# 1. Вступ

Клітинні автомати є потужним інструментом моделювання та дослідження, що наразі використовуються в науці, наприклад у моделюванні автомобільного трафіку, планів евакуації та навіть розповсюдження сигналів у мозоку людини, а в 2020 році набули популярність дослідження розповсюдження вірусів, також змодельовані за допомогою клітинних автоматів. Але  перед тим як використовувати їх в якості інструмента, непогано було б ознайомитися з принципами їх роботи та потренуватися створювати більш прості моделі, перш ніж приступати до чогось складнішого.

Зараз майже не існує систем, які б дозволяли це зробити людині, що не має якихось додаткових знань чи навичок: потрібно або вчити власну мову якоїсь з обчислювальних систем, або вчитися програмувати та створювати її власноруч.

Отже, ми вирішили розробити систему моделювання клітинних автоматів, яка б мала зручний графічний інтерфейс та було зрозуміла у використанні будь-кому, хто захотів би познайомитися з клітинними автоматами в дослідницьких, навчальних або розважальних цілях.

# 2. Визначення клітинних автоматів

Клітинний автомат зазвичай визначають як динамічну систему, в якій простір, час та стани системи є дискретними та мають наступні властивості:

* Простір представлений сіткою в кінцевій кількості вимірів (зазвичай — в одному чи двох)
* Кожна клітина в клітинному автоматі може перебувати в одному стані з їх кінцевої множини
* Система клітинного автомату розвивається протягом кількох часових етапів. Стан усіх клітин оновлюється синхронно на кожному етапі
* Стан клітин оновлюється відповідно до набору правил, які ставлять у відповідність поточному стану клітини та її сусідів новий стан цієї клітини.

# 3. Класичні КА

Напевно найвідомішим прикладом клітинного автомату є “Гра життя” Конвея, що була створена як проста екологічна модель, яка містить клітини, що можуть мати лише два стани: бути живою або мертвою, та які живуть та вмирають відповідно до кількох простих правил:

* Жива клітина, що має менше ніж два живих сусіда, вмирає через «малонаселеність»
* Жива клітина, що має два або три живих сусіда, продовжує жити у «наступному поколінні»
* Жива клітина, що має більше ніж три живих сусіда, вмирає через «перенаселеність»
* Мертва клітина, що має рівно три живих сусіда, стає живою «у наслідку розмноження»

На рисунку можна побачити поведінку одної зі структур, що створюються протягом еволюцїї цього автомату, а саме глайдер.

# 4. Блокові КА

Ще одним окремим класом клітинних автоматів є блокові, правила конструювання яких суттєво відрізняються від розглянутих раніше класичних, але які є дуже корисним для моделювання фізичних властивостей. Блоковим є клітинний автомат, у якому:

1. Масив клітин розбитий на множину кінцевих, окремих однорідних частин – блоків.
2. Блоки не перетинаються та обмін інформацією між блоками відсутній.
3. Є правила для блоків, які розглядають та оновлюють вміст усього блоку (а не окремої клітини як у звичайному КА). Одне й те саме правило застосовується до всіх блоків.
4. Розбиття змінюється від кроку до кроку таким чином, щоб відбувалося перекриття блоків, що використовуються на сусідніх кроках. Це правило є дуже суттєвим, адже при використанні одного й того самого розбиття на всіх кроках автомат був би розбитий на сукупність незалежних підсистем, та не зміг функціонувати як одне ціле.

Найчастіше використовуваним розбиттям є сусідство Марголуса, в якому сітка розбита на чотирьох-клітинні блоки, які здвигаються на одну клітину вздовж кожного виміру після кожного кроку часу. Це виглядає наступним чином: правила перетворення застосовуються спочатку на блоки, обмежені синіми лініями, на наступному кроці – на блоки, обмежені червоними лініями, і так далі.

Таблиця правил переходу у наступний стан для блокового автомата містить вихідний стан для кожного з можливих варіантів вхідного складається усього з 16 елементів.

# 5. Порівняння систем моделювання КА

Перед початком розробки було оглянуто такі існуючі системи, що надають можливість моделювати клітинні автомати, як

* програма Wolfram Mathematica,
* машина клітинних автоматів CAM-6
* Веб-сайт CelLab.

Перші дві системи не є доступними широкому колу користувачів через їх платність, та жодна з трьох не надає можливості працювати навіть з найпростішими клітинними автоматами без наявності додаткових знань, а саме мов програмування або обчислення. У цей час наша система мала зрозумілість та зручність однією з головних вимог, чим вигідно відрізняється від розглянутих аналогів.

# 6. Структурна схема системи

Для реалізації серверної сторони проєкту потрібна була об’єктно-орієнтована мова програмування для забезпечення зручної роботи з різними видами правил, яка гарно працює з багатопотоковістю для забезпечення належної швидкості обчислення станів автоматів. Остаточним вибором стала мова С#, що відповідає усім цим вимогам.

Для клієнтської частини одразу була обрана мова JavaScript, адже вона є об'єктивно найпопулярнішою у веб-розробці, а точніше - її фреймворк Angular, що підтримує гнучку систему вкладених компонентів, яка виявилася дуже зручною при розробці графічного інтерфейсу задання правил.

Отже система працює наступним чином: на сторінці редагування правил (RulesEditorComponent) користувач може створювати власні, або змінювати існуючі набори правил. Далі він переходить на сторінку симуляціїї (SimulationComponent), обирає один з них, задає початковий стан, та розпочинає симуляцію. Тоді клієнт надсилає на сервер інформацію про початковий стан та обране правило у спрощеному числовому вигляді. Сервер  перетворює моделі правил у більш складну систему класів, після чого запускає симуляцію, під час якої на кожному зі станів автомату кожна клітина (або їх блок) обробляються паралельно, що дозволяє дуже швидко обчислювати автомати навіть з дуже великою розмірністю сітки. Після того як сервер обчислив деяку задану кількість станів, він відправляє їх на клієнт, який відображає їх на сітці. Процес обміну даних між клієнтом та сервером продовжується, доки користувач не натисне кнопку Stop.

# 7. Компоненти правил

У системі є можливість створювати клітинні автомати на основі 4 типів правил:

* пряме - треба поставити у відповідність точному  стану сусідства клітини її наступний стан
* тоталістичне - наступний стан клітини залежить від ск\уми обраних клітин її сусідства
* складене - можна об’єднати правила за допомогою логічних операторів І/АБО/НЕ/ВиключнеАБО
* блокове - для створення блокових клітинних автоматів. задаються початковий та наступний стан сусідства Марголуса.

# 8. Сторінка редагування правил

* список існуючих наборів правил
* поточний набор правил, його назва та правила з яких він складається
* кольорова схема набору. за допомогою схеми користувач може змінювати кількість станів, що будуть використовуватися в автоматі, їх числове відображення та кольори, якими вони будуть відображатися на екрані

Зображений набор правил для гри життя:

* тоталістичне правило “якщо сума сусідів дорівнює трьом то клітина стає живою”
* складене правило якщо клітина є живою ТА сума її сусідів дорівнює двом, вона залишається живою

# 9. Приклад вигляду набору правил для блокового КА

Зображене правило НРР (рух гратчастого газу по діагоналях)

# 10. Сторінка симуляції

* Сітка, по який користувач може малювати
* Вибір набору правил
* Вибір стану (кольору) для малювання
* Кнопки старт, стоп
* кнопка очищення сітки
* кнопка випадкового заповнення сітки

Тут обране правило Гра життя та виконано випадкове заповнення. Після запуску можна побачити наступне:

# 11. Результат симуляції гри життя

# 12. Модифіковане правило HPP

На одному з попередніх слайдів було показано правило поведінки гратчастого газу для блокового автомату (частинки газу рухаються по діагоналям, при зіткненні розлітаючись у протилежних напрямках). Це правило є предустановленим, але його можна модифікувати, щоб наприклад симулювати поведінку газу в вакуумі. Для цього я додала ще один стан, який описує клітини пляшки (зелений) та додала 4 правила, що описують правила взаємодії газу з нею (вони відскакують та рухаються у протилежному напрямку)