Т2. №3(35)’2024 https://doi.org/10.31474/2786-9024/v2i3(35).319553

Математичне та комп'ютерне моделювання

ВИКОРИСТАННЯ AGENTSCRIPT ДЛЯ СТВОРЕННЯ БАГАТОРІВНЕВИХ МОДЕЛЕЙ АГЕНТНОГО МОДЕЛЮВАННЯ

Є.О. Єжова[1], Н.О. Маслова[1]

[1] Applied Mathematics and Informatics Department, Donetsk National Technical University, Drohobych, Ukraine

E-mail: yelyzaveta.yezhova@donntu.edu.ua

Отримано 24.12.2024
Прийнято до публікації 28.12.2024
Опубліковано 31.12.2024

АНОТАЦІЯ

Актуальність дослідження зумовлена необхідністю пошуку доступних та ефективних інструментів для моделювання складних систем, які були б придатними як для наукових досліджень, так і для навчальних цілей. Існуючі платформи, хоча й надають значні можливості, часто є складними для освоєння, що створює бар’єри для їх широкого застосування. Метою статті є дослідження можливостей платформи AgentScript для створення агентних моделей у веб-середовищі, а також аналіз її переваг та обмежень у порівнянні з традиційними інструментами. Практична значимість роботи полягає у висвітленні способів застосування AgentScript для швидкого прототипування, розробки моделей середньої складності та інтерактивного відображення. Інструмент дозволяє реалізувати моделювання без необхідності встановлення спеціалізованого програмного забезпечення, що сприяє його популяризації серед студентів та дослідників. Наукова значимість роботи полягає у визначенні перспектив використання AgentScript для моделювання багаторівневих систем. Особливу увагу приділено аналізу архітектури платформи, заснованої на шаблоні Model-View-Controller, що забезпечує ефективну організацію моделювання. Наведено огляд функціональних можливостей AgentScript, описано процес створення агентів, середовища та правил поведінки, продемонстровано приклади реалізації моделей для дослідження поширення хвороб, симуляції лісових пожеж і взаємодії в екосистемах, виконано порівняння платформи з іншими інструментами, що підкреслює її переваги, такі як доступність та простота інтеграції з сучасними веб-технологіями. Отримані результати показують, що AgentScript є ефективним інструментом для досліджень і навчання у сфері агентного моделювання, який здатний задовольнити потреби користувачів із різним рівнем підготовки.

***Ключові слова**: мультиагентні системи, моделювання, прототипування, AgentScript, динамічні системи, веб-технології, симуляція, візуалізація даних.*

Т2. №3(35)’2024 https://doi.org/10.31474/2786-9024/v2i3(35).319553

ВСТУП

У сучасній науці агентне моделювання (agent-based model, ABM) стало одним із провідних методів аналізу складних систем завдяки своїй здатності описувати взаємодії між окремими компонентами (агентами) та їх колективною поведінкою. Підхід дозволяє моделювати взаємодію індивідуальних компонентів у таких сферах, як біологія [1], екологія [2], соціальні науки [3], економіка, епідеміологія, транспорт, а також у розв’язанні задач оптимізації. ABM охоплює не лише взаємодії між агентами, але й їх адаптацію до змін у середовищі та їх здатність до навчання, що дозволяє точно відтворювати складні соціальні або природні явища [4]. У таких системах часто виникають непередбачувані наслідки, що робить ABM надзвичайно корисним інструментом для дослідження систем з численними взаємозалежними компонентами.

Завдяки агентному підходу стає можливим досліджувати динамічні процеси, що виникають у багаторівневих системах, та отримувати нові інсайти щодо їх поведінки. Наприклад, в економіці агентні моделі дозволяють вивчати еволюцію ринкових тенденцій, в епідеміології – поширення хвороб серед населення, а в екології – вплив різних факторів на розвиток екосистем. Крім того, цей підхід надає можливість створювати експериментальні середовища, в яких можна перевіряти гіпотези, змінювати параметри систем та спостерігати за наслідками в контрольованих умовах. Це особливо важливо для передбачення поведінки складних систем, де традиційні методи аналізу можуть бути недостатньо ефективними.

АНАЛІЗ ЛІТЕРАТУРНИХ ДЖЕРЕЛ ТА ПОСТАНОВКА ПРОБЛЕМИ

У останні роки проведено значну кількість досліджень, присвячених аналізу інструментів для ABM. У [5] представлено ґрунтовний порівняльний огляд інструментів ABM, який є одним із найдетальніших у цій галузі. Автори дослідження оцінюють понад 80 інструментів, аналізуючи їх за такими критеріями, як доступність та тип ліцензії, вихідний код, агентна поведінка, мова програмування/API, інтегроване середовище розробки (IDE), компілятор, операційна система, платформа реалізації, масштабованість, моделювальна стійкість та галузі застосування. Основна мета цієї роботи полягає у тому, щоб допомогти вченим та інженерам обрати відповідний інструмент ABM для своїх дослідницьких завдань.

Стаття [6] описує понад 40 інструментів ABM, використовуючи інформацію з відкритих джерел та особистий досвід авторів. Огляд фокусується на застосуванні ABM у різних сферах, таких як біологія, медицина, фізика, хімія, соціальне та економічне моделювання, кібербезпека, екологія, транспорт та оптимізація мереж постачання. Крім того, дослідження приділяє увагу ефективності виконання ABM на високопродуктивних обчислювальних системах.

У статті [7] виконано порівняння чотирьох вільно доступних бібліотек для комп’ютерного моделювання на основі агентів соціальних наук: RePast, Swarm, Quicksilver та VSEit. Дослідники дійшли висновку, що програмне забезпечення RePast демонструє найкращі результати у порівнянні з іншими бібліотеками.

Стаття [8] пропонує порівняльний аналіз 24 існуючих агентних платформ, розробляючи критерії для їх оцінки за властивостями, зручністю використання, продуктивністю, прагматичністю та безпекою. У роботі також представлено класифікацію платформ, яка допомагає користувачам ідентифікувати схожі властивості та робити вибір залежно від поставлених завдань.

У статті [9] досліджено мови програмування та інструменти розробки для багатоагентних систем, зокрема декларативні, імперативні та гібридні мови, а також інтегровані середовища розробки. Автори також розглядають фреймворки, які не прив’язані до конкретних мов програмування, що розширює можливості розробників.

У роботі [10] описано структуру програмного пакета, призначеного на вирішення завдань розподілу ресурсів. Особливістю пакета є застосування режиму «інтелектуальний агент» для пошуку ефективного рішення за часом. Розглядаються класифікація інтелектуальних агентів та особливості їх застосування у вирішенні завдань розподілу ресурсів. Тож це є достатньо ранньою демонстрацією застосування агентних технологій в рішенні класичних задач.

Постановка проблеми. Попри значні успіхи у розвитку інструментів ABM, багато з них залишаються складними для освоєння та використання. Популярні платформи, такі як NetLogo [11] та Repast [12], хоча й пропонують розширені можливості для моделювання, вимагають від користувачів глибоких технічних знань або спеціального програмного забезпечення. Це обмежує доступність цих інструментів для широкого кола дослідників, викладачів і студентів, які могли б застосовувати їх у своїй роботі. Таким чином, виникає потреба у пошуку більш доступних і простих у використанні платформ, які не поступаються згаданим у функціональності.

ISSN 2786-9024

Т2. №3(35)’2024 https://doi.org/10.31474/2786-9024/v2i3(35).319553

Мета статті: Оцінити функціональні можливості платформи AgentScript для створення агентних моделей у веб-середовищі та визначити її переваги й недоліки у порівнянні з іншими інструментами моделювання.

Для досягнення мети необхідно реалізувати наступні задачі:

1. Виконати порівняння AgentScript з іншими інструментами, підкресливши її доступність та простоту інтеграції з сучасними веб-технологіями.
2. Висвітлити способи застосування AgentScript для швидкого прототипування, розробки моделей середньої складності та інтерактивного навчання.
3. Надати приклади реалізації моделей.
4. Оцінити перспективи використання AgentScript для моделювання багаторівневих систем.

МЕТОДИ ДОСЛІДЖЕНЬ

У дослідженні застосовано методи структурного та порівняльного аналізу, проведено огляд наукової і методичної літератури, а також онлайн-ресурсів із використанням інформаційного та аналітичного підходів. Для перевірки функціональності програмного коду використано експериментальні методи.

РЕЗУЛЬТАТИ ДОСЛІДЖЕННЯ

AgentScript [13] є прикладом інноваційної системи, яка пропонує мінімалістичний підхід до ABM. Платформа заснована на концепціях NetLogo, але значно спрощує процес створення моделей завдяки інтеграції з веб-середовищем. Її головною перевагою є орієнтація на використання у браузері, що усуває необхідність встановлення складного програмного забезпечення і робить платформу доступною для широкого кола користувачів. Крім того, AgentScript пропонує спрощений набір інструментів для створення агентних моделей, що робить його ідеальним вибором для швидкого прототипування, інтерактивного навчання та реалізації невеликих або середніх за складністю симуляцій. Завдяки використанню сучасних веб-технологій, таких як HTML5 і JavaScript, платформа забезпечує можливість створення візуалізацій, які можуть бути легко адаптовані до конкретних завдань. В Табл. 1 наведено результати порівняння платформ агентного моделювання AgentScript, NetLogo та Repast, які орієнтовані на симуляцію агентних систем, але мають свої особливості й орієнтацію на виконання різних типів завдань.

Табл.1. Характеристики платформ

Характеристика	AgentScript	NetLogo	Repast
Мова програмування	JavaScript	NetLogo (спеціалізована мова)	Java, Groovy, Python, C#
Цільова аудиторія	Освіта, дослідження, веб-розробка	Освіта, моделювання для початківців	Дослідники, розробники складних моделей
Легкість використання	Легка, орієнтована на веб	Дуже легка, інтуїтивно зрозуміла	Складна, вимагає програмування
Сфера застосування	Просте моделювання, інтеграція з веб-інтерфейсами	Моделювання в соціальних, біологічних та економічних системах	Складні багаторівневі моделі, великі симуляції
Можливість кастомізації	Висока, завдяки JavaScript	Обмежена, залежить від мови	Дуже висока, з підтримкою кількох мов
Продуктивність	Обмежена, підходить для невеликих моделей	Помірна, для середніх моделей	Висока, підтримує масштабовані моделі
Візуалізація	Проста, орієнтована на веб	Інтуїтивна, інтегрована	Складна, залежить від інструментів Java
Платформа	Браузери, Node.js	Windows, macOS, Linux	Windows, macOS, Linux
Підтримка бібліотек	Обмежена, через JavaScript-екосистему	Вбудовані інструменти та розширення	Велика кількість бібліотек для розробки
Спільнота та документація	Менша, але зростаюча	Широка, багато навчальних матеріалів	Дуже широка, орієнтована на дослідників

 https://doi.org/10.31474/2786-9024/v2i3(35).319553

AgentScript вирізняється простотою використання, високою кастомізацією завдяки JavaScript, та інтеграцією з веб-технологіями, що робить його ідеальним для швидкого прототипування, навчання й створення простих моделей. Платформа забезпечує кросплатформність, працює в браузерах і дозволяє використовувати сучасні веб-інтерфейси без потреби у спеціалізованому програмному забезпеченні.

AgentScript побудований на JavaScript та використовує архітектуру Model-View-Controller (Рис. 1), що забезпечує ефективну організацію моделювання та зручність роботи навіть для користувачів із базовими знаннями у програмуванні.

Model-View-Controller (MVC, "Модель-Вид-Контролер") – це архітектурний шаблон, який розділяє дані додатка і керуючу логіку на три окремі компоненти: модель, уявлення та контролер. Це поділ дозволяє кожному компоненту виконувати свою специфічну функцію, що робить додаток більш модульним і полегшує подальші зміни, тестування і підтримку [15].

Model визначає агентів та їхню поведінку. View відповідає за візуалізацію моделі, даючи користувачеві можливість спостерігати за перебігом симуляції в режимі реального часу. Controller управляє взаємодією між моделлю і поданням, обробляє введення користувача і підтримує роботу моделі.

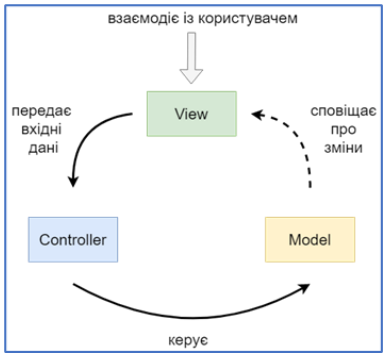


Рис. 1. Діаграма взаємодії між компонентами шаблону MVC [14]

Особливостями термінології шаблону MVC є застосування понять черепаха (окремі мобільні агенти в симуляції, кожна черепаха може мати атрибути); патч (стаціонарні агенти, які складають мережу або середовище, зберігають значення атрибутів); посилання (зв’язки між агентами).

Агенти в AgentScript можуть мати різні атрибути, які визначають їхній стан і поведінку. Ці атрибути визначають, як агенти діють і реагують у симуляції (Табл. 2).

Табл.2. Приклади можливих атрибутів агентів у AgentScript

Атрибут	Опис
Позиція	Розташування агента в симуляції, часто представлене координатами (x, y) у середовищі сітки
Колір	Зовнішній вигляд агента, який може змінюватися з часом залежно від його стану чи дій
Енергія	Загальний атрибут, який може зменшуватися з часом або через такі дії, як рух, і, можливо, потребуватиме поповнення через певну поведінку
Швидкість	Швидкість, з якою рухається агент, або частота його дій
Здоров'я	Стан, який може визначати життєздатність агента
Напрямок	Орієнтація або кут агента, важливий для руху та прийняття рішень

Агенти в AgentScript дотримуються попередньо визначених правил, які визначають їхню поведінку. Ці правила можуть бути простими або складними, і вони керують діями агента на основі його поточного стану або середовища навколо нього.

До можливих варіантів поведінки належать: рух, взаємодія або прийняття рішень. При цьому «черепахи» можуть пересуватися сіткою або середовищем, агенти можуть взаємодіяти з іншими агентами, приймати рішення на основі обставин, брати участь у соціальній поведінці.

Для 3D візуалізації застосовується Three.js, що є бібліотекою JavaScript з кросбраузерністю та інтерфейсом прикладного програмування, що використовується для створення та відображення анімованої 3D-комп'ютерної графіки у веббраузері [16].

Зазвичай модель розгортається через веб-сервер. AgentScript розгортається через сторінки GitHub (що перетворює репозиторій на сервер) та шляхом публікації в NPM (Node Package Manager).

Також можливо використовувати спеціальний тип веб-сервера, який не потребує встановлення на робочому столі, а працює повністю у браузері, серед них CodePen [17].

Написання моделей на AgentScript. В AgentScript моделі створюються шляхом визначення агентів, їх середовища, а також правил і поведінки, які регулюють їх взаємодію.

ISSN 2786-9024

Т2. №3(35)’2024 https://doi.org/10.31474/2786-9024/v2i3(35).319553

Створення агентів. Агенти – активні об’єкти моделювання, які виконують дії та взаємодіють із середовищем або іншими агентами. Створення агенту містить наступні кроки:

– *визначення властивостей агента* – кожен агент створюється з певними властивостями, такими як положення, колір, енергія та напрямок;

– *визначення початкових станів* – установлення початкових значень для цих властивостей, щоб визначити початкові умови;

– *визначення поведінки* – визначення дій, які може виконувати агент, як-от переміщення, взаємодія чи зміна свого стану.

Визначення середовища. Середовище в AgentScript складається з патчів (або сітки), де агенти можуть рухатися та взаємодіяти.

Встановлення правил і поведінки. Правила – це умови та дії, які визначають, як агенти поводяться та взаємодіють у симуляції.

Події та взаємодія. Події в AgentScript ініціюють взаємодію між агентами або між агентами та їх середовищем.

Візуалізація з переглядом в AgentScript. Компонент View в AgentScript має вирішальне значення для візуалізації симуляції шляхом зображення агентів і середовища. Зазвичай для графіки використовується `HTML <canvas>` [18], що дозволяє налаштовувати елементи відображення, такі як кольори, форми та динамічні зміни у реальному часі.

Елемент `HTML <canvas>` служить основною областю для 2D візуалізації. Налаштування полотна передбачає визначення його розміру та доступ до його двовимірного контексту візуалізації.

Щоб представити агентів візуально, можна намалювати фігури (наприклад, кола або квадрати) на місці кожного агента на полотні. Агенти можуть мати такі атрибути, як колір, розмір і форма, щоб розрізняти їх або вказувати на їхній стан.

Середовище, представлене сіткою ділянок, можна візуалізувати як сітку кольорових клітинок на полотні. Кожна клітинка представляє ділянку з атрибутами, такими як рівень ресурсів, тип місцевості або інші особливості, з якими можуть взаємодіяти агенти.

ПРАКТИЧНЕ ЗАСТОСУВАННЯ AGENTSCRIPT

Моделювання поширення хвороби за допомогою агентного підходу. Моделювання поширення хвороб є важливою складовою вивчення епідеміологічних процесів та розробки ефективних стратегій боротьби з інфекціями. Одним із перспективних методів для аналізу таких процесів є ABM, яке дозволяє детально відтворити взаємодію між індивідуальними агентами (особами, які можуть бути носіями хвороби) та їх колективною поведінкою в рамках великої популяції.

ABM поширення хвороби засноване на використанні моделей, де кожен агент має певний набір характеристик, таких як вік, здоров’я, місце перебування та рівень інфікованості. Агент знаходиться в одному з кількох можливих станів: здоровий (зелений), інфікований (червоний) або такий, що одужав (синій). Випадкові 40% агентів починають інфікованими (відсоток задається користувачем).

Агентне середовище, яке визначає взаємодії агентів, представлене у вигляді сітки, де кожен агент займає певну клітинку і може переміщатися в межах цього середовища (Рис. 2). Агенти рухаються в випадкових напрямках. Вони відбиваються від краю полотна. Якщо агент інфікований, він заражає здорових агентів у межах радіуса зараження, який задається користувачем. Інфіковані агенти переходять у стан "одужалий" після визначеного часу (recoveryTime) та починають мати імунітет до захворювання .

```
// Поширення хвороби
if (agent.state === "infected") {
    agents.forEach(otherAgent => {
        if (otherAgent.state === "healthy") {
            const distance = Math.sqrt(
                (agent.x - otherAgent.x) ** 2 +
                (agent.y - otherAgent.y) ** 2
            );
            if (distance < infectionRadius) {
                otherAgent.state = "infected";
            }
        }
```

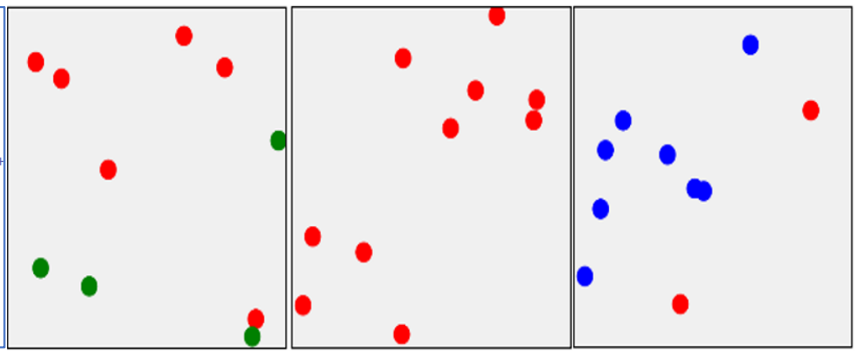


Рис. 2. Фрагмент програмного коду та симуляція поширення хвороби

Параметри агентів наступні:

 https://doi.org/10.31474/2786-9024/v2i3(35).319553

– Кількість агентів, що використано в даній симуляції 200 (можлива довільна кількість агентів, задається користувачем.

– *Стан агентів*: Здоровий (зелений) – 60% - агент не заражений. Інфікований (червоний) – 40% (задається користувачем на початку симуляції) - агент заразний протягом 100 кроків. Вилікуваний/одужалий (синій) – виникає після одужання, агент більше не може заразитися.

– *Радіус зараження*: 2 клітинки.

– *Тривалість зараження*: 100 кроків, після чого агент стає таким, що одужав.

– *Швидкість руху*: 1 крок за один такт.

– *Початкові умови*: Розподіл агентів по сітці випадковий. Початковий стан кожного агента визначається з ймовірностями 60% (здоровий) і 40% (інфікований).

– *Зміна параметрів під час симуляції*: Інфікований агент заражає здорових агентів у межах свого радіуса зараження. Інфіковані агенти змінюють стан на "одужалий" після закінчення 100 тактів. Агенти рухаються випадково, відбиваючись від країв середовища.

Далі проведено три експерименти. При першому експерименті змінюється густота населення, яка визначається кількістю агентів на одиницю площі. Для цього послідовно встановлюються кілька рівнів густоти: низький 50 агентів, середній 100 агентів і високий 150 агентів. На кожному рівні густоти агенти випадковим чином розподіляються по сітці. Агенти можуть перебувати в різних станах: здорові, інфіковані або такі, що одужали.

Відсоток інфікованих та радіус зараження залишається тим же, що в початковій симуляції 40% та 2 клітинки відповідно.

На представленому графіку (Рис. 3) відображено динаміку кількості агентів у різних станах (здорові, інфіковані та ті, що одужали) залежно від часу (вимірюється в умовних кроках моделювання).

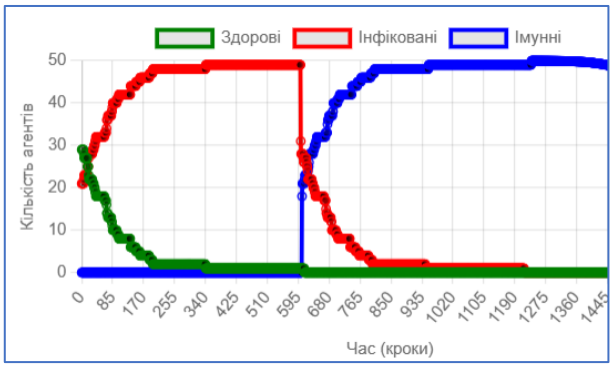
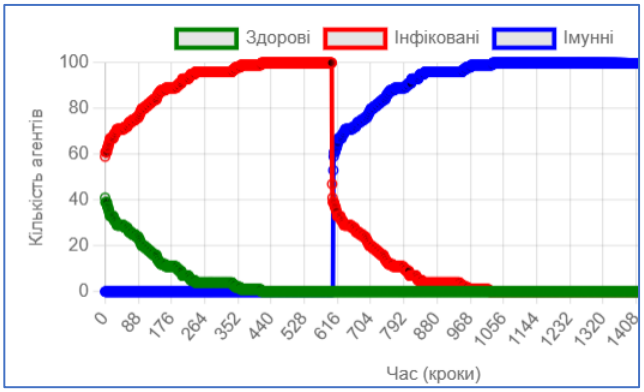
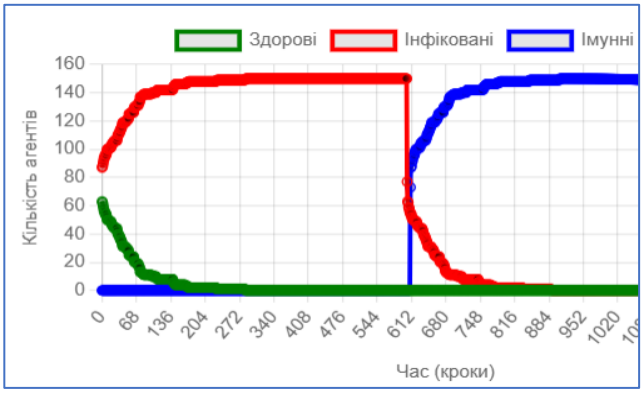


Рис. 3. Графік симуляції поширення хвороби при густоті 50 агентів/на одиницю площі

На початку симуляції 30 здорових агентів. Із часом їхня кількість поступово зменшується, оскільки інфекція поширюється серед популяції. До моменту приблизно 510 кроків всі здорові агенти заражаються, і їх кількість стає нульовою. Інфіковані агенти з'являються на перших кроках симуляції у кількості 20. Їх кількість швидко зростає, досягаючи піку на 425-510 кроках. Після цього кількість інфікованих різко зменшується, і на 680 кроках всі агенти або одужують, або отримують імунітет. На початку кількість агентів, що одужали дорівнює нулю. Після 510 кроків, коли частина агентів починає одужувати, кількість агентів, які одужали починає швидко зростати. До 680 кроків всі агенти стають такими, що одужали, і ця кількість залишається стабільною до кінця симуляції (Рис. 4).



а)



б)

Рис. 4. Графік симуляції поширення хвороби при густоті: а) 100 агентів, б) 150 агентів

Після завершення епідемії вся популяція стає здоровою, що свідчить про розвиток колективного імунітету.

Отже, за низької густоти інфекція поширюється повільніше, оскільки агенти розташовані на значній відстані один від одного, що зменшує кількість контактів між здоровими та інфікованими. Пік захворюваності настає пізніше та характеризується меншою кількістю

ISSN 2786-9024

 https://doi.org/10.31474/2786-9024/v2i3(35).319553

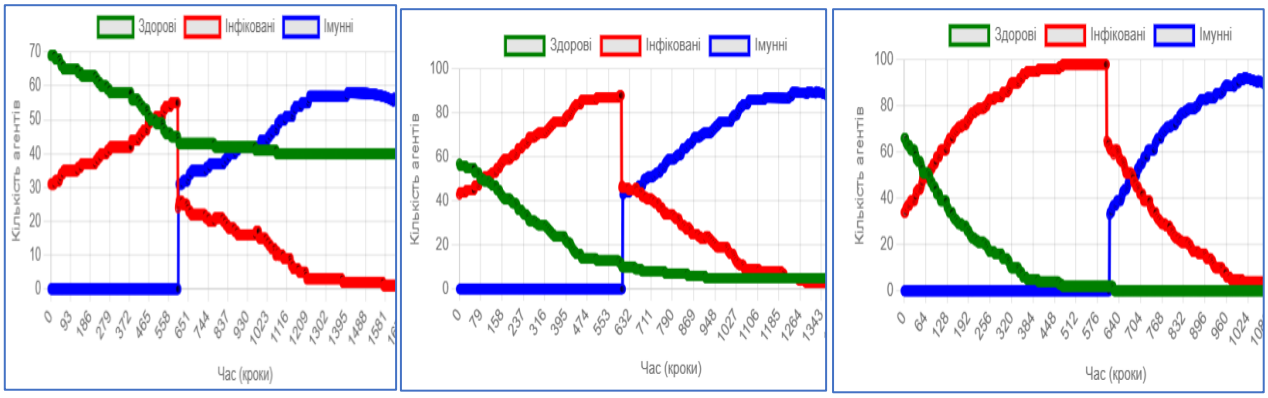
одночасно інфікованих агентів порівняно з іншими рівнями густоти. Крива інфікованих має пологий підйом і спад, що вказує на довшу тривалість епідемії.

При середній густоті інфекція поширюється швидше, оскільки ймовірність контактів між агентами зростає. Пік захворюваності настає раніше, а кількість одночасно інфікованих агентів досягає значно більшого значення порівняно з низькою густотою. Тривалість епідемії скорочується через швидше виснаження популяції здорових агентів.

За високої густоти інфекція поширюється найбільш інтенсивно, оскільки велика кількість агентів знаходиться у безпосередній близькості. Пік захворюваності настає дуже швидко і характеризується максимально можливою кількістю інфікованих агентів у популяції. Епідемія проходить стрімко, із найменшою тривалістю, оскільки всі здорові агенти заражаються за короткий проміжок часу.

Вища густота призводить до більшого піку захворюваності, що свідчить про критичний вплив частоти контактів між агентами на швидкість поширення інфекції. У популяціях із низькою густотою поширення інфекції обмежується через меншу кількість контактів, що сприяє розтягуванню епідемії у часі. Отримані результати підтверджують, що густота населення є ключовим фактором, який впливає на динаміку поширення інфекції.

При другому експерименті варіюється радіус зараження, що визначає максимальну відстань, на якій інфікований агент може передати інфекцію здоровому агенту. Для кожного рівня густоти населення радіус зараження поступово змінюється, починаючи з мінімального, який відповідає безпосередньому контакту між агентами, і закінчуючи значеннями, що охоплюють більшу кількість сусідніх агентів (Рис. 5, а – в).



а) **б)** **в)**

Рис. 5. Графік симуляції поширення хвороби при різних радіусах:
а) радіус 2, б) радіус 5, в) радіус 10

Отже, зі збільшенням радіуса зараження швидкість поширення інфекції зростає, а тривалість епідемії зменшується. Малий радіус зараження обмежує поширення інфекції, дозволяючи епідемії тривати довше, але з меншою інтенсивністю. Великий радіус зараження забезпечує швидке охоплення всієї популяції, створюючи різкий і інтенсивний пік інфікованості. При будь-якому радіусі зараження спостерігається стабільне завершення епідемії, коли всі агенти переходять у стан імунітету, але час досягнення цього стану та кількість пікових інфікованих змінюються залежно від радіуса.

При третьому експерименті варіюється час, протягом якого агент залишається інфікованим, до переходу у стан здорового. Встановлюються різні періоди інфекції, які імітують поведінку вірусів із різною тривалістю інфекційного періоду. Зі збільшенням тривалості інфекційного періоду інтенсивність епідемії та швидкість її поширення зростають. Короткий інфекційний період сприяє розтягуванню епідемії у часі, але зменшує інтенсивність піку захворюваності. Довгий інфекційний період забезпечує швидке поширення інфекції, але призводить до більш високих піків інфікованості. Загальна тривалість епідемії скорочується із зростанням тривалості інфекційного періоду, оскільки більша кількість агентів заражається у найкоротші терміни.

Серед ще можливих варіантів розширення даної системи є розширення індивідуальних характеристик агентів та їх взаємодії, що відбуваються в популяції. Крім того, АВМ дає змогу врахувати різноманітні стратегії контролю хвороби, наприклад, введення карантину, вакцинацію або лікування, що дозволяє аналізувати ефективність різних заходів у реальному часі.

Імітація поширення лісової пожежі за допомогою АВМ. АВМ в даному контексті дозволяє створювати детальну модель взаємодії між різними компонентами лісової екосистеми, такими як рослинність, метеорологічні умови

 https://doi.org/10.31474/2786-9024/v2i3(35).319553

та вогонь, з урахуванням просторових і часових аспектів поширення пожежі. Симуляція показує дерева як агентів зі станами здорового, горючого або обгорілого. З імовірністю вогонь може перекинутися на сусідні дерева. Також на пожежу впливають зовнішні фактори, такі як напрямок і швидкість вітру. Вітер збільшує ймовірність перекидання вогню на сусідні дерева. Кожне дерево може поширювати вогонь на сусідні дерева (вгору, вниз, ліворуч, праворуч) з певною ймовірністю залежно від швидкості вітру. Якщо дерево горить деякий час, воно згоряє.

Кожне дерево горить протягом фіксованого проміжку часу (як задається користувачу), перш ніж перетворитися на попіл (Рис. 6).

```
// Оновлення стану лісу (поширення пожежі)
function updateForest() {
    forest.forEach(tree => {
        if (tree.state === 'burning') {
            tree.burningTime++;
            if (tree.burningTime > 11.6) {
                tree.state = 'burnt';  /// Дерево деякий час обгорає
            }
            // Поширити вогонь на сусідні дерева з певною ймовірністю
            spreadFire(tree);
        }  });
```

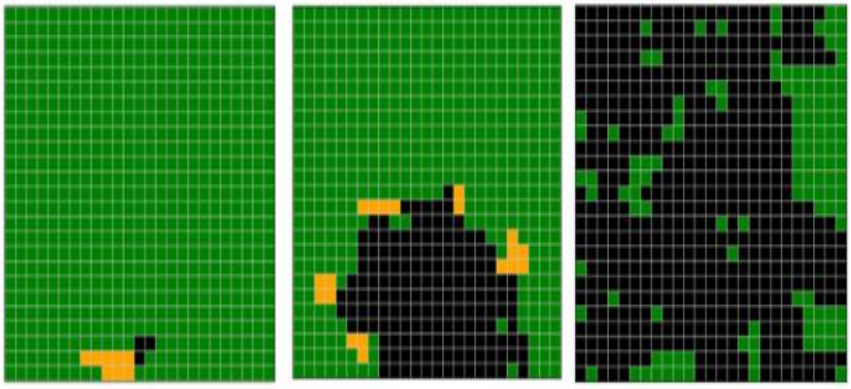


Рис. 6. Фрагмент програмного коду та симуляція лісової пожежі

Параметри агентів наступні:

– *Стан дерев*: Здорове (зелений) – 80% - дерево не горить. Горюче (помаранчевий) – активується при контакті з вогнем, дерево активно горить. Згоріле (чорний) – виникає після завершення фази горіння, дерево, що більше не бере участі в поширенні вогню.

– *Тривалість горіння одного дерева*: 50 кроків.

– *Ймовірність поширення вогню*: Базова – 0.3. Зі збільшенням швидкості вітру – до 0.7.

– *Початкові умови*: Задано одну осередкову точку загоряння (випадкова клітинка в сітці). Всі інші дерева знаходяться в стані "здорове".

– *Зміна параметрів під час симуляції*: Коли дерево підпалюється, його стан змінюється на "горюче". Після 50 кроків дерево стає "згорілим". Горюче дерево з ймовірністю 30–70% поширює вогонь на сусідні дерева (верх, низ, ліворуч, праворуч).

АВМ дозволяє не тільки прогнозувати розвиток пожежі в реальному часі, а й проводити аналіз ефективності різних стратегій боротьби з вогнем, таких як створення протипожежних ліній, контроль за рівнем вологи в ґрунті чи застосування спеціальних технологій для гасіння вогню. За допомогою симуляцій можна оцінити ймовірні наслідки різних сценаріїв, визначити найбільш вразливі ділянки лісу та спрогнозувати зони ризику, що є важливим для планування дій в умовах надзвичайних ситуацій.

Імітація взаємодії в екосистемі через АВМ. АВМ дозволяє створити математичні моделі, які відтворюють поведінку окремих агентів (організмів, рослин, тварин) і взаємодії між ними в межах екосистеми, що дає можливість більш глибоко аналізувати динаміку природних процесів.

Одним із класичних прикладів такої взаємодії є модель хижаків, наприклад, вовків, і їх здобичі, наприклад, кроликів. В даному контексті АВМ дозволяє дослідити, як зміни в популяціях цих видів можуть впливати на їх чисельність та загальний стан екосистеми. Крім того, агенти взаємодіють між собою через механізми полювання (вовки намагаються зловити кроликів) та здобичі (кролики розмножуються, щоб підтримати свою популяцію).

Вовки, як хижаки, полюють на кроликів, щоб забезпечити своє виживання та відтворення. Вовки перевіряють відстань до кроликів. Якщо вовк наближається до кролика, він "з'їдає" його, отримуючи енергію кролика. З іншого боку, кролики розмножуються, що створює потік нових особин, необхідних для збереження популяції. Якщо рівень енергії кролика перевищує певний поріг, він може створити новий "нащадок" поруч із собою, витрачаючи частину своєї енергії.

Кролики – агенти, що рухаються по екосистемі, споживають їжу для відновлення енергії та можуть розмножуватися. Втрачають енергію за кожен крок. Якщо енергія кролика досягає нуля, він стає "мертвим" і перестає рухатися.

Вовки – хижаки, які полюють на кроликів для підтримання енергії. Вони втрачають енергію під час руху, а при досягненні нульової енергії стають "мертвими". Їжа

ISSN 2786-9024

 https://doi.org/10.31474/2786-9024/v2i3(35).319553

– розподілена по всьому полю. Кролики можуть споживати їжу для відновлення енергії (Рис.7).

Параметри агентів наступні:

– *Види агентів*: Кролики (здобич) – 100 особин.

Атрибути кроликів:

– *Початкова енергія*: 50.

– Розмноження відбувається, якщо енергія ≥ 100 (втрачається 50 після створення нового кролика).

– *Втрата енергії за хід*: 1.

– *Відновлення за їжу* – +30.

Вовки (хижаки) – 20 особин.

Атрибути вовків:

– *Початкова енергія:* 70.

– *Енергія у разі, якщо кролика спіймано:* +50.

– *Втрата енергії за хід*: 2.

– *Початкові умови*: Кролики та вовки розташовані випадковим чином у сітці. Розташування їжі: Випадкове. Початковий рівень енергії кожного агента фіксований.

– *Зміна параметрів під час симуляції*: Кролики рухаються випадково, шукають їжу та відновлюють енергію, якщо знаходять її. Якщо енергія кролика ≥ 100, він створює нового кролика, втрачаючи 50 енергії. Якщо енергія кролика ≤ 0, він помирає. Вовки полюють на кроликів, наближаючись до них і з’їдаючи їх. Якщо вовк їсть кролика, його енергія збільшується на 50. Якщо енергія вовка ≤ 0, він помирає.

```
// Репродукція кроликів
function reproduceRabbits() {
    let newRabbits = [];
    rabbits.forEach(rabbit => {
        if (rabbit.energy >= 15) {  // Якщо у кролика
достатньо енергії, він розмножується
            rabbit.energy -= 13;  // Енергія витрачається на
розмноження
            newRabbits.push({
                x: rabbit.x + Math.random() * 20 - 10, //
Новий кролик поруч
                y: rabbit.y + Math.random() * 20 - 10,
                energy: rabbitEnergy
            }); } });
    rabbits = rabbits.concat(newRabbits);  // Додаємо нових
кроликів в екосистему
```

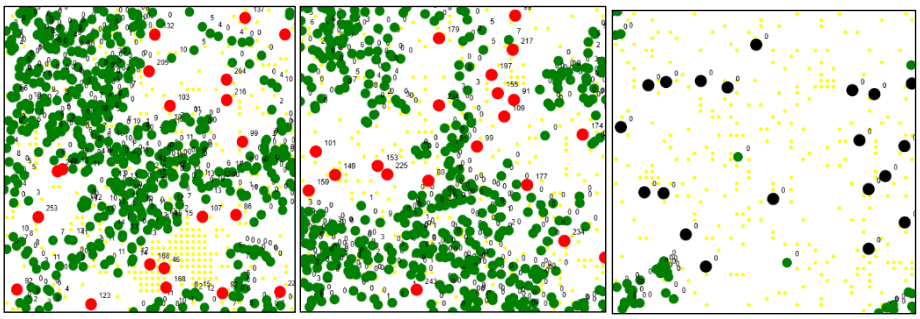


Рис. 7. Фрагмент програмного коду та симуляція взаємодії в екосистемі

АВМ цього процесу дозволяє не лише спостерігати за динамікою популяцій у реальному часі, а й проводити експерименти з різними параметрами, такими як зміна рівня хижацтва або можливість кроликів до адаптації та уникання хижаків. Вивчення цих процесів допомагає глибше зрозуміти механізми регулювання популяцій та їх взаємодії, а також може бути застосоване для оптимізації стратегій збереження біорізноманіття та управління природними ресурсами.

Усі моделі використовують `HTML <canvas>` для візуалізації, а також інтерактивні налаштування для регулювання параметрів під час роботи симуляції.

Далі в Табл. 3-6 наведено взаємодію компонентів платформи AgentScript за MVC.

Табл. 3. Компонент Model (Модель) для кожної симуляції

Параметри	1 симуляція	2 симуляція	3 симуляція
Агенти	Здоровий, інфікований, одужалий. Радіус зараження: 2 клітинки. Час інфікування: 100 кроків.	Стан дерева: здорове (зелений), в огні (помаранчевий), згоріле (чорний). Час горіння: 50 кроків. Ймовірність поширення вогню: базова – 30%, зі вітром – 70%.	Кролики: мають енергію, розмножуються за енергії ≥ 100. Вовки: полюють на кроликів, отримуючи енергію. Їжа: генерується випадково на сітці.
Середовище	Розмір сітки: 100×100 клітинок.	Розмір сітки: 50×50 клітинок.	Розмір сітки: 100×100 клітинок.
Граничні умови	Агенти відбиваються від країв сітки.	Вогонь за межі сітки не поширюється.	Рух агентів обмежений межами сітки.

 https://doi.org/10.31474/2786-9024/v2i3(35).319553

Правила	Інфікований агент заражає здорових агентів у радіусі. Зміна стану агента: інфікований – одужалий.	Горюче дерево поширює вогонь на сусідні дерева з певною ймовірністю. Після 50 кроків дерево стає згорілим.	Кролики шукають їжу, втрачають енергію та розмножуються. Вовки полюють на кроликів, втрачаючи енергію, якщо не знаходять їжу.

Табл. 4. Компонент View (Подання) для кожної симуляції

Параметри	1 симуляція	2 симуляція	3 симуляція
Візуалізація станів	Відображення агентів різного стану: зелений (здоровий), червоний (інфікований), синій (такий, що одужав).	Відображення дерева різних станів у реальному часі.	Відображення агентів: кролики, вовки та їжа.
Динамічне оновлення	Динамічне оновлення положення агентів і станів у реальному часі.	Інтерактивне налаштування швидкості симуляції.	Графічне представлення змін станів (наприклад, зникнення кроликів або вовків).
Інструмент візуалізації	Використання HTML <canvas> для 2D-візуалізації.	Використання HTML <canvas> для 2D-візуалізації.	Використання HTML <canvas> для 2D-візуалізації.

Табл. 5. Компонент Controller (Контролер) для кожної симуляції

Параметри	1 симуляція	2 симуляція	3 симуляція
Налаштування параметрів	Обробка налаштувань, таких як радіус зараження та тривалість інфікування.	Обробка вводу користувача для зміни швидкості симуляції або початкового стану дерев.	Управління параметрами, такими як кількість їжі, кроликів і вовків.
Моделювання взаємодій	Контроль руху агентів і їх взаємодії.	Передача даних до Model для розрахунку поширення вогню.	Передача даних про взаємодії (полювання, розмноження) Model.
Оновлення стану	Оновлення станів агентів після кожного кроку симуляції.	Оновлення View для відображення стану дерев.	Оновлення View після кожного кроку.

Табл. 6. Схема взаємодії для кожної симуляції

Параметри	1 симуляція	2 симуляція	3 симуляція
Налаштування параметрів	Користувач налаштовує радіус зараження та кількість інфікованих.	Користувач визначає початкові умови (точка займання).	Користувач задає початкові параметри (кількість кроликів і вовків).
Передача даних до Model	Controller передає ці дані до Model.	Controller передає ці дані Model.	Controller передає ці дані до Model.
Розрахунок у Model	Model розраховує зміну стану агентів.	Model розраховує поширення вогню.	Model розраховує зміну енергії та стану агентів.
Оновлення View	Controller передає оновлені дані до View.	Controller оновлює View.	Controller оновлює View.
Візуалізація стану	View відображає оновлені стани агентів.	View показує динаміку горіння.	View візуалізує динаміку популяцій.

Як було зазначено вище, поділ на компоненти компонент може бути незалежно змінений або дозволяє спростити процес розробки, оскільки кожен вдосконалений без впливу на інші частини системи.

ISSN 2786-9024

 https://doi.org/10.31474/2786-9024/v2i3(35).319553

ОБГОВОРЕННЯ РЕЗУЛЬТАТІВ

Результати дослідження підтверджують, що AgentScript є ефективним інструментом для створення агентних моделей у веб-середовищі. Виявлено, що платформа демонструє низку переваг, які роблять її конкурентоспроможною у порівнянні з популярними інструментами, такими як NetLogo та Repast. Зокрема, основними перевагами AgentScript є простота використання, доступність через браузер та легкість інтеграції з сучасними веб-технологіями.

Однією з ключових особливостей платформи є її архітектура MVC, яка сприяє організації коду та забезпечує модульність моделей. Це дозволяє користувачам чітко розділяти логіку, дані та візуалізацію, що підвищує ефективність розробки та подальшого налагодження моделей. Використання HTML5 для візуалізації дає можливість створювати інтерактивні графічні зображення у реальному часі, що є важливим для навчальних і демонстраційних цілей. Платформа також може бути використана у сферах, не розглянутих у статті, таких як освітні експерименти, де студенти можуть моделювати різноманітні сценарії, або при розробці адаптивних систем, що змінюють свою поведінку залежно від умов середовища.

Проведені експерименти підтвердили, що AgentScript добре підходить для моделювання систем середньої складності (в прикладах - поширення хвороб, лісові пожежі та взаємодія в екосистемах). Ці моделі дозволили оцінити динаміку систем у різних сценаріях і продемонстрували здатність платформи гнучко налаштовувати параметри моделювання.

Водночас у ході дослідження виявлено певні обмеження AgentScript. Платформа менш придатна для роботи з великими обсягами даних або надзвичайно складними моделями через обмеження продуктивності, характерні для браузерного середовища. Крім того, відносно невелика кількість готових бібліотек та прикладів ускладнює початок роботи для новачків.

ВИСНОВКИ

У статті досліджено можливості використання платформи AgentScript для ABM складних систем у веб-середовищі. На основі проведеного аналізу виявлено, що AgentScript є ефективним інструментом для швидкого прототипування, навчання та реалізації моделей середнього рівня складності для моделювання завдяки простоті використання, доступності через браузер і інтеграції з сучасними веб-технологіями.

Основними перевагами платформи є:

модульна архітектура Model-View-Controller, яка спрощує розробку та підтримку моделей;

підтримка HTML5 і JavaScript, що забезпечує інтерактивну візуалізацію даних у реальному часі;

доступність для користувачів із базовими технічними знаннями завдяки мінімальним вимогам до встановлення.

Практичні приклади моделювання, представлені у статті, включають симуляцію трьох моделей. Вони демонструють гнучкість і адаптивність AgentScript для вирішення різноманітних задач.

Водночас виявлено певні обмеження платформи, такі як зниження продуктивності при роботі з великими наборами даних та відсутність розвиненої бібліотеки готових моделей. Це визначає її оптимальну сферу застосування – навчальні проекти, експериментальні дослідження та розробку середньо складних моделей.

Перспективи розвитку AgentScript включають інтеграцію з хмарними обчисленнями для підвищення масштабованості та продуктивності, а також використання інтерфейсів на основі штучного інтелекту для автоматизації аналізу й оптимізації моделей. Це відкриває нові можливості для застосування платформи в галузях, що потребують обробки великих даних і динамічного моделювання.

Таким чином, AgentScript можна розглядати як перспективне рішення для агентного моделювання у веб-середовищі, яке поєднує простоту, доступність і функціональність. Подальші наробки можуть бути спрямовані на розширення можливостей платформи, інтеграцію із зовнішніми джерелами даних та покращення її продуктивності для роботи з моделями більшої складності.

ЛІТЕРАТУРА

[1] Bora S., Emek S. Agent-Based modeling and simulation of biological systems. Modeling and computer simulation. 2019. URL: https://doi.org/10.5772/intechopen.80070. Дата звернення: 08.12.2024.

[2] Grimm V., Railsback S. F. Agent-Based models in ecology: patterns and alternative theories of adaptive behaviour. Agent-Based computational modelling. Heidelberg. P. 139–152. URL: https://doi.org/10.1007/3-7908-1721-x_7. Дата звернення: 08.12.2024.

[3] Klein, D., Marx, J., Fischbach, K. Agent-based modeling in social science, history, and philosophy. An introduction. 2018. Vol. 43, No. 1 (163), Special Issue: Agent-Based Modeling in Social Science, History, and Philosophy (2018), P. 7-27. URL: https://doi.org/10.12759/hsr.43.2018.1.7-27. Дата звернення: 08.12.2024.

[4] Challenges, tasks, and opportunities in modeling agent-based complex systems / L. An et al. Ecological modelling. 2021. Vol. 457. P. 109685. URL:

https://doi.org/10.1016/j.ecolmodel.2021.109685. Дата звернення: 08.12.2024.
[5] Agent Based Modelling and Simulation tools: a review of the state-of-art software / S. Abar et al. Computer science review. 2017. Vol. 24. P. 13–33. URL: https://doi.org/10.1016/j.cosrev.2017.03.001. Дата звернення: 08.12.2024.
[6] Allan, R.J.: Survey of Agent Based Modelling and Simulation Tools. 2010. In: Technical Report DL-TR-2010-007, Science and Technology Facilities Council. Дата звернення: 08.12.2024.
[7] Tobias, R., Hofmann, C.: Evaluation of free Java-libraries for social-scientific agent based simulation. In: Journal of Artificial Societies and Social Simulation, 2004. Vol. 7. Дата звернення: 08.12.2024.
[8] Kravari K., Bassiliades N. A survey of agent platforms. Journal of artificial societies and social simulation. 2015. Vol. 18, no. 1. URL: https://doi.org/10.18564/jasss.2661. Дата звернення: 08.12.2024.
[9] Allan, R. J. (2010). Survey of agent based modelling and simulation tools (pp. 1362-0207). New York: Science & Technology Facilities Council. Дата звернення: 08.12.2024.
[10] Маслова Н. А. Використання інтелектуальних агентів при вирішенні завдань розподілу ресурсів / Н. А. Маслова, О. В. Мовчан // Штучний інтелект. – 2014 . – №3. – С. 80-89. – URL: http://nbuv.gov.ua/UJRN/II_ 2014 _3_11.
[11] P. Pimenta, "Application of model-driven engineering to multi-agent systems : a language to model behaviors of reactive agents", Diss. Université Montpellier, 2017. Дата звернення: 08.12.2024.
[12] Collier N. T., Ozik J., Tatara E. R. Experiences in Developing a Distributed Agent-based Modeling Toolkit with Python. 2020 IEEE/ACM 9th Workshop on Python for High-Performance and Scientific Computing (PyHPC), GA, USA, 13 November 2020. 2020. URL: https://doi.org/10.1109/pyhpc 51966.2020.00006. Дата звернення: 08.12.2024.
[13] AgentScript. [Онлайн]. URL: http://agentscript.org/. Дата звернення: 08.12.2024.
[14] Модель-вид-контролер. [Онлайн]. URL: https://uk.wikipedia.org/wiki/Модель-вид-контролер. Дата звернення: 08.12.2024.
[15] Code smells for Model-View-Controller architectures / M. Aniche et al. Empirical Software Engineering. 2017. Vol. 23, no. 4. P. 2121–2157. URL: https://doi.org/10.1007/s10664-017-9540-2. Дата звернення: 08.12.2024.
[16] Three.js. [Онлайн]. URL: https://en.wikipedia.org/wiki/Three.js. Дата звернення: 08.12.2024.
[17] Codepen. [Онлайн]. URL: https://codepen.io/your-work. Дата звернення: 08.12.2024.
[18] Casabona J. HTML and CSS: visual quickstart guide. Pearson Education, Limited, 2021. Дата звернення: 08.12.2024.

USING AGENTSCRIPT TO PRODUCE MULTI-LEVEL AGENT-BASED MODELLING MODELS

Yelyzaveta Yezhova, Nataliia Maslova

The relevance of the study is driven by the need to find affordable and effective tools for modelling complex systems that would be suitable for both research and educational purposes. Existing platforms, although providing significant opportunities, are often difficult to learn, which creates barriers to their widespread use. The purpose of the article is to study the capabilities of the AgentScript platform for creating agent-based models in the web environment, as well as to analyse its advantages and limitations in comparison with traditional tools. The practical significance of the work lies in highlighting the ways in which AgentScript can be used for rapid prototyping, development of models of medium complexity and interactive display. The tool allows to implement modelling without the need to install specialised software, which contributes to its popularisation among students and researchers. The scientific significance of the work lies in determining the prospects of using AgentScript for modelling multilevel systems. Particular attention is paid to the analysis of the platform architecture based on the Model-View-Controller template, which ensures efficient modelling. The article provides an overview of AgentScript functionality, describes the process of creating agents, environments and behavioural rules, demonstrates examples of implementing models for studying the spread of diseases, simulating forest fires and interactions in ecosystems, and compares the platform with other tools, which emphasises its advantages, such as accessibility and ease of integration with modern web technologies. The results show that AgentScript is an effective tool for research and education in the field of agent-based modelling, which is able to meet the needs of users with different levels of training.

***Keywords**: multi-agent systems, modelling, prototyping, AgentScript, dynamic systems, web technologies, simulation, data visualisation.*

REFERENCES

[1] Bora S., Emek S. Agent-Based modeling and simulation of biological systems. Modeling and computer simulation. 2019. URL: https://doi.org/10.5772/intechopen.80070. Accessed: 08.12.2024.
[2] Grimm V., Railsback S. F. Agent-Based models in ecology: patterns and alternative theories of adaptive behaviour. Agent-Based computational modelling. Heidelberg. P. 139–152. URL: https://doi.org/10.1007/3-7908-1721-x_7. Accessed: 08.12.2024.
[3] Klein, D., Marx, J., Fischbach, K. Agent-based modeling in social science, history, and philosophy. An introduction. 2018. Vol. 43, No. 1 (163), Special Issue: Agent-Based Modeling in Social Science, History, and Philosophy (2018), P. 7-27. URL: https://doi.org/10.12759/hsr.43.2018.1.7-27. Accessed: 08.12.2024.
[4] Challenges, tasks, and opportunities in modeling agent-based complex systems / L. An et al. Ecological modelling. 2021. Vol. 457. P. 109685. URL:

ISSN 2786-9024

https://doi.org/10.1016/j.ecolmodel.2021.109685. Accessed: 08.12.2024.
[5] Agent Based Modelling and Simulation tools: a review of the state-of-art software / S. Abar et al. Computer science review. 2017. Vol. 24. P. 13–33. URL: https://doi.org/10.1016/j.cosrev.2017.03.001. Accessed: 08.12.2024.
[6] Allan, R.J.: Survey of Agent Based Modelling and Simulation Tools. 2010. In: Technical Report DL-TR-2010-007, Science and Technology Facilities Council. Accessed: 08.12.2024.
[7] Tobias, R., Hofmann, C.: Evaluation of free Java-libraries for social-scientific agent based simulation. In: Journal of Artificial Societies and Social Simulation, 2004. Vol. 7. Accessed: 08.12.2024.
[8] Kravari K., Bassiliades N. A survey of agent platforms. Journal of artificial societies and social simulation. 2015. Vol. 18, no. 1. URL: https://doi.org/10.18564/jasss.2661. Accessed: 08.12.2024.
[9] Allan, R. J. (2010). Survey of agent based modelling and simulation tools (pp. 1362-0207). New York: Science & Technology Facilities Council. Accessed: 08.12.2024.
[10] Maslova N. A., Movchan O. V. Use of intelligent agents in solving resource allocation problems // Artificial Intelligence. – 2014. – No. 3. – P. 80–89. – URL: http://nbuv.gov.ua/UJRN/II_%202014%20_3_11. Accessed: 08.12.2024.
[11] P. Pimenta, “Application of model-driven engineering to multi-agent systems: a language to model behaviors of reactive agents”, Diss. Université Montpellier, 2017. Accessed: 08.12.2024.
[12] Collier N. T., Ozik J., Tatara E. R. Experiences in Developing a Distributed Agent-based Modeling Toolkit with Python. 2020 IEEE/ACM 9th Workshop on Python for High-Performance and Scientific Computing (PyHPC), GA, USA, 13 November 2020. 2020. URL: https://doi.org/10.1109/pyhpc 51966.2020.00006. Accessed: 08.12.2024.
[13] AgentScript. [Online]. URL: http://agentscript.org/. Accessed: 08.12.2024.
[14] Model-view-controller. [Online]. URL: https://uk.wikipedia.org/wiki/ Model-view-controller. Accessed: 08.12.2024.
[15] Code smells for Model-View-Controller architectures / M. Aniche et al. Empirical Software Engineering. 2017. Vol. 23, no. 4. P. 2121–2157. URL: https://doi.org/10.1007/s10664-017-9540-2. Accessed: 08.12.2024.
[16] Three.js. [Online]. URL: https://en.wikipedia.org/wiki/ Three.js. Accessed: 08.12.2024.
[17] Codepen. [Online]. URL: https://codepen.io/your-work. Accessed: 08.12.2024.
[18] Casabona J. HTML and CSS: visual quickstart guide. Pearson Education, Limited, 2021. Accessed: 08.12.2024.