка або решітка більшої розмірності, якщо того потребує предметна область певної моделі. Причина такого широкого використання одновимірної та двовимірної решітки – це зручність їх графічного представлення.

Розглянемо більш детально одновимірну (рисунок 1.1.а) та двовимірну решітку (рисунок 1.1.б) з клітинками квадратної форми. Одновимірна решітка КА має вигляд стрічки з одної клітинки в ширину, яка тягнеться по осі Ox. Але одновимірний КА частіше всього представляють на двовимірній решітці, в якій кожен рядок – це одна ітерація КА на одновимірній решітці. Таке представлення дозволяє показати багато ітерацій роботи КА, знайти в них певні закономірності та виділити шаблони поведінки.

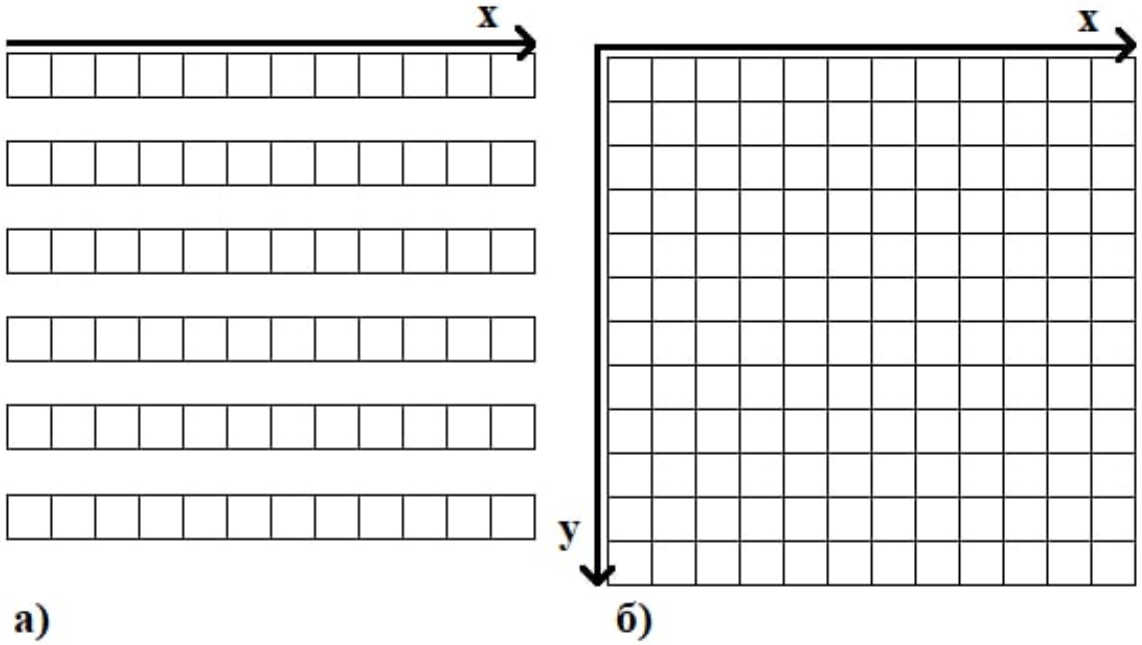

Рисунок 1.1 – Виміри решітки КА.

Двовимірну решітку у більшості випадків представляють на двовимірній площині, що тягнеться по осі Ox та Oy. На відміну від одновимірного КА

решітка двовимірного КА показує в момент часу лише одну ітерацію роботи КА. При наявності засобів представлення решітки у тривимірному просторі можна побачити одразу деяку кількість ітерацій КА на двовимірній решітці. Тобто, зображення кількох ітерацій n-мірної решітки можливо у n+1 просторі. Або потрібно розміщувати поруч двовимірні решітки, як це було зроблено з одновимірними решітками.

Робота «Cellular Automata Machines» [2] від Norman Margolus розглядає машину для обчислення КА, яка називається CAM-7. Ця машина складається з 8192 поверхневих модулів, кожен з яких має решітку 512x512 клітинок, що дозволяє моделювати КА на двовимірній та тривимірній решітці. Для тривимірних симуляцій використовується конфігурація 512x512x512x16. Ця машина має тривимірний дисплей, в якому в ряд виставлені 512 поверхонь розміром 512x512 клітинок. Такий дисплей використовується, щоб показати моделі роботи КА, в яких наявна невелика кількість даних, що дозволяє симулювати поведінку світла або інших фізичних явищ у трьох вимірах.

Таким чином якщо важливо показати роботу КА, то використовуються одновимірні та двовимірні решітки. Якщо ж показ не є обов’язковим для цілей певного дослідження, то можуть використовуватися решітки з трьома або більшою кількістю вимірів.

1.2 Форма клітинок

Другий параметр решітки – це форма її клітинок. Найчастіше клітинки решітки мають форму квадратів (рисунок 1.1), шестикутників або трикутників (рисунок 1.2). Проте клітинкам не заборонено мати інші форми. Наприклад, може використовуватися решітка з прямокутників, ромбів, паралелограмів, прямих трикутників або решітка, що поєднує в собі кілька різних форм геометричних фігур. Але в загальному решітка повинна мати клітинки однакової форми та розміру, що обмежує їх вибір.

Форма решітки дозволяє визначити кількість прямих сусідів певної клітинки. Клітинка у квадратній решітці має 8 сусідів (4 по гранях клітинки та 4

по вершинах), шестикутна має 6 сусідів по гранях та трикутна має 12 сусідів (3 по гранях та 9 по вершинах).

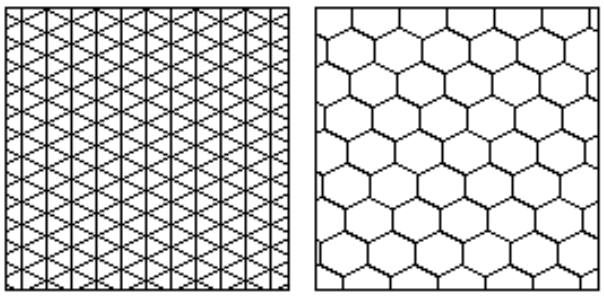

Рисунок 1.2 – Решітка КА з трикутників та шестикутників

Якщо модель КА вважає своїми сусідами лише ті клітинки, які дотикаються до певної решітки своїми гранями або вершинами, то це впливає на види завдань, які може розв’язувати КА.

Стандартна форма клітинок – це квадрат, що дає середню кількість сусідів по гранях та сусідів загалом. Також цю решітку найпростіше будувати та представляти у графічному та електронному вигляді. Наприклад, КА «Гра життя» використовує саме квадратну решітку. Також така решітка підходить для моделювання поведінки води, бо молекула води містить чотири можливих місця для гідрогенних зв’язків, як зазначає робота [3], що використовувала квадратну решітку для створення моделей розчинення у воді, симуляції гідрофобного ефекту та змішування води й олії.

Коли виникає потреба у більшій кількості сусідів по гранях, то використовується решітка з шестикутників. Робота [9] використовує шестикутну решітку для симуляції поведінки та росту багатоклітинних організмів різних форм. Предметна область цієї роботи вимагає можливість будувати фігури, які будуть найкраще виглядати на шестикутній, а не на квадратній решітці. Ця робота створила моделі для росту ниткоподібних структур (діляться лише клітинки, що знаходяться на краях «нитки»), складних ниткоподібних структур (через певні проміжки нитка розгалужується) та симетричного організму під кодовою назвою «черв’як», що мала відростки по боках. На рисунку 1.3 показані результати роботи цих моделей.

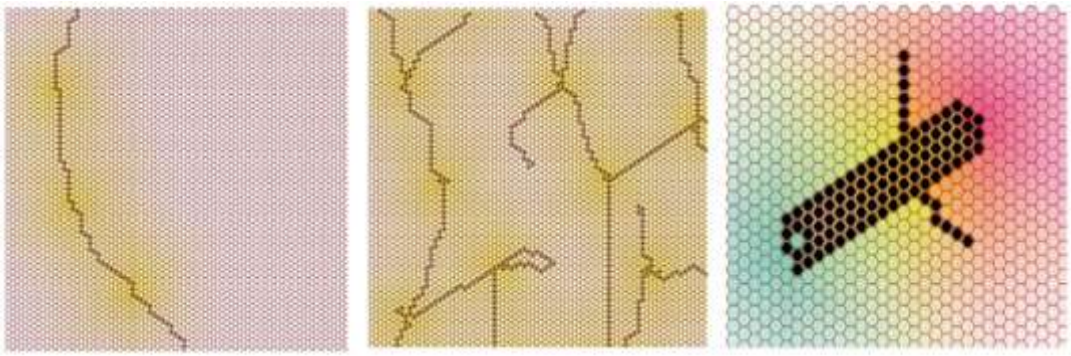

Рисунок 1.3 – Використання шестикутної решітки

І нарешті трикутна решітка дає найбільшу кількість сусідів, що може використовуватися для дуже точних моделей, які дозволяють створити фігури набагато більшої кількості форм, ніж це доступно шестикутній решітці.

1.3 Окіл

Кожна клітинка має певний окіл, в межах якого інші клітинки вважаються її сусідами. Для визначення цього околу існує два основних підходи: околиця фон Неймана (рисунок 1.4.а) та околиця Мура (рисунок 1.4.б). Околиця фон Неймана визначає сусідами для певної клітинки ті клітинки, до яких вона дотикається своїми гранями. Околиця Мура вважає сусідами ті клітинки, до яких клітинка дотикається своїми вершинами. Всі клітинки, які знаходяться в межах околиця фон Неймана або околиці Мура вважаються сусідніми для клітинки, яка на рисунку 1.4 позначена жовтим кольором. Ці околиці можуть мати ранг (зелені клітинки показують околицю першого рангу, а зелені та сині – другого рангу).

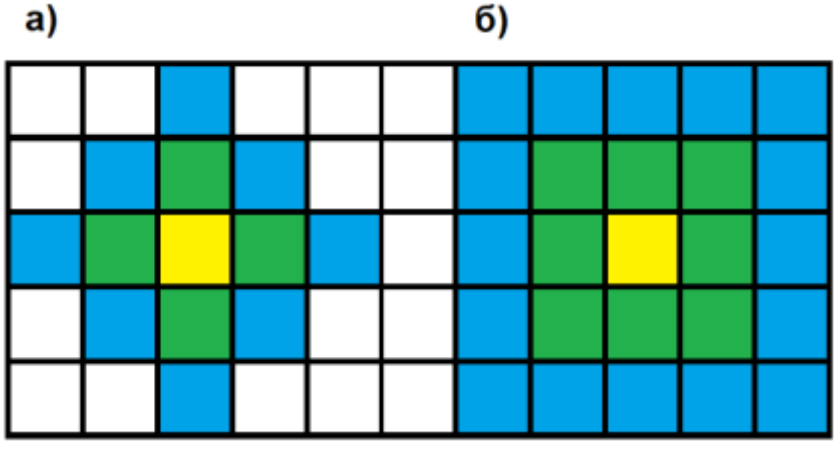

Рисунок 1.4 Околиця фон Неймана та околиця Мура

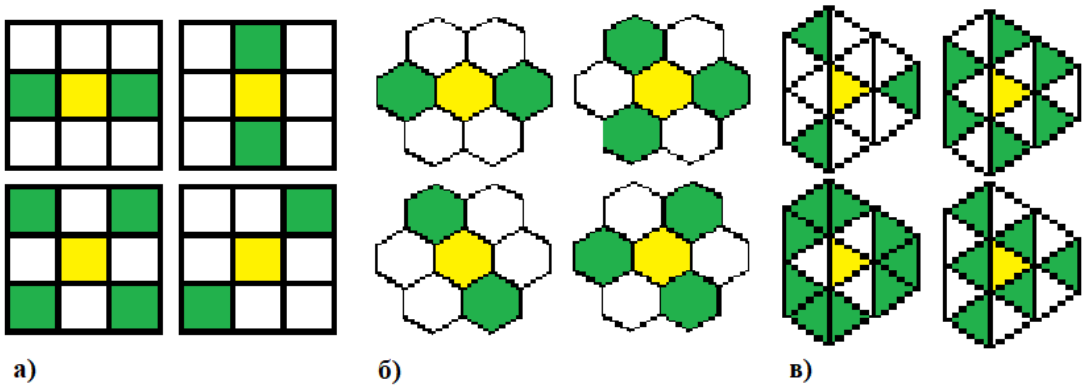

Рисунок 1.5 Інші види околиць

На рисунку 1.5 показані інші способи визначення сусідів на основі околиць фон Неймана та Мура. Не існує обмежень, яким чином потрібно визначати околиці клітинки та її сусідів. На основі цих правил можливо придумати значно більше правил визначення сусідів. Також можна розглянути гру шахи як клітинний автомат та побачити, які існують інші способи знайти сусідів (знайти клітинку для фігури, яку вона може «побити»).

1.4 Розмір

Останній параметр решітки – це її розмір. Не існує чітких обмежень по розміру решітки. Кожна задача та модель КА використовує решітку того розміру, який потрібен для її завдання. Розмір решітки або обмежує сама модель, або він обмежується обчислювальними можливостями та розміром пам’яті певної машини. Збільшити розмір решітки допомагають паралельні обчислення на кількох машинах. Тому якщо користувач має потребу в більшій решітці, він повинен використати більш потужний обчислювальний пристрій або кілька пристроїв.

Решітка може бути нескінченною, але все ж її якось обмежують. Існує два види обмежень: встановлення чітких меж решітки або замикання решітки. У першому випадку з’являється необхідність по-іншому обраховувати КА, які розташовані біля межі решітки. Для цього зазвичай ставлять за межею решітки додаткові нулеві клітинки, які враховуються в обчисленні, але не змінюють свій

стан. Існує можливість встановити для граничних клітинок решітки інші правила для обчислення нового стану КА, але таке використовується рідше.

Тому найпростіший варіант, який використовується для решітки КА та дозволяє спростити обчислення, це замикання граней. Для одновимірного КА потрібно замкнути дві сторони у певного роду кільце (рисунок 1.6.а). Для двовимірного КА можна замкнути дві бокові сторони, верхню та нижню частину або замкнути всі чотири сторони. У перших двох випадках решітка прийме форму циліндру (рисунок 1.6.б), а в останньому решітка стане тором (рисунок 1.6.в).

Робота [4] використовувала решітку, яка була згорнута у тор та в якій стани КА обчислювалися асинхронно. Така модель могла моделювати розчинення речовини у воді, гідрофобний ефект, водну дифузію, поведінку незмішуваних речовин, їх розподіл та мембранну проникність.

Не завжди необхідна велика решітка для обчислення певних задач. У роботі [2] описується машина CAM-6 для клітинних автоматів, яка використовує решітку розміром 256x256 клітинок. На стан кожної клітинки в цій машині виділяється 4 біти. І така решітка та обчислювальні можливості машини дозволяють їй моделювати різні фізичні процеси. Наприклад, тут створювалася модель руху молекул газу в закритому просторі, вивчалося відбивання частинок від стінок посудини, моделювалося відбиття та заломлення після проходження через лінзу та моделювався ріст дендритної структури з частинок, які були випадковим чином розміщені на решітці.

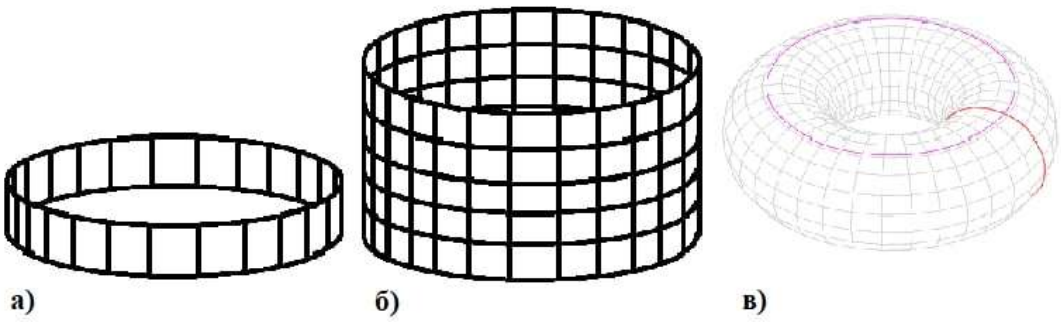

Рисунок 1.6 – Замикання сторін квадратної решітки

1.5 Використання решітки

Решітка у моделі КА – це поле, що використовується для роботи КА. Як правило, на кожній клітинці такого поля розташований один агент КА, що змінює свій стан кожну ітерацію перегляду всіх агентів (є винятки, про які буде згадано пізніше). Логічно припустити, що на одній клітинці може бути кілька агентів, але такий підхід широко не використовується.

Робота [6] використовувала квадратну двовимірну решітку розміром 15x15 клітинок для створення КА «Hunting for the Customer» (Полювання за покупцем). Кожне клітинка на решітці цієї моделі була покупцем, що купував продукти в деякої фірми. На решітці розташовувалася певна кількість фірм, які намагалися максимізувати кількість покупців, що купували їх товари. Для цього вони мали таке правило руху: якщо кількість покупців менше певного значення k, то фірма сканує свої сусідні клітинки з околиці фон Неймана та переміщається в ту із них, яка збільшить кількість доступних покупців. Після такого руху повинні були оновлюватися всі змінні цієї моделі. В результаті роботи такої моделі утворювалися кластери фірм, які розміщувалися по колу навколо центру поля, але сам центр поля залишався незайнятим.

КА можуть використовувати для своєї роботи не одну решітку, а декілька. Наприклад, у роботі [11], що досліджує побудову стохастичної моделі руху людей, використовуються два поля: статичне та динамічне. На статичному полі (решітці) розміщуються агенти КА, а динамічне поле зберігає слід частинок, який вони залишаються після свого руху. На початку обчислень динамічне поле є порожнім, але після початку обчислень воно починає заповнюватися та використовуватися для обчислень шляху частинки. Таке поле характеризується тим, що з часом воно забуває рухи частинок, які вже не впливають на обчислення маршрутів інших частинок. Таке динамічне поле має той же розмір, що й статичне, але воно має інші властивості.

Деякі завдання (наприклад обробка зображень) можуть використовувати декілька решіток, на кожній з яких працюють КА, а потім результат їх роботи буде об’єднаний. Саме так працює паралельне обчислення на декількох

машинах, кожна з яких обраховує свою частину решітки. Але коли обчислення проводяться на одній машині, то достатньо будувати модель на одній решітці, що в різних моделях може набувати свої властивостей (у моделі руху людей можуть бути нерухомі перешкоди, які частинки повинні обходити).

РОЗДІЛ 2. АГЕНТИ

2.1 Зв'язки з іншими агентами

Для обчислень КА використовує власний стан та стан сусідів. Саме ці значення використовуються для визначення нового стану КА за певними правилами. Тому перед початком обчислень необхідно встановити зв'язки з іншими КА (агентами). Ці зв'язки можуть не бути визначені, але КА повинен якимось чином контактувати з іншими КА, бо тоді неможливі будь-які обчислення у цій моделі.

Найпростіший метод встановлення зв'язків – це створення зв'язків з КА, які знаходяться на сусідніх клітинках від певного КА. Це формує зв'язки сусідства. Сусідніми клітинками можуть вважатися інші клітинки в околицях фон Неймана та Мура або інших околиць, які є похідними від цих видів околиць. Цей спосіб встановлення зв'язків дуже просто запрограмувати, але в такому підході існують певні проблеми.

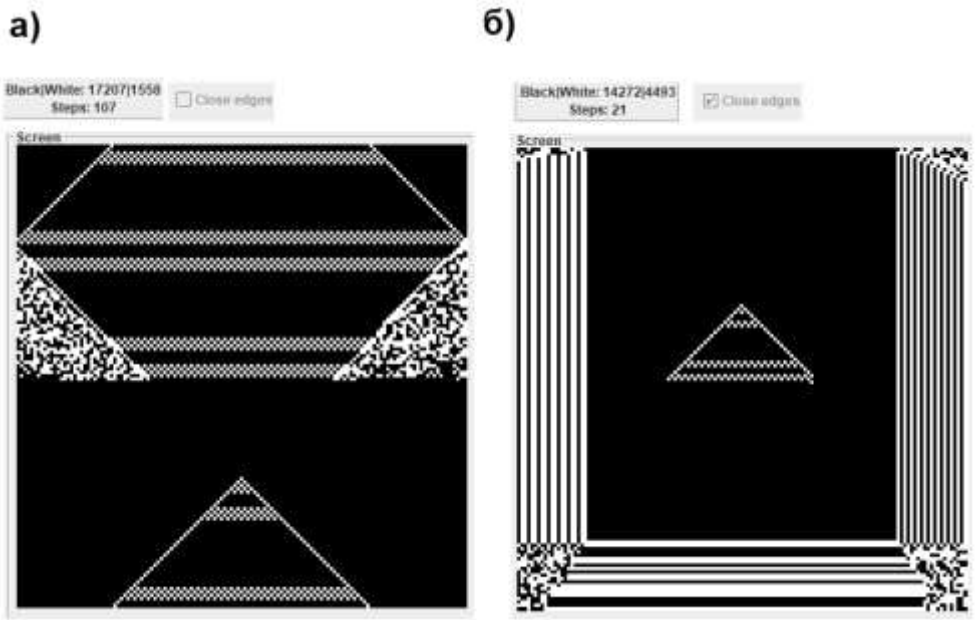

Рисунок 2.1 – Правило 2361597081

Перша проблема – це визначення сусідів для клітинок, що знаходяться на границі решітки КА. Ця проблема виникає, якщо не замикати границі решітки між собою. Якщо за межами решітки встановлювати додаткові клітинки з нульовим станом, то вони можуть досить сильно вплинути на роботу КА. На рисунку 2.1, що взятий з роботи [12], показана робота правила 2361597081

(подання у форматі коду Вольфрама) із замкнутими вертикальними та горизонтальними границями у тор (рисунок 2.1.а) та із закритими границями, за межами яких встановлені клітинки з нульовим станом (рисунок 2.1.б). Цей КА працює на двовимірній решітці за певним правилом, у якому наявність нульового стану (білого кольору) в будь-якому місці сильно впливає на роботу КА. В самому початку була лише одна біла клітинка в центрі поля, з якої КА почав рости. Але у варіанті б) такі клітинки були й за межами поля, що набагато прискорило заповнення поля білим кольором. У варіанті а) показаний КА на 107 кроці, а варіант б) показує лише 21 крок.

Можна уникнути цієї ситуації, якщо встановити за межами решітки чорні клітинки, але навіть вони будуть певною мірою впливати на роботу такого правила 2361597081. В деяких ситуаціях цим впливом можна знехтувати, а в деяких залишається лише замикати границі.

Друга проблема – це невелика відстань, на якій КА може отримати доступ до інших КА та якимось чином вплинути на їхній стан. Для розв’язання цієї проблеми було придумано багато способів, про які буде описано далі.

Перше розв’язання цієї проблеми запропоноване у роботі [10]. Ця робота досліджує модель, в якій на кожному кроці люди надають перевагу проголосувати за того чи іншого кандидата під час передвиборчої кампанії. Для цього тут використані автомати з пам’яттю, які на кожному кроці випадковим чином перебудовують зв’язки між собою. Тут наявна матриця ймовірностей переходів між типами клітинок для кожного кроку. Також автомати з часом можуть забувати досвід (зв’язки з іншими КА та який тип ці КА мали) або зберігати його весь час. Забування в цій моделі реалізовуються кількістю кроків, яку пам’ятає КА.

Інший варіант взаємодії між КА розглянутий у роботі [9], яка моделює ріст та поведінку багатоклітинних організмів. Для такого моделювання необхідні кілька видів зв’язку між однаковими агентами, які грають роль живих клітин. Перший тип зв’язку – це прямий контакт одної клітини організму з іншою по одній зі сторін шестикутних клітинок решітки. Окрім такого зв’язку в

цій моделі використовуються два типи зв’язку за допомогою сигнальних речовин. Перший тип сигнальних речовин може вільно поширюватися по всьому полю, а другий лише по незайнятих клітинках поля. Саме поле характеризується «густиною середовища», що моделює зміну кількості сигнальних речовин через певну відстань. Ця модель має такі вбудовані параметри, зміни яких досліджувалися: кількість виділення сигнальних речовин, чутливість рецепторів та кількість сусідів. В цій моделі ріст починається з одної клітини, яка є «зиготою» живого організму. Під час росту часто можуть траплятися ситуації, коли клітина повинна зробити один з кількох виборів, і обраний нею вибір є чисто випадковим.

Ще один спосіб спілкування між агентами розглянутий в роботі [13]. Тут присутні модифіковані КА, які називаються «клітинний автомат з локаторами». В цій моделі присутнє середовище «ефір», куди кожен КА може відправити певний сигнал з кінцевого алфавіту, який отримають всі інші КА. Це дозволяє одночасно спілкуватися з агентами, які знаходяться дуже далеко від певного КА. Також КА може отримувати сигнали лише з певних визначених напрямків.

Розглянемо, як такий тип автоматів з локаторами може розв’язувати деякі типи задач. Перша така задача – це задача синхронізації стрільців. На певному полі КА можуть перебувати в стані 0, 1 або 2. Ця задача вимагає, щоб в якийсь момент часу всі активні клітинки перейшли в стан 2 одночасно. Стандартне рішення пропонує використовувати дві хвилі зміни станів з одного джерела, одна з яких рухається втричі швидше, ніж інша. Швидша хвиля доходить до іншого краю поля, відбивається та зустрічається з повільною хвилею посередині. Потім ці 2 хвилі діляться на 4 і рухаються в різні боки від центру. Процес повторюється, і кількість хвиль подвоюється, поки довжина відрізків ряду не буде дорівнювати 1. І в цей момент всі солдати стріляють. Такий метод розв’язку вимагає 3r часу для r солдат.

КА з локаторами може розв’язати задачу синхронізації стрільців лише за два такти за допомогою трьох станів та двох сигналів.

Інша задача – це побудова найкоротшого шляху. Є дві активні клітинки, між якими повинен утворитися стабільний шлях. Звичайний КА може розв’язати таке за допомогою 14 станів. Водночас КА з локаторами може зробити це не більше ніж за 4 такти, для чого йому потрібно лише два стани та два сигнали. У роботі [13] наведено доведення такого розв’язку.

2.2 Стан

В момент часу класичний КА знаходиться в якомусь одному стані з кінцевої множини станів. У найпростішому випадку набір станів складається з двох станів: 0 та 1, що графічно позначаються білим та чорним кольором відповідно. Два стани – це мінімально можлива кількість станів для КА. Обмежень на максимальну кількість станів КА немає. КА може мати будь-яку кількість станів, яка необхідна для розв’язання певної обчислювальної задачі в рамках конкретної моделі. Стани позначаються цифрами, буквами або іншим набором символів. На початку обчислень кожен КА знаходиться у певному стані, який може змінюватися в ході обчислень.

Найчастіше прості моделі використовують КА з двома станами. В попередньому пункті для розв’язання задачі синхронізації стрільців та побудови найкоротшого шляху використовувалися КА з 3 та 14 станами відповідно. Робота [7] досліджує модель про взаємодії жертви та хижака, що побудована на основі КА з восьми станів (0 – порожня клітинка, 1 – жертва, 2-7 – стани життя хижака).

Збільшення кількості станів дозволяє розв’язати більшу кількість задач, але, як правило, це не особливо ускладнює поведінку КА. Для прикладу, робота [8] досліджувала одновимірний КА з трьома станами (кольорами). В результаті дослідження було з’ясовано, що приблизно 85% триколірних КА формують регулярні візерунки, що не виглядають набагато складнішими, ніж візерунки автомата з двох кольорів. Лише невелика кількість КА з будь-якою кількістю станів може створювати складні та випадкові візерунки. Але деякі автомати зі складною випадковою поведінкою можуть з часом вироджуватися в регулярні

візерунки. Наприклад, правило 1599 для триколірного автомата переходить у 31 повторюваних структур після 8282 кроків.

2.3 Правила

У кожного КА повинен існувати набір правил, за яким будуть відбуватися обчислення. Заведено вважати, що всі агенти певного КА використовують один і той же набір правил, що визначає поведінку КА. Кожне правило з цього набору показує, як зміниться стан певного агента на наступному кроці. За цими правилами кожен агент обраховує свій стан, для чого він використовує стани інших агентів, з якими він пов’язаний, та свій стан (це не обов’язкова умова). Зазвичай, правила для певного КА впливають лише на його стан та враховують лише його стан та стан інших агентів.

У роботі [8] виділяються такі чотири класи правил для КА:

- Всі клітинки дуже швидко переходять в однаковий стабільний стан.

- Всі клітинки швидко стабілізуються або можуть періодично змінювати свої стани.

- Утворюються хаотичні та неперіодичні структури, які дуже чутливі до змін.

- Утворюються складні структури, що взаємодіють між собою та існують довгий час.

Більша частина правил для будь-якої кількості станів КА належить до першого та другого класу, і лише невелика кількість відноситься до третього та четвертого класу.

Робота [8] розглядає правила КА по-іншому та вводить клас КА, який називається «мобільний автомат». У цій роботі досліджується поведінка цього класу КА на одновимірній решітці. Мобільний КА характеризується тим, що на певному кроці його роботи оновлюються не всі клітинки, а лише певна їх кількість, які називаються «активними» клітинками.

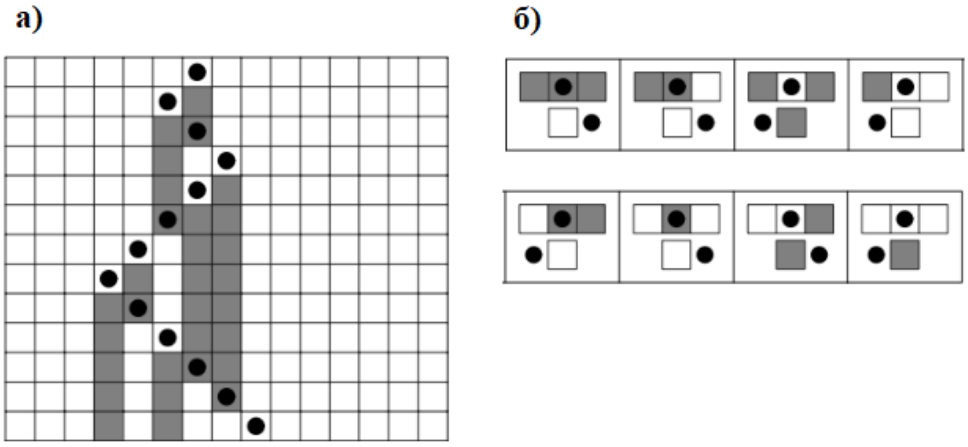

Рисунок 2.2 – Мобільний автомат

На рисунку 2.2.б) показаний приклад правил для мобільного автомату, що дозволяють утворити таку структуру, яка показана на рисунку 2.2.а). Цей набір правил задає лише одне активну клітинку, що переміщається по одномірній решітці КА.

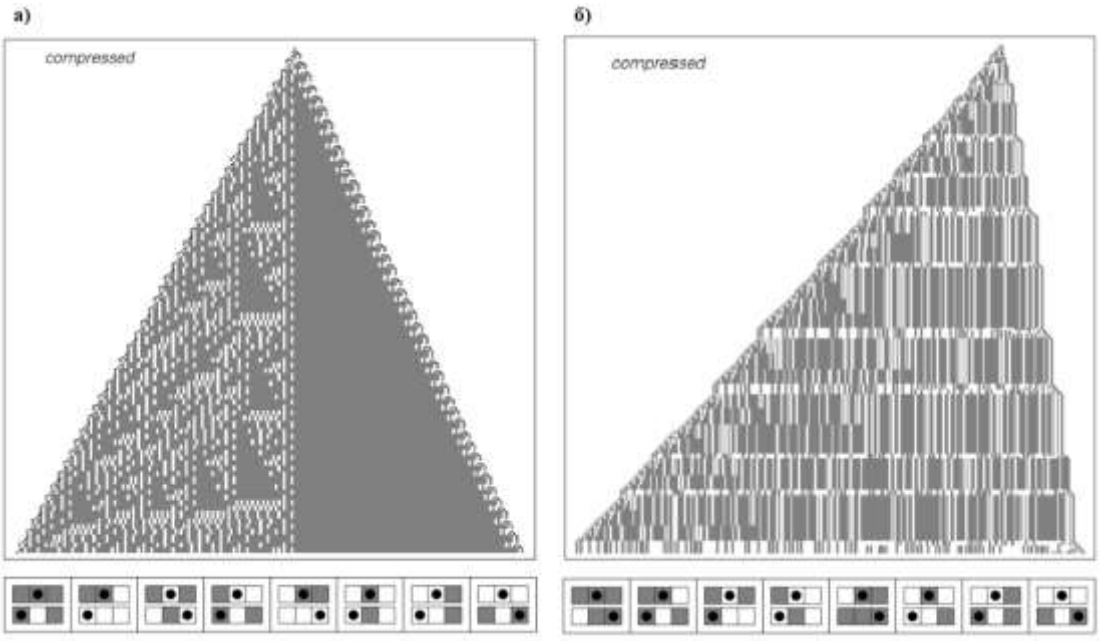

Рисунок 2.3 – Приклади випадкових автоматів (стиснутий вигляд)

Якщо дозволити мобільним автоматам оновлювати не лише стан активної клітинки, а і її прямих сусідів, то понад 99% таких КА покажуть просту поведінку, що повторюється. Але й такі КА можуть генерувати складні й випадкові структури, як показано на рисунку 2.3.

Також можна дозволити бути активною не лише одній клітинці, а декільком. Це збільшить кількість випадків, в яких зустрічається складна

поведінка. Але все ж досі велика кількість КА має просту поведінку, серед якої часто зустрічаються повторювані та вкладені структури. Тому якщо збільшувати складність правил КА, то в загальному його поведінка рідко стає набагато складнішою.

У звичайній ситуації після визначення набору правил можна чітко прорахувати, який результат покаже КА через певну кількість кроків. Але таке працює лише до тих пір, поки ймовірність застосування певного правила дорівнює 100%. У роботі [12] було розроблено програмне забезпечення, яке дозволяло встановити ймовірність неспрацювання правила або змішати між собою два правила. Змішування правил було впроваджено таким чином, що при певній ймовірності p, яку обирав користувач, спрацьовувало одне правило, та з ймовірністю 1 - p спрацьовувало друге правило. Ця система дозволила перевірити стійкість деяких структур, які утворювалися внаслідок дії правил КА.

У роботі [11] була використана ймовірність для переходу агентів з одної клітинки на іншу. Для цього агент обраховував ймовірність для всіх чотирьох своїх сусідніх клітинок, навіть якщо вони були зайнятими, а потім переходив на одну з цих клітинок. Якщо обрана ним клітинка була зайнята, агент міг обрати іншу клітинку або залишитися на місці. Ця система працювала на двовимірній решітці у синхронному режимі.

2.4 Режим перегляду

Існує два основні режими перегляду агентів: синхронний та асинхронний режим. Синхронний КА має обрахувати нові стани всіх агентів і лише потім ці агенти змінять свій стан. Однак асинхронний КА обраховує стан одного агенту та одразу ж змінює його. Такі два принципово різні підходи використовуються для різних цілей та задач.

Для синхронного режиму не важливий порядок обчислення нових станів його агентів, бо всі агенти одночасно змінять свій стан. Для нього необхідна окрема структура, куди будуть записані нові стани агентів, які потім замінять їх старі стани. Це потрібно, щоб агенти використовували для обчислень старі

стани інших агентів, а не їх змінені стани (як це використовується в асинхронному режимі).

В асинхронному оновленні важливий порядок обчислення. Робота [5] поділяє асинхронні методи на два типи: крокозалежні та часозалежні. В першому випадку порядок клітинок чітко визначений певним алгоритмом, тоді як в другому кожному агенту присвоюється час, коли він оновиться наступний раз. Крокозалежний алгоритм може мати фіксований або змінний (випадковий) порядок перегляду агентів. Випадковий порядок встановлюється для кожної ітерації роботи КА наступним чином: спершу випадково обирається клітинка, а потім обирається нова клітинка з решти клітинок. При цьому ймовірність обирання всіх клітинок однакова. Ці методи дозволяють переглянути всі клітинки КА в асинхронному режимі. Також може існувати асинхронний режим, що за одну ітерацію переглядає не всі клітинки.

Можливий перегляд лише деяких клітинок, що визначенні певними правилами або програмно, як було з мобільними автоматами. У роботі [8] були використанні синхронні мобільні автомати.

2.5 Використання агентів

Агенти та клітинні автомати загалом використовуються для моделювання багатьох процесів. Але їх використання не обмежується лише цим. КА можливо використовувати буквально для обчислення певних завдань. Так Поль Рендель у 2000 році [15] створив такий КА, що імітував роботу машини Тюрінга. Цей КА мав 3 можливих стани та міг обробляти 3 символи на стрічці.

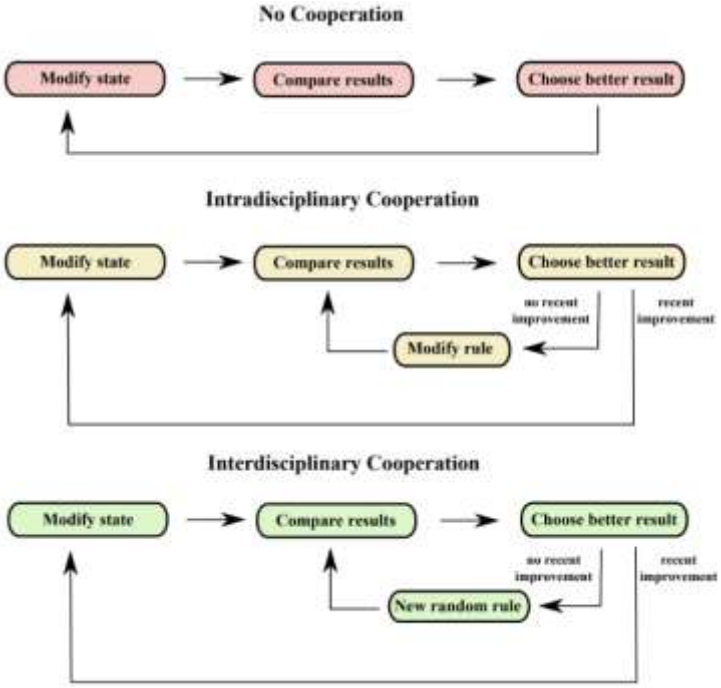

Рисунок 2.4 – Стратегії

Інше дослідження [1] вивчало питання генерування знань за допомогою КА та порівнювало різні стратегії (рисунок 2.4). Тут встановлювалася певна ціль (стан стрічки КА), яка мала бути досягнута з початкового стану за певну кількість кроків. Для виконання цього був обраний спосіб вимірювання успіху спроби (двійкове порівняння поточного стану з цільовим).

Перша стратегія – стратегія без співпраці. Ця стратегія змінювала випадковий біт у початковому стані S1, утворюючи стан S2, вимірювала близькість станів S1 та S2 до цільового та обирала найкращий з них. Це повторювалося 500 разів і фінальним успіхом був останній стан. Тобто, ця стратегія обирала одне правило та шукала краще розв’язання проблеми, змінюючи лише початковий стан.

Друга стратегія – внутрішньодисциплінний метод, що значно покращував результат. Якщо спроба досягти успіху провалилася після зміни початкового стану, то модифікувалося правило (змінювалося на один біт), після чого обиралося краще правило. Якщо ж результат покращився, то модифікувався початковий стан.

Третя стратегія – міждисциплінарний метод. Якщо результат не покращився, то обиралося випадкове правило. Порівнювалися два правила, і гірше відкидалося. Якщо результат покращився, то модифікувався початковий стан.

Це дослідження намагалося розв'язати за допомогою цих стратегій три типи проблем: просту (10 клітинок), середню (20 клітинок) та складну (100 клітинок). Кожна стратегія мала розв'язати 1000 однакових проблем.

Для розв'язання простих проблем найкраще показав себе третій підхід, коли він повністю розв'язав 30% всіх проблем, тоді як перший та другий підхід розв'язали 10% та 12% відповідно.

Але перший, другий та третій підхід розв'язали лише 6.1%, 5.2% та 2% середніх проблем відповідно. Також ці підходи розв'язали 16.5%, 8.7% та 6.5% складних проблем відповідно. Причину цього можна пояснити тим, що другій та третій стратегії потрібно більше кроків, щоб досягти цілі.

Таким чином другий та третій підхід є більш гнучкими та можуть розв'язати більшу кількість задач, але це потребує багато часу. Перший підхід найкраще розв'язує складні задачі, бо він покладається на відоме правило, а не шукає нові правила.

Проте все залежить від правила, яке було обране першим підходом або іншими двома. Деякі правила можуть досягнути цільового стану швидше за інші або вони достатньо гнучкі, що дозволяє їм досягти багатьох цільових станів (деякі правила не зможуть досягти цільового стану при будь-яких змінах початкового стану).

ВИСНОВКИ

Отже, ця робота розглянула складові компоненти клітинних автоматів та їх використання для різних систем. Було детально вивчено можливості змінювати певні характеристики клітинних автоматів та області, де такі зміни можуть бути використані. Також було розглянуто модифікації клітинних автоматів, що спрощували розв’язання завдань різного роду.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Jäger G. Using elementary cellular automata to model different research strategies and the generation of new knowledge. *Complex systems*. 2018. Vol. 27, no. 2. P. 145–158.
2. Toffoli T., Margolus N. H. Cellular automata machines. *Complex systems*. 1987. Vol. 1, no. 5. P. 967–993.
3. Kier L., Seybold P. Cellular automata modeling of complex biochemical systems. *Encyclopedia of complexity and systems science*. 2009. P. 848–865.
4. Kier L. B., Cheng C.-K., Testa B. Cellular automata models of biochemical phenomena. *Future generation computer systems*. 1999. Vol. 16, no. 2-3. P. 273–289.
5. Schönfisch B., de Roos A. Synchronous and asynchronous updating in cellular automata. *Biosystems*. 1999. Vol. 51, no. 3. P. 123–143.
6. Benati S. A cellular automaton for the simulation of competitive location. *Environment and planning B: planning and design*. 1997. Vol. 24, no. 2. P. 205–218.
7. Ermentrout G., Edelstein-Keshet L. Cellular automata approaches to biological modeling. *Journal of theoretical biology*. 1993. Vol. 160, no. 1. P. 97–133.
8. Wolfram S. A new kind of science. Wolfram Media, Inc., 2002.
9. Марков М. Использование клеточных автоматов для моделирования онтогенеза. *Молодой учёный*. 2010. I, № 5. С. 118–125.
10. Алёшкин А., Обухова А., Жуков Д. Математическое и программное обеспечение стохастических клеточных автоматов с памятью. *Современные информационные технологии и ИТ-образование*. 2017. Т. 13, № 2. С. 25–39.
11. Витова Т. Б. Построение и исследование клеточно-автоматной стохастической модели движения людей. Красноярск, 2017. 171 с.

12. Войцеховський Є.О. Дослідження стохастичної поведінки клітинних автоматів : Курсова робота. Київ, 2021. 34 с.
13. Гасанов Э. Э. Клеточные автоматы с локаторами. *Интеллектуальные системы. Теория и приложения*. Т. 24, № 2. С. 119–132.
14. Простейшие клеточные автоматы и их практическое применение. *Хабр*. URL: https://habr.com/ru/post/273393/ (дата звернення: 09.06.2022).
15. Rendell P. This is a turing machine implemented in conway's game of life. Rendell-Attic. URL: http://rendell-attic.org/gol/tm.htm (дата звернення: 09.06.2022).