# **Огляд літератури**

**1.1 Загальні відомості**

**Основні поняття клітинних автоматів.**

Клітинний автомат (КА, cellular automata) — це сукупність "кольорових" клітин у сітці заданої форми, яка розвивається через декілька дискретних етапів часу відповідно до набору правил, що базуються на станах сусідніх клітин. Правила еволюції застосовуються ітеративно для будь-якої кількості ітерацій.

Існують також інші визначення. Найпоширенішими є:

* математичний об'єкт з дискретним простором та часом;
* регулярна структура двійкових скінченних автоматів з однаковими правилами переходів, що виражені у вигляді булевих функцій від станів сусідніх автоматів;
* стилізовані, синтетичні світи, що визначені простими правилами, подібно правилам настільної гри;
* математична ідеалізація фізичної системи, в якій час та простір дискретні, а фізичні величини приймають скінченну множину значень.

Властивості класичних КА:

* зміна значень всіх клітин відбуваються одночасно після обчислення нового стану кожної клітини решітки.
* решітка однорідна. Неможливо відрізнити жодні два місця на решітці по ландшафту.
* взаємодії локальні. Лише околишні клітинки (як правило, сусідні) здатні вплинути на дану клітинку;
* множина станів клітинки кінцева. Ця умова потрібна, щоб для отримання нового значення стану клітини треба було виконати кінцеву кількість операцій.

Якщо з будь-якого початкового стану можна привести клітинного автомат в будь-яку задану конфігурацію шляхом варіювання значення вхідного параметра, такий КА називають повним.

У кожний момент часу кожен елемент КА приймає один стан зі скінченного набору станів. В залежності від цих станів в наступний момент часу набір елементів може прийняти новий стан. Якщо для елементів КА множини можливих станів відрізняються, такий клітинний автомат називається полігенним. Але на практиці використовуються комірки з еквівалентною множинами можливих станів алгебраїчною структурою – лінійні КА.

Елементи можуть бути геометрично розташовані різноманітним чином. Розмірність простору може бути довільною, а число елементів – як безкінечним, так і скінченним. В останньому випадку виникає додаткова міра свободи в граничних умовах. Вони можуть бути різними, але на практиці використовуються постійні у часі (найчастіше – нульові) або періодичні граничних умовах. У динамічних КА геометрія може змінюватися з часом, а якщо геометрія різна на різних ділянках простору, такі кліткові КА називають неоднорідними.

Сусіди – інші елементи, від яких залежить елемент КА. Можна назвати поняття сусідства ключовим для КА. Сусідство у більшій мірі визначається геометрією КА. Для різних цілей можлива зміна числа вхідних станів елемента.

Для кожного клітинного автомату визначені певні локальні правила. Відповідно до локального правила змінюється стан елемента КА протягом часу. КА, в якому локальні правила різні для різних елементів, називається різнорідним. Локальне правило може бути недетермінованим, тобто змінюватися в часі або мати випадкову природу. Зазвичай, це правило встановлює залежність між поточним станом елементу та його сусідів та майбутнім станом цього елементу на наступному кроці.

Основний напрям дослідження клітинних автоматів — алгоритмічна розв'язність окремих задач. Також розглядаються питання побудови початкових станів, при яких клітинний автомат вирішуватиме задану задачу. Залишається відкритим, наприклад, питання про можливість побудови машини Тюринга у грі «Життя».

**Класифікація клітинних автоматів.**

Стівен Вольфрам у своїй книзі A New Kind of Science запропонував 4 класи, на які всі клітинні автомати можуть бути поділені в залежності від типу їх еволюції. Класифікація Вольфрама була першою спробою класифікувати самі правила, а не типи поведінки правил окремо. В порядку зростання складності класи виглядають наступним чином:

* Клас 1: Результатом еволюції майже всіх початкових умов є швидка стабілізація стану та його гомогенність. Будь-які випадкові конструкції в таких правилах швидко зникають.
* Клас 2: Результатом еволюції майже всіх початкових умов є швидка стабілізація стану, або виникнення коливань. Більшість випадкових структур в початкових умовах швидко зникає, але деякі залишаються. Локальні зміни в початкових умовах надають локальний характер на подальший хід еволюції системи.
* Клас 3: Результатом еволюції майже всіх початкових умов є псевдо-випадкові, хаотичні послідовності. Будь-які стабільні структури, які виникають, майже відразу ж знищуються оточуючим їх шумом. Локальні зміни в початкових умовах надають широкий, невизначений вплив на хід усієї еволюції системи.
* Клас 4: Результатом еволюції майже всіх правил є структури, які взаємодіють складним і цікавим чином з формуванням локальних, стійких структур, які здатні виживати тривалий час. В результаті еволюції правил цього класу можуть виходити деякі послідовності Класу 2, описаного вище. Локальні зміни в початкових умовах надають широкий, невизначений вплив на хід усієї еволюції системи. Деякі клітинні автомати цього класу мають властивість універсальності по Тьюрингу. Останній факт був доведений для Правила 110 і гри «Життя».

Крім цього, клітинні автомати також класифіковані за іншими критеріями. За порядком переходу елементів в новий стан клітинні автомати поділяються на синхронні та асинхронні. У синхронних КА всі клітинки переходять у новий стан одночасно за сигналом глобального таймера. При цьому як вхідні стани використовуються старі стани сусідніх клітинок. У асинхронних КА клітинки переходять у новий стан у випадковому порядку, причому новий стан клітинки відразу може використовуватися її сусідами як вхідний.

За можливістю руху клітин автомати поділяються на рухливі та нерухомі. Рухливі КА характеризуються можливістю зміни положення клітинки в решітці під час еволюції системи. У нерухомих КА положення клітини під час еволюції залишається постійним.

За ступенем залежності від випадкових факторів автомати поділяють на детерміновані та імовірнісні. У детермінованих КА стан комірки n+1 в наступний момент часу однозначно визначається станом цієї клітинки і її найближчих сусідів у попередній момент часу. У цьому випадку стан даного елемента в момент часу n +1 є однозначною функцією F від двох змінних — стану цього елемента і суми станів його найближчих сусідів у попередній момент часу n. При такому визначенні клітинний автомат не має пам'яті. КА з пам'яттю можна отримати, припустивши, що функція F залежить, наприклад, також від стану елемента в ще більш ранній момент часу.

КА, в яких стани комірок в наступний момент часу визначаються на основі деяких ймовірностей, називаються імовірнісними КА (ІКА). У класичних ІКА правила переходів мають абстрактний характер і не пов'язані однозначно з реальними процесами, що відбуваються в модельованій системі. У таких автоматах при моделюванні процесу для кожної клітинки датчиком випадкових чисел генерується випадкове число Q (0 < Q < 1), що порівнюється з імовірністю w реалізації цього процесу. Якщо Q < w, то процес реалізується.

Іноді використовуються правила, записані у вигляді звичайних диференціальних рівнянь (клас КА-ЗДР). У цьому випадку стани комірок задаються набором змінних, значеннями яких можуть бути будь-які дійсні числа. Для таких автоматів диференціальні рівняння розв'язуються для кожної комірки окремо протягом фіксованого відрізку часу, при цьому кожна клітинка може мати різні початкові умови. Цей клас КА дуже щільно прилягає до диференціальних рівнянь в частинних похідних.

Моделі типу КА-ЗДР займають проміжний стан між КА і ІКА, а також між простими КА і ДР в частинних похідних. Основною ідеєю КА-ЗДР є розбиття модельованої області на рівновеликі комірки і розв'язання системи ЗДР незалежно в кожній клітинці з різними початковими умовами. У деяких моделях просторове розташування комірок неістотне, а в інших кількість сусідніх комірок і розмірність простору відіграють вирішальну роль (випадки поширення хвиль або виникнення стаціонарних просторових структур у нерухомому середовищі). У моделях КА-ЗДР передбачається, що клітинка містить дуже велику кількість частинок, що дозволяє застосовувати ЗДР і неперервні функції. Ця обставина залишає тільки один спосіб для моделювання дифузії, а саме просте опосередкування концентрації по сусіднім коміркам.

За структурою КА поділяють в залежності від кількості вимірів. Найбільш вживані одно- та дво-вимірні.

Як ґратки беруть поле, комірки якого є трикутники, чотирикутники чи шестикутники.

В одновимірному (лінійному) КА решітка являє собою ланцюжок клітинок (одновимірний масив), в якій для кожної з них, крім крайніх, є по два сусіди. Для усунення крайових ефектів решітка «загортається» у тор. Це дозволяє використовувати наступне співвідношення для всіх клітин автомата:

y '[i] = f (y [i-1], y [i], y [i +1]),

де f – функція переходів клітинки;

y '[i] – стан i-ої клітинки в наступний момент часу;

y [i-1] – стан (i-1)-ої клітинки в даний момент часу;

y [i] – стан i-ої клітинки в даний момент часу;

y [i +1] – стан (i +1)-ої клітинки в даний момент часу.

У двовимірному (площинному) КА решітка реалізується двовимірним масивом. У ній кожна клітина має вісім сусідів. Для усунення крайових ефектів решітка так само, як і в попередньому випадку, «загортається» у тор. Це дозволяє використовувати наступне співвідношення для всіх клітинок автомата:

y '[i] [j] = f (y [i] [j], y [i-1] [j], y [i-1] [j +1], y [i] [j +1] , y [i +1] [j +1], y [i +1] [j], y [i +1] [j-1], y [i] [j-1], y [i-1] [j-1]).

Варіюючи різні параметри, можна отримати КА необхідної конфігурації. Гнучкість конфігурації та універсальність обчислень забезпечили високу популяризацію кліткових автоматів у різних сферах. Свобода у виборі параметрів конфігурації дуже зручна для використання, але це накладає додаткову складність у класифікації та систематизації знань теорії кліткових автоматів. Тим не менше, найбільш використовуване на практиці лише невелике сімейство конфігурацій кліткових автоматів. Як правило, кожен з них має свою назву. Наведено невеликий список найбільш використовуваних варіантів конфігурацій:

* Мозаїчний автомат. КА, що використовує у локальному правилі кожного елемента не тільки стан елемента та його сусідів, але і значення загального вхідного параметра, який може змінюватися час від часу. Зміна цього параметра веде до перевизначення набору правил зміни станів у всьому просторі елементів КА. Якщо з будь-якого початкового стану можна привести клітковий автомат в будь-яку задану конфігурацію шляхом варіювання значення загального вхідного параметра, такий КА називають повним.
* Ітеративний автомат. КА, в якому лише один елемент використовує для зміни свого стану значення вхідного параметра
* Односторонній клітковий автомат. Такий автомат припускає лише односторонню взаємодію елементів. Наприклад, в одновимірному масиві елементів значення кожного елемента залежить лише від його стану і від стану лівого (або правого) сусіда. Незважаючи на удавану вироджуваність звичайного КА, односторонні КА досить універсальні і використовуються для розпізнавання мовних форм.
* Л-система. Цей тип КА використовується для моделювання біологічних систем. Це динамічні КА (як правило, одно- чи двовимірні), в яких з часом один елемент може замінятися декількома або може бути видаленим із системи згідно з заданими правилами.
* Відмовостійка система. У таких системах моделюється робота КА в реальних умовах: з деякою ймовірністю кожен елемент КА може перейти в стан, що не відповідає локальному правилом. Завданням є створення алгоритмів, для яких робота КА буде правильною в незалежності від

допущених помилок.

**1.2 Гра “Життя”.**

**Основні правила гри “Життя”.**

Гра «Життя» (англійською Conway's Game of Life) – двовимірний клітинний автомат з квадратною сітку та двома можливими станами клітини: «жива» та “мертва”. Кожна клітина має 8 сусідів. Існують два основних правила: умова загибелі та умова народження.

* Якщо мертва клітина має рівно 3 сусіда, на наступному кроці вона оживає
* Якщо жива клітина має менше 2 чи більше 3 сусідів, вона оживає

Гра “Життя” відноситься до клітинних автоматів 4 класу, містячи складні постійні структури та хаотичне розповсюдження клітин навіть при незначних змінах. Було доведено, що цей клітинний автомат є Тьюринг-повним, тобто за допомогою нього можна реалізувати розрахунок майже будь-якої математичної операції. Його дослідженням займаються вже декілька поколінь вчених, знаходячи все складніші стійкі структури, що мають певні цікаві особливості.

Частина поля, яка містить живі та неживі клітини, називається конфігурацією клітинного автомату.

**Конфігурація «натюрморт»**

Натюрморти - конфігурації «Життя» або іншого клітинного автомата, які не змінюються в процесі еволюції.

Приклади:

Найбільш поширений натюрморт - блок - конфігурація у формі квадрата 2 × 2. Два блоки, розміщені в прямокутнику 2 × 5, утворюють бі-блок - найпростіший псевдонатюрморт. Блоки використовуються в якості складових частин у безлічі складних пристроїв, наприклад, в планерній рушниці Госпера.



Рис. 1.2.1 Блок та бі-блок

Другий за поширеністю натюрморт - вулик. Вулики часто виникають четвірками в конфігурації, що називають пасікою.

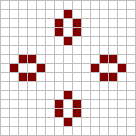


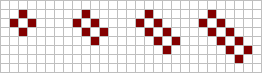
Рис. 1.2.2 Вулик та пасіка

Третій за поширеністю натюрморт - коровай. Короваї нерідко з'являються парами. У свою чергу, подвійні короваї також з'являються в парах, званих пекарнями.



Рис. 1.2.3 Коровай, подвійний коровай, пекарня

Ящик складається з чотирьох живих клітин в околиці фон Неймана центральної мертвої клітини. Додавання однієї живої клітини по діагоналі до центральної клітини перетворює ящик в човен, а додавання симетрично ще однієї клітини - в корабель. Природне подовження цих трьох конфігурацій дає баржу, довгий човен і довгий корабель відповідно. Подовження можна продовжувати як завгодно довго.



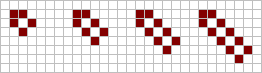
Рис. 1.2.4 Ящик, баржа та її подовження

Рис 1.2.5 Човен, довгий човен та їх подовження

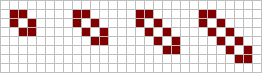


Рис. 1.2.6 Корабель, довгий корабель та інші його подовження

**Конфігурація «осцилятор».**

Осцилятор - конфігурація клітинного автомата, яка після кінцевого числа поколінь повторюється в тій же самій орієнтації і позиції. Іншими словами, осцилятор - це будь-який зразок, який є попередником самого себе. Еволюція осцилятора триває як завгодно довго.

Залежно від контексту, космічні кораблі також можуть вважатися осцилляторами, але зазвичай вони розглядаються окремо.  
Мінімальна кількість поколінь, через яке осцилятор повертається у вихідну конфігурацію, називається періодом осцилятора. Осцилятор з періодом 1 зазвичай називається стійкою конфігурацією, так як він не змінюється. Таким чином, натюрморти є підмножиною осциляторів.

Прикладом осцилятору може слугувати, зокрема, фумарола. Її період складає 5 кроків:



Рис 1.2.7 Фумарола

Знайдено й велику кількість інших осциляторів (наприклад, галактика Кока з періодом 8 чи пентадекатлон з періодом 15). Зокрема, для створення нового осцилятору достатньо об’єднати два вже відомих осциляторів в одну конфігурацію. Таким чином, можна зробити висновок, що на нескінченному полі кількість можливих осциляторів також нескінченна.

Осцилятори є прикладом більш складної стабільної структури, яка зазвичай відповідає 2 класу клітинних автоматів, проте здатна існувати у клітинному автоматі, що проявляє ознаки хаотичності.

**Конфігурація «космічні кораблі».**

Конфігурація «Життя» або іншого клітинного автомату називається космічним кораблем, якщо через певну кількість поколінь вона знову з'являється без доповнень або втрат, але зі зміщенням відносно вихідного положення.

Першим виявленим космічним кораблем став планер. Планер був знайдений під час відстеження еволюції R-пентаміно (стартової комбінації з п’яти сусідніх заповнених клітин, що розпочинає найдовшу еволюцію серед усіх можливих таких комбінацій) в 1970 році Річардом Гаєм. Еволюцію планеру можна побачити на зображеннях нижче:

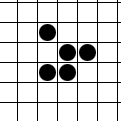
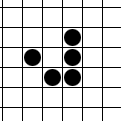
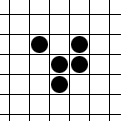
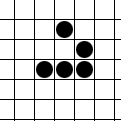


Рис. 1.2.8 Фумарола

Космічний корабель періоду p, Який переміщується на (m, n) протягом його періоду, де m ≥ n, має тип (m, n) / p. Як було доведено Конуеєм в 1970 році, p ≥ 2m + 2n.

Космічні кораблі, що рухаються по горизонталі або вертикалі, називаються ортогональними кораблями. Якщо рух космічного корабля відбувається по діагоналі під кутом 45 °, такий корабель називається діагональними. Космічні кораблі, що рухаються під іншими кутами, називаються косими або наклонними. У 2010 році був сконструйований перший наклонний космічний корабель типу (5120, 1024) / 33699586.

Швидкістю світла в заданому клітинному автоматі називають найбільшу швидкість поширення інформації. Швидкість світла в «Житті» дорівнює швидкості переміщення шахового короля - швидкості в одну клітку за покоління по горизонталі, вертикалі або діагоналі. Зазвичай швидкість світла позначається літерою c.

Швидкість космічного корабля визначається відношенням відстані зміщення до періоду. Часто швидкість виражається через c. Так, швидкість планера в «Життя» дорівнює c / 4, так як він переміщується на одну клітину по діагоналі за чотири покоління. Найпростіший ортогональний космічний корабель, ВКК, рухається зі швидкістю c / 2.

У загальному випадку, якщо космічний корабель в двовимірному клітинному автоматі на квадратній сітці переміщується на вектор (x, y) через n поколінь, його швидкість дорівнює v = с \* max (| x |, | y |) / n.

Найбільш поширеними різновидами космчних кораблів, окрім планерів, є легкі, середні та важкі космічні кораблі:

Рис. 1.2.9 Легкі, середні та важкі космічні кораблі 

Крім цього, існує велика кількість більших космічних кораблів, як ортогональних, так і діагональних. Наприклад, ортогональний космічний корабель, виявлений Тімом Коу 11 червня 2016 року, що називається “макаронний монстр” - це перший космічний корабель швидкості 3с / 7. Він зображений на малюнку праворуч.

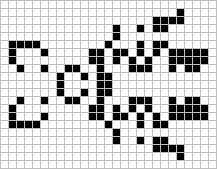
Існують певні структури, що тісно пов’язані з космічними кораблями. Тагалонг - конфігурація, яка не є сама по собі космічним кораблем, але може бути приєднана до космічного корабля, щоб сформувати новий космічний корабель. Іншими словами, тагалонг - це частина космічного корабля, яку можна видалити без руйнування корабля. Тагалонг, розташований перед космічним кораблем, називають пушалонгом.

Рис 1.2.10 Тагалонг для двох легких космічних кораблів

**1.3 Існуючі рішення**

**Реалізації двовимірних клітинних автоматів.**

Найбільш поширені реалізації автоматів: мураха Ленгтона та гра “Життя”. Приклад реалізації мурахи Ленгтона це реалізація Фабріса Вайнберга.

Функціонал:

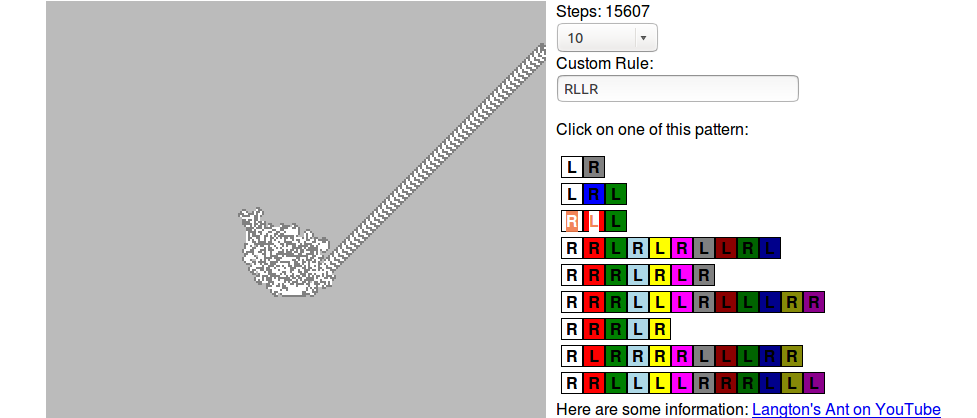
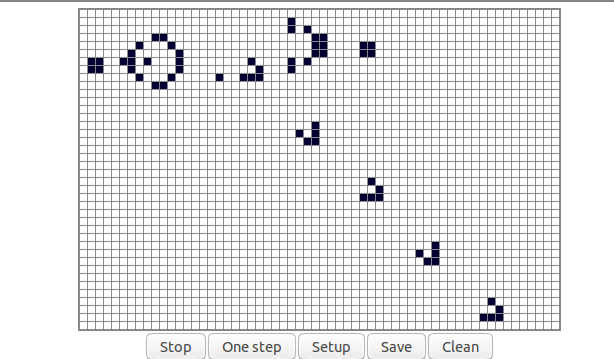
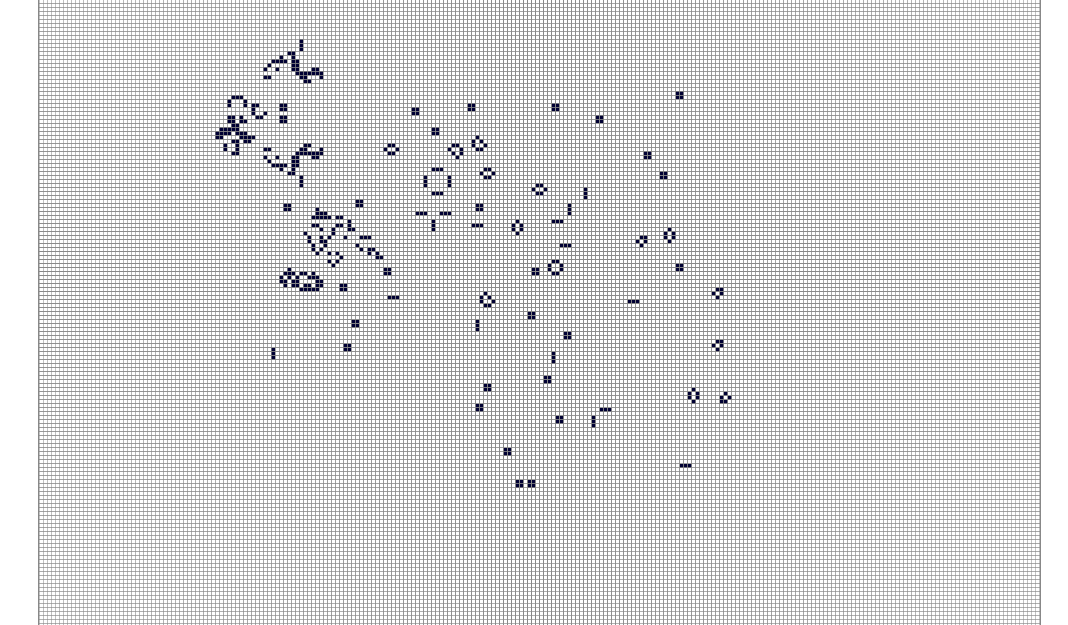
* Запустити класичний варіант мурахи і розширені варіанти з більшою кількістю станів
* Обрати швидкість, з якою мураха рухається автоматом, розглядаючи більш незначні кроки чи загальну картину.

Рис. 1.3.1 Приклад виконання програми із стандартними правилами:

У випадку ж гри “Життя” різноманітних реалізацій ще більше. Гру “Життя” програмно реалізовували ще на старих комп’ютерах 70-х років, проте зараз можливо створити реалізацію зі зручним інтерфейсом та додатковими можливостями. Одним з прикладів таких програм може слугувати реалізація на веб-сайті michurin.ru, що становить з себе JavaScript варіант автомату з можливістю налаштування розміру поля, швидкості переходу до нового стану, покроковому огляду еволюції, а також можливістю обрати одну з заздалегідь створених конфігурацій, серед яких велика кількість натюрмортів, осциляторів та космічних кораблів, а також рушниця Госпера. Крім того, як і багато інших реалізацій автомату, він дозволяє змінювати стан клітин під час виконання, вносячи корективи в роботу автомату чи, наприклад, продовжуючи еволюцію, що вже стала тривіальною

Рис. 1.3.2 Результат виконання планерної рушниці Госпера:

Рис. 1.3.3 Результат виконання з випадковими початковими умовами

**Реалізації тривимірних клітинних автоматів.**

сubes.io – WebGL-реалізація тривимірного клітинного автомату.

Переваги:

* зручний інтерфейс
* можливість регулювати форми початкової конфігурації
* можливість регулювати правила переходу до наступного стану
* змінювати стан клітин в процесі роботи
* змінювати формат відображення

Недоліки:

* відсутня можливість задавати власну початкову конфігурацію
* програма працює лише з автоматами, клітини в яких мають лише два можливих стани

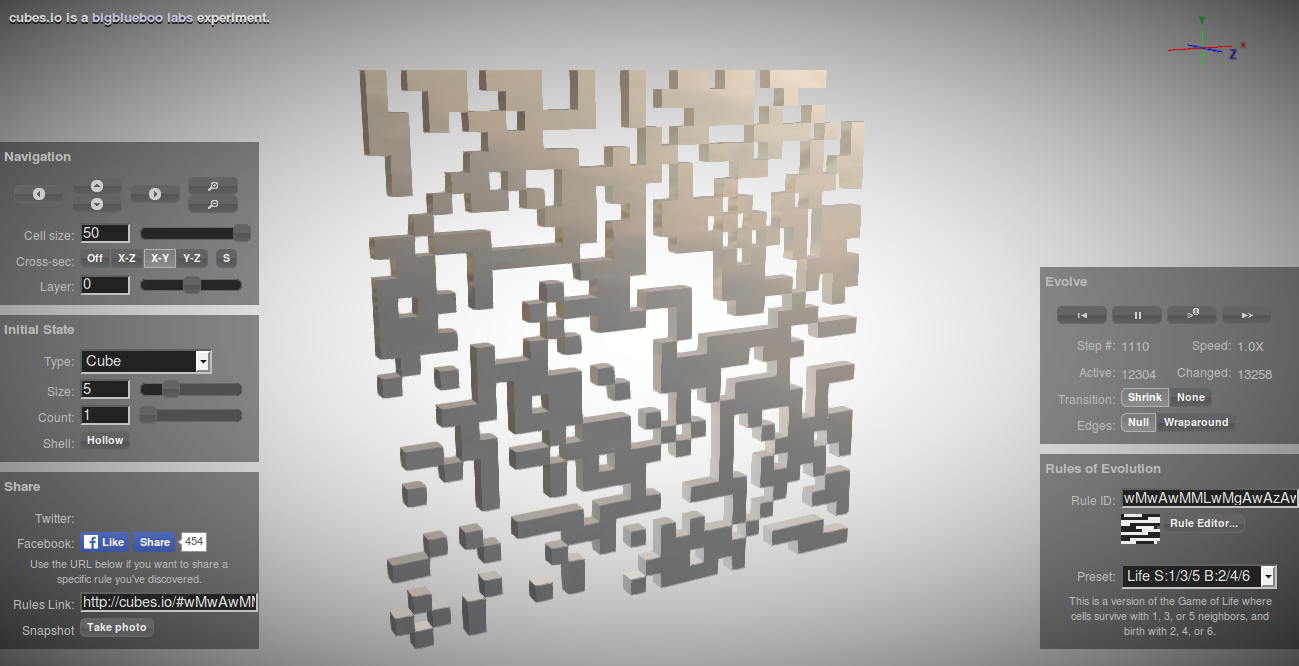


Рис. 1.3.4 Приклад виконання сubes.io

Таким чином, разом з можливістю призупинення та покрокового перегляду, застосунок надає можливість детально розглянути процес еволюції обраної конфігурації при даних правилах. Також програма динамічно обчислює освітлення та віддзеркалення зображення на поверхні клітин, що дозволяє чіткіше бачити окремі клітини та таким чином допомагає у візуалізації клітинного автомату. Навігацію камери реалізовано за допомогою миші та клавіатури, що дозволяє зручно повертати камеру на необхідний кут.

Загалом візуалізація тривимірного клітинного автомату на веб-сайті cubes.io є досить гарною репрезентацією процесів, що відбуваються в автоматах, що розглядаються, проте не дає достатнього контролю над їх еволюцією чи можливості керувати її та направляти в потрібному напрямку. Такі висновки дозволяють визначити ціль даної роботи як реалізацію системи, що дозволятиме в тому ж ступені коректно та зручно відображати тривимірні клітинні автомати.

**2. Алгоритм вирішення задачі**

**Основна ідея роботи.**

Необхідно створити програмний застосунок, здатний задовільнити потреби, що виникають при розгляді тривимірних клітинних автоматів. Зазначу, яких саме цілей необхідно досягти при розробці даного застосунку:

* Основною метою роботи програми є обрахунок еволюції тривимірних клітинних автоматів. У відриві від способу зображення клітинного автомату, обрахунок еволюції становить з себе процес визначення, які клітини мають загинути, а які — зародитись на наступній ітерації. Зазвичай для визначення наступного стану автомату використовуються чітко визначені правила, що у даному випадку базуються на кількості заповнених сусідів відповідної клітини. Так як розроблюваний застосунок підтримує ймовірнісні клітинні автомати, то клітина може зародитись, якщо вона має кількість сусідів, що дорівнює випадково визначеному для конкретного випадку числу, яке знаходиться в заздалегідь визначеному діапазоні. Границі загибелі клітини від самотності та перенаселення також визначаються випадково в кожному окремому випадку на основі заданих меж значень. Окрім цього, автомат може мати різний розмір, хоча й завжди матиме лише кубічну форму. Також важливо коректно обраховувати наступну ітерацію: зміни, що відбуваються при переході до наступної ітерації, не мають впливати на кінцевий результат, адже клітини, які змінять свій стан, визначаються лише на основі попереднього стану. Підтримка даного функціоналу є основною для застосунку, проте й найпростішою в розробці: загалом цей функціонал реалізується чітко заданими простими алгоритмами переходу до наступного стану.
* Даний застосунок має підтримувати коректне відображення процесу еволюції тривимірного клітинного автомату. Одною з основних функціональних можливостей застосунку, який потребується для досліждення, є можливість відображення як поточного стану автомату, так і процесу переходу до наступного стану. Основною проблемою в даному випадку є той факт, що, на відміну від двовимірних автоматів, відобразити на екрані відразу весь тривимірний автомат немає можливості. Тому необхідно вирішити проблему його відображення таким чином, щоб залишити можливість цілком розглянути структуру клітинного автомату незалежно від ітерації. При цьому велика кількість клітин з декількома можливими станами може ускладнити розуміння структури поточної ітерації та її зміни при переході до наступної ітерації. Тому необхідно забезпечити таку форму відображення, при якій окремі клітини можна чітко розрізнити, навіть при великій їх кількості, а перехід до наступного кроку дає чітке представлення, яка частина клітин змінила свій стан.
* Розпізнавання кінцевого стану клітинного автомату є проблемою, що постає не лише перед тривимірними, але й перед двовимірними клітинними автоматами. Вона полягає в тому, що клітинні автомати 4 класу часто припиняють свою еволюцію, переходячи до стабільного стану, що містить лише стійкі структури (наприклад, натюрморти та осцилятори у грі “Життя”). Проте зазвичай це не означає, що стан клітин автомату більше не змінюється. Зазвичай еволюція стає тривіальною, коли всі зміни відбуваються лише серед циклічних структур, що повторюються через певну кількість ітерацій. При формуванні великої кількості подібних структур повне повторення стану автомату може відбутись через декілька сотень ітерацій чи навіть більше. Тому розпізнавання цього етапу еволюції є складною задачею — необхідно порівняти поточний стан клітинного автомату з усіма попередніми ітераціями, адже якщо стан автомату повністю повторює один з минулих станів — це є головною ознакою зациклення. Застосунок повинен вирішувати цю задачу методом, що дозволить обчислювати тисячі ітерацій та перевіряти нетривіальність еволюції навіть для великих обмежених клітинних автоматів.
* Необхідно надати користувачу можливість змінювання стану клітин автомату в реальному часі. Однією з основних проблем програмних реалізацій тривимірних клітинних автоматів є складність їх задання за допомогою графічного інтерфейсу. Навіть якщо проблема відображення тривимірного автомату вже вирішена, можливість змінювання стану клітин у трьох вимірах потребує особливого підходу.
* Застосунок повинен надавати можливість зберігати поточний стан клітинного автомату та завантажувати його в разі необхідності. Подібний функціонал дозволить зберігати знайдені стабільні структури чи конфігурації, здатні до довготривалої еволюції, що значно перевищує складністю початковий стан.

Розглянемо детальніше, яким саме чином планується вирішити вищезазначені проблеми.

**Відображення стану тривимірного автомату.**

Відображення поточного стану тривимірного автомату зазвичай виконується за допомогою тривимірних моделей, що зображають окремі клітини. Зокрема, у випадку клітинного автомату з двома можливими станами клітини, наявність в певній позиції моделі вказує на “живий” стан відповідної клітини, а відсутність моделі — на “мертвий” стан. Розповсюдженим способом зображення заповненої клітини є звичайна модель кубу з певною текстурою та шейдерами для більш деталізованого відображення стану автомату.

При подібному способі відображення стану виникає необхідність прийняти певні рішення, що впливатимуть на те, наскільки легко буде розрізнити окремі клітини та загалом зрозуміти структуру автомату. Зазвичай доцільно розміщувати моделі впритул один до одного, таким чином представляючи неперервну послідовність клітин. Наявність між клітинами проміжку може спростити розпізнавання меж клітин, проте ускладнить розгляд пустих клітин, адже через наявність цього проміжку зрозуміти, скільки пустих клітин знаходиться між двома заповненими, становиться складніше. Ще одним можливим варіантом є явне зображення сітки навіть між пустими клітинами. Це остаточно вирішує проблему з підрахунком кількості пустих клітин між двома заповненими, однак створює іншу проблему: через велику кількість зайвих ліній розглядати структуру клітин, що знаходяться далеко від краю автомату, становиться набагато складніше. Через це подібний підхід не є доцільним. Проте для чіткого розпізнавання меж заповнених клітин ребра кубу відмічені чорним кольором, що дозволяє не використовувати проміжки між клітинами.

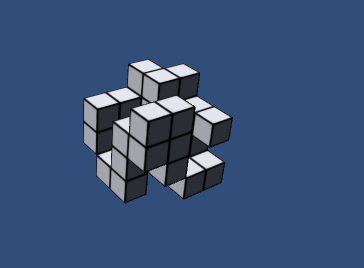
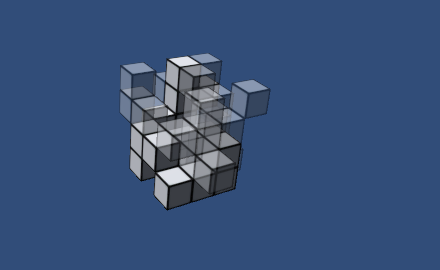


Рис.2.1 Приклад подібної системи відображення

Окрім вибору власне способу відображення окремої клітини, виникає необхідність у вирішенні й інших питань. Наприклад, для створення ефекту об’єму клітин, який необхідний для зручного розгляду стану тривимірного автомату, необхідно застосовувати освітлення. Наявність джерела світла у системі відображення забезпечує перспективу моделей клітин, дозволяючи таким чином розрізнити, на якому рівні у тривимірній структурі знаходяться клітини, що розглядаються. Також необхідний репрезентативний спосіб зобразити появу та зникнення клітини. При зміні стану до користувача необхідно чітко донести інформацію про те, як саме змінюється стан автомату. Для цього можна використати певну анімацію моделей клітин, що з’являються чи, навпаки, зникають при переході до наступної ітерації. Серед можливих анімацій варто зазначити поступову зміну прозорості від повністю невидимого до звичайного стану при появі та навпаки при зникненні клітини, а також зменшення та, відповідно, збільшення клітини з часом. Нижче можна побачити приклад анімації зміну стану клітин методом змінення їх прозорості.

Рис. 2.2 Стан автомату в процесі переходу

Однак для реалізації повноцінного засобу розгляду структури тривимірних автоматів недостатньо лише задати тривимірну модель його стану. Дуже важливо також надати засоби зміни точки огляду. Вочевидь, для детального огляду стану автомату необхідна можливість сповільнювати та пришвидшувати, а також зовсім зупиняти процес переходу до наступної ітерації. Проте незалежно від швидкості зміни ітерацій функціонал для огляду автомату з будь-якого кута є обов’язковим. Найпростіший спосіб реалізації цього функціоналу — повернення камери навколо центру клітинного автомату. Наближення та віддалення зображення зручно контролювати за допомогою коліщатка миші. Повернення ж зображення можна реалізувати за допомогою руху миші. Можливим варіантом є автоматичне повернення за кутами Ейлера при руху миші, але при цьому виникає проблема використання інших елементів інтерфейсу: для натискання певної кнопки чи інтеракції з іншими елементами інтерфейсу необхідно рухати мишею, проте це викликатиме поворот кута огляду, що ускладнить роботу з програмою. Тому більш доцільною системою є поворот у вказаному напрямку при затисканні лівої кнопки миші. Це дозволить вільно повертати зображення в будь-якому напрямку інтуїтивно зрозумілими рухами, при цьому не заважаючи користуватись інтерфейсом програми. У призупиненому стані це дозволить детально розгледіти поточну ітерацію, а під час роботи автомату дозволить побачити процес переходу до нового стану з кута, який можна динамічно обирати в процесі роботи.

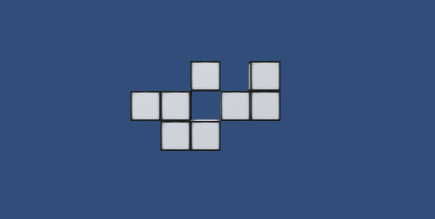
Проте навіть такі засоби можуть виявитись недостатніми для візуалізації внутрішньої структури автомату. При великій кількості заповнених клітин зовнішні клітини можуть закрити огляд внутрішніх, при цьому зміна кута огляду не допоможе. Тому важливим функціоналом, який потребує застосунок, є можливість відобразити лише один шар структури автомату. Найбільш зручним варіантом реалізації цієї функції є можливість обрати один з чотирьох форматів відображення: тривимірний та по одному формату на кожну можливу ортогональну площину відображення (вісі X/Y, X/Z та Y/Z відповідно). При відображенні в одній з ортогональних площин усі клітини стають невидимі, окрім одного шару клітин у відповідній площині. При цьому камеру необхідно розташувати перпендикулярно площині, що розглядається, таким чином формуючи псевдодвовимірне відображення автомату. Вибір конкретного шару для відображення можна реалізувати за допомогою повзунка, при цьому перехід від одного шару до іншого дозволятиметься робити динамічно, навіть під час роботи автомату. Завдяки цьому застосунок дозволить розглянути структуру автомату будь-якої складності у зручному форматі, у тому числі динамічно під час роботи автомату.

Рис.2.3 Приклад відображення окремого шару

Окрім вищезазначеного функціоналу, програма також підтримуватиме можливість відображення клітин автомату різними кольорами. Детальніше про те, яким чином клітинам буде задаватись необхідний колір, описано нижче. Подібну можливість можна використовувати декількома способами. Найтривіальнішим з них є помічення важливих клітин. У випадку використання кольорової ідентифікації клітин окремі клітини чи навіть регіони можна відобразити певним кольором. Це дозволить чітко бачити дану клітину чи регіон навіть на фоні сотень чи тисяч інших клітин. Наприклад, користувача цікавить стан конкретної клітини в процесі еволюції конфігурації. Він може позначити цю клітину окремим кольором, щоб чітко вирізняти її на фоні інших клітин. Далі користувач може розпочати еволюцію системи. Якщо розгляду клітини заважають інші клітини, доцільним є використання відображення лише шару, який містить цю клітину. Позначення ж цілого регіону одним кольором може бути використано для відстеження, чи заповнена хоч одна клітина з цього регіону, або при розгляді конкретної ділянки еволюції системи.

Іншим варіантом використання системи кольорової ідентифікації клітин є наслідування кольору при переході до наступного стану клітинного автомату. При застосування цього режиму якщо зароджується певна клітина при переході до наступної ітерації, то колір даної клітини визначається як середнє значення кольорів клітин-сусідів із випадковим зміщенням. Початкове задання кольору клітин можна провести до запуску еволюції. Після цього початкова кольорова матриця розповсюдиться автоматом, створюючи фрактальні кольорові візерунки. Це можна використовувати для формування тривимірних графічних структур, що будуть становити з себе випадковий результат еволюції початкової конфігурації. Також подібну кольорову систему можна використовувати для імітації різноманітних тривимірних процесів, наприклад, змішування різнорідних речовин. У цьому випадку кожна речовина буде позначена окремим кольором, а клітини проміжних кольорів відображатимуть пропорції та ступінь змішування речовин у даному регіоні.

**Розпізнавання переходу автомату до тривіального стану.**

**Випадок детермінованого клітинного автомату**

При дослідженні властивостей клітинного автомату одним з критеріїв оцінки може слугувати час еволюції конфігурації, тобто кількість ітерацій від початкової до переходу в тривіальний стан. Тривіальним станом зазвичай називають стан, при якому подальша еволюція системи є легко передбачуваною. Наприклад, автомат знаходиться в тривіальному стані, якщо він містить лише стабільні структури, що не змінюються з часом (“натюрморти” в термінах гри “Життя”), циклічні структури (“осцилятори”) та циклічні структури, що рухаються в певному напрямку (“космічні кораблі”). Проблема визначення тривіального стану має бути вирішена для надання функціоналу з підрахунку часу еволюції конфігурації, що дозволить детальніше дослідити особливості тривимірного клітинного автомату.

Навіть для детермінованого автомату реалізація подібного функціоналу породжує певні проблеми. Найпростішим способом перевірки тривіальності системи є перевірка, чи змінився стан хоча б однієї клітини автомату. Якщо цього не сталося, це означає, що автомат або пустий, або складається лише з натюрмортів. Проте подібний підхід не дозволяє визначити перехід до тривіального стану, що містить циклічні структури. Тому цей метод необхідно розширити, порівнюючи поточний стан з усіма попередніми станами автомату. Якщо поточний стан повністю відповідає одному з попередніх, то у випадку детермінованого автомату це означатиме зациклювання еволюції, адже з кожного стану можна перейти лише в один наступний стан. Проте певні еволюції можуть тривати тисячі ітерацій, що у випадку великого розміру автомату може викликати проблему: кожна наступна ітерація потребуватиме все більше й більше часу на обрахунок. Тому необхідний ефективний спосіб порівняння станів автомату, що дозволить обраховувати подібні довготривалі еволюції.

Доволі простим рішенням проблеми є перетворення стану автомату в послідовність бітів для подальшого побітового порівняння поточного стану з попередніми. Так як клітинний автомат, що розглядається, може приймати лише один з двох можливих станів, то стан клітини можна описати одним бітом. В цьому випадку побітове порівняння двох значень є найшвидшим способом. Проте навіть у цьому випадку особливо великі автомати з довготривалою еволюцією можуть викликати проблеми на великій швидкості обрахунку чи слабкій конфігурації комп’ютера. Тому для прискорення обрахунку подібних еволюцій пропонується формувати хеш-значення поточного стану, та порівнювати саме його. Формування хеш-значення з поточного стану також займає певний час та має виконуватись на кожній ітерації, проте це прискорить порівняння двох станів між собою, яке на пізніших ітераціях виконуватиметься тисячі разів на кожному кроці, тому час на формування хешу стану буде виправданим. Так як ймовірність, що два різних стану автомату матимуть однаковий хеш, є майже нульовою, то порівняння хешу достатньо, щоб визначити, чи зациклився клітинний автомат, перейшовши таким чином у тривіальний стан.

Якщо ж клітинний автомат перейшов у тривіальний стан з наявністю “космічних кораблів”, тобто циклічних структур, що постійно рухаються в певному напрямку, цей варіант додатково розглядати немає необхідності. Якщо в процесі руху “космічний корабель” опиниться достатньо близько від іншої стабільної структури, то еволюція може продовжитись, таким чином цей стан не є тривіальним. Якщо ж “корабель” рухатиметься без перешкод, він з часом досягне краю автомату та більше не зможе зберігати свою постійність. Таким чином, скінченність автомату виключає необхідність ускладнення обрахунку переходу до тривіального стану у випадку виникнення рухомих стабільних структур.

Прикладами тривіальних станів клітинних автоматів можуть слугувати натюрморти з гри “Життя”.

**Випадок недетермінованих (ймовірнісних) клітинних автоматів.**

Вищезазначений підхід працює для детермінованих клітинних автоматів. Принциповою особливістю таких автоматів є те, що вони з певного стану завжди переходять в один конкретний стан. Проте ймовірнісні клітинні автомати не мають такої властивості. Через наявність елементу випадковості, певні стани з певною ймовірністю можуть перейти у два чи більше різних наступних стана. Через це навіть перехід до стану, що вже траплявся в процесі еволюції автомату, не гарантує, що еволюція клітинного автомату стала тривіальною. Загалом, у подібних умовах тривіальний стан описується значно складніше, адже певні структури, що випадково змінюються у часі, можна суб’єктивно також назвати тривіальними, якщо існує лише невелика кількість станів, в які дана конфігурація може перейти з певним розподілом ймовірностей.

Розглянемо декілька можливих варіантів еволюції ймовірнісних клітинних автоматів. Найпростішим варіантом розвитку є поступове зменшення кількості заповнених клітин та “смерть” клітинного автомату, при якій всі клітини становляться пустими. Зазвичай подібний сценарій характерний для клітинних автоматів, в яких для зародження клітини потребується занадто велика кількість сусідів, як і для збереження життя вже існуючої. У цьому випадку якщо більшість клітин початкової конфігурації й має достатню для життя кількість сусідів, клітини на краю заповненої області їх не матимуть, що викличе каскадну реакцію гибелі клітин, а через низьку швидкість відродження кількість клітин буде знижуватись до нуля.

Наведемо приклад даного варіанту еволюції автомату. Нехай маємо автомат, що заданий наступними правилами: клітини зароджуються при кількості сусідів, що дорівнює випадковому значенню між 10 та 12, гине від самотності при кількості сусідів, що менше за 6-9, та від перенаселення, якщо сусідів, якщо їх більше за 12-14. У цьому випадку зображена нижче конфігурація зникне через 3 кроки. Так як для формування клітини необхідно щонайменше 10 сусідів, а клітина на пустому шарі може мати лише 9 сусідів із сусіднього шару, то на пустих шарах клітини ніколи не зароджуються. Конфігурація не може вийти за початкові межі та швидко гине.

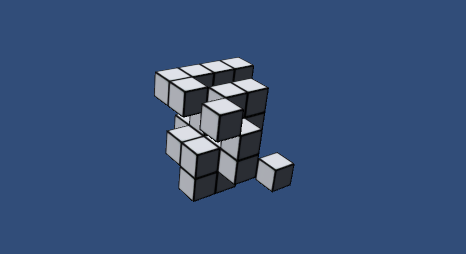


Рис. 2.4 Приклад еволюції автомату

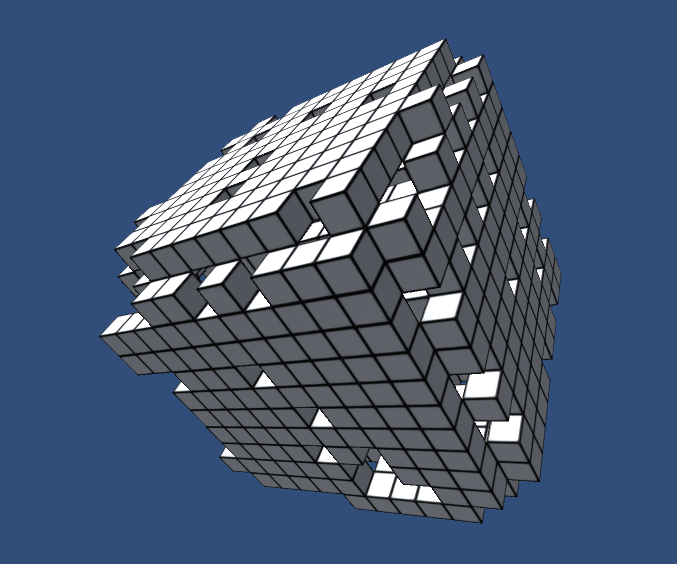
Зворотнім варіантом є заповнення усього поля. Відповідно, якщо смерть клітини наступає лише при дуже великій кількості сусідей, може трапитись ситуація, коли навіть на границях автомату клітини не гинуть. Повністю автомат заповниться лише у тому випадку, якщо максимальна кількість сусідів, при якому клітина все ще не гине від перенаселення, перевищує можливу кількість сусідів (у випадку тривимірного автомату — 26). Однак автомат все одно може бути заповнений губчатою структурою, в якій заповнені клітини перемежовуються пустими, формуючи необхідну кількість сусідів для того, щоб ні одна клітина не гинула, але й жодна пуста клітина не може заповнитись.

Рис. 2.5 Конфігурація, отримана в результаті описаного процесу еволюції:

Прикладом цього варіанту розвитку може слугувати автомат з наступною конфігурацією: зародження клітини відбувається при 6-7 сусідах, загибель від самотності — при менш ніж 4-5 сусідах, а загибель від перенаселення — при більш ніж 8-9 сусідах. Подібні правила дозволяють початковій конфігурації доволі швидко розповсюджуватись, при цьому клітини гинуть недостатньо швидко, тому структура заповнює увесь автомат. Сформована в результаті структура становить вкладені один в одного куби з заповнених клітин, що розмежовані шаром товщиною в одну пусту клітину. Подібна структура постійно змінюється всередині, тому її важко назвати тривіальним станом.

Окрім цих двох тривіальних варіантів існують й складніші типи еволюції. Певні правила клітинних автоматів здатні формувати стабільні структури в процесі еволюції. Перехід до тривіального стану може відбуватись шляхом зупинення будь-яких змін, що не входять до складу стабільних структур. Для цього не обов’язково необхідне заповнення всього автомату, а також при цьому не відбувається його остаточна смерть. Однак стабільні структури в ймовірнісних автоматах мають певні особливості. Якщо ми розглядаємо варіант ймовірнісних клітинних автоматів, при якому зародження клітини можливе в певному диапазоні кількостей сусідів, то структура може залишатись стабільною, зокрема, якщо ймовірнісний фактор ніяк не впливає на еволюцію системи. Це може відбутись, якщо всі пусті клітини, що є сусідами даної системи, мають меншу кількість сусідів, ніж необхідно для їх заповнення, а всі заповнені клітини мають кількість сусідів, яка є більшою, ніж верхня межа загибелі від самотності, та меншою, ніж нижня межа загибелі від перенаселення. В певних випадках такий діапазон кількості сусідів взагалі відсутній, і можливий варіант, що клітина одночасно має шанс загинути від самотності та перенаселення. Тоді існування стабільних структур взагалі неможливо — будь-яка структура через певний час випадково втрачає певні клітини і, як результат, стабільність.

Найбільший інтерес представляють правила, за яких автомат здатен до активного розповсюдження, що не призводить до швидкого переходу у тривіальний стан. Зазвичай це автомати 3 та 4 класу. Саме такі клітинні автомати можуть викликати проблеми при підрахунку часу еволюції конфігурації. Описана вище проблема порівняння поточної ітерації з усіма попередніми стає ще більш суттєвою, адже навіть одне повторення не гарантує зациклення, а відсутність повторення ще не гарантує відсутність зациклення. Наприклад, якщо в процесі еволюції конфігурації з’явилась стабільна структура з випадково зароджуючимися клітинами, не впливаючими на стабільність структури, то вона може завадити коректному детектуванню зациклення і, відповідно, переходу в тривіальний стан.

На основі розглянутих сценаріїв можна зробити висновок, що для ймовірнісних автоматів недоцільно використовувати той самий підхід до визначення часу еволюції, що й для детермінованих автоматів. Більшість структур, що є стабільними протягом нескінченно великих проміжків часу, взагалі не змінюються з часом. Формування стабільних структур з клітинами, що випадково змінюють свій стан, не порушуючи при цьому стабільність структури, є менш ймовірним. Виникнення ж зациклених структур у випадку ймовірнісного клітинного автомату є ще менш ймовірним. Так як подібний функціонал не є основною метою розробки програми, адже він лише слугує для зручного визначення часу еволюції системи, а ймовірність виникнення ситуацій, в яких з’являються складніші стабільні структури, є досить незначною, програма підтримуватиме лише визначення переходу до тривіального стану, що виражений повною відсутністю змін протягом значного часу. Надання складнішої системи визначення тривіального стану суттєво знизить продуктивність програми, що не є доцільним для обробки незначного проценту ситуацій другорядними підсистемами.

**Задання стану клітин автомату.**

**Безпосереднє задання стану окремих клітин.**

Можливість задання стну клітин є обо’язковим функціоналом програми. Він дозволить досліджувати окремі, заздалегідь задані конфігурації на предмет їх стабільності та відповідності певним критеріям (наприклад, великий час еволюції). Однак процес задання стану клітин тривимірного автомату ускладнюється тим, що користувач не може бачити усієї структури відразу та не здатен рухати покажчик у трьох вимірах одночасно. Перша проблема вирішується засобами відображення тривимірних автоматів, що були описані в одному з попередніх підрозділів. Друга проблема ж є розвитком цих ідей з додаванням елементів інтерактивності.

Головні проблеми, які потрібно вирішити при реалізації даного функціоналу, це спосіб відображення поточної позиції покажчику та спосіб зручного переміщення покажчика у тривимірному просторі. При відображенні поточної позиції покажчику необхідно розглянути два варіанти: коли покажчик знаходиться на заповненій клітині, та коли він знаходиться на пустій клітині. Доволі репрезентативним способом зображення покажчика є періодичне миготіння відповідної клітини. Якщо клітина пуста, на ній періодично з’являтиметься напівпрозорий куб, чия прозорість поступово мінятиметься від нуля до певної величини, відображаючи таким чином поточну обрану клітину. Якщо ж клітина заповнена, відбуватиметься зворотній процес: періодично куб в даній клітині ставатиме напівпрозорим, поступово змінюючи прозорість від повністю непрозорого до певної величини та назад.

Друга проблема ж вирішується шляхом використання двовимірної репрезентації одного з шарів автомату. Так як програма містить функціонал з відображення певного шару автомату, до нього можна додати можливість обирати конкретну клітину курсором миші чи стрілками клавіатури. Змінювати ж шар, на якому необхідно встановити стан клітини, можна за допомогою повзунка чи коліщатка миші. Подібна система дозволить обрати будь-яку клітину та змінити її стан зручним чином. Доцільним також є надання пожливості затиснути ліву кнопку миші та, проводячи по конкретному шару, заповнювати всі клітини, по яким проведено курсором. Це дозволить заповнювати великі масиви клітин швидше, ніж при індивідуальному їх заданні.

Окрім цього, необхідно надати можливість задання кольору клітин. Для цього можна використати спеціальний режим задання кольору. В цьому режимі замість зміни стану клітини відбуватиметься вибір даної клітини задля задання її кольору. Після вибору клітини спеціальна панель дозволить обрати колір цієї клітини на палітрі чи безпосередньо повзунками. Одночасно може бути обрана лише одна клітина.

**Початкове задання конфігурації автомату. Завантаження та запис.**

Окрім можливості задати значення окремих клітин, необхідним є також функціонал з задання початкової конфігурації автомату. Так як при заданні початкової конфігурації може виникнути потреба в нетривіальному заповненні великої кількості клітин, то необхідне рішення, що дозволить зменшити необхідність подібного підходу к мінімуму.

Першим способом вирішення цієї проблеми є випадкове задання початкової конфігурації. В багатьох випадках для перевірки властивостей певного набору правил тривимірного автомату найбільш доцільним є саме використання випадкового початкового стану, адже воно дозволить швидко розглянути еволюцію автомату на різних початкових умовах, що не є виродженими випадками. Доцільним є заповнення лише певної частини автомату замість усього поля, адже таким чином можна побачити, наскільки розширюється займана площа у процесі еволюції. Якщо ж все поле буде заповнено випадковими станами клітин, простору для розширення еволюція не матиме, що може нівелювати цінність експерименту. Також можливо надати користувачеві функціонал з визначення щільності заповнення початкової ділянки поля, що можна визначати через ймовірність заповнення окремої клітини.

Окрім задання випадкового початкового стану, необхідно також надати функціонал зі збереження та завантаження певних конфігурацій. Якщо користувач досліджує не загальні властивості певного автомату, а властивості конкретних конфігурацій, в нього може виникнути необхідність багатократно задавати одну й ту саму конфігурацію. Збереження дозволить задати складну конфігурацію лише один раз, використовуючи випадкове задання чи безпосереднє заповнення окремих клітин, а далі зберегти конфігурацію для подальшого користування. Також збереження поточного стану може бути корисним для перезапуску еволюції, що дозволить повернути автомат в початковий стан до початку еволюції.

Завантаження поточного стану заповнить необхідні клітини згідно збереженої конфігурації, відновлюючи стан автомату. Інформацію про стан автомату можна зберігати шляхом запам’ятовування правил поточного автомату, його розмірності та бітового представлення станів клітин, подібного до того, що пропонувалось використовувати для підрахунку переходу до тривіального стану. Крім того, доцільним є збереження кольорової ідентифікації клітин автомату. Задання необхідної кольорової схеми автомату з метою зручного відображення клітинного автомату чи використання наслідування кольору може зайняти значну кількість часу, тому можливість зберегти кольорову конфігурацію може зекономити цей час.

Зберігати таку інформацію можна у базі даних, що дозволить зручно отримувати конфігурації за різними критеріями, такими як назва, час створення, розмірність та правила автомату. Іншим варіантом є збереження стану методом серіалізації. Подібний підхід дозволить створювати зручні в розповсюдженні файли поточної конфігурації, що міститимуть усі необхідні дані. Такі файли можна копіювати на зовнішній носій чи передати мережею Інтернет для розгляду та використання іншими користувачами. Також подібний підхід не потребує використання бази даних, що спростить встановлення застосунку — достатньо лише завантажити *.exe* файл та запустити його. Крім того, використання локальних файлів дозволить використовувати застосунок без підключення до мережі Інтернет.

Завантаження стану потребуватиме коректної обробки даних з серіалізованого файлу. Цього можна досягти шляхом перевизначення масивів, що зберігають тривимірні моделі клітин, з подальшим заданням їх стану та кольору. Також необхідною є можливість переглянути доступні збережені файли станів. У випадку даної програми відображується список доступних у спеціальному каталогу файлів за іменами, вибір одного з яких дозволить завантажити відповідну конфігурацію.

Таким чином застосунок надає повноцінний інструментарій для роботи з тривимірними клітинними автоматами: обрахунок їх еволюції, зручну систему відображення стану автомату, систему кольорової диференціації клітин, спрощену систему визначення переходу до тривіального стану, інструментарій по заданню стану та кольору клітини, а також можливість запису та завантаження стану клітинного автомату разом із правилами.