11. Suciu A., Lebu D., Marton K. (2011). Unpredictable Random Number Generator Based on Mobile Sensors. IEEE International Conference on Intelligent Computer Communication and Processing. doi: 10.1109/ICCP.2011.6047913

12. Environment sensors. Android Developers. URL: https://developer.android.com/develop/sensors-and-location/sensors/sensors_environment.

Надійшла до редколегії 26.05.2025 р.

Остапець Денис Олександрович, кандидат технічних наук, доцент, доцент кафедри ЕОМ УДУНТ, м. Дніпро, Україна, e-mail: odaua@i.ua, ORCID: https://orcid.org/0000-0003-1778-7770 (науковий керівник здобувача вищої освіти Опрятного А.О.)

Опрятний Артур Олександрович, здобувач вищої освіти кафедри ЕОМ УДУНТ, м. Дніпро, Україна, e-mail: artur.opriatnyi@icloud.com, ORCID: https://orcid.org/0000-0001-7145-9677

УДК 519.21:004.021 DOI: 10.30837/0135-1710.2025.185.070

В.В. КРАСНІКОВ, П.Е. СИТНІКОВА

АЛГОРИТМ ЗАВОЮВАННЯ ДЛЯ СТОХАСТИЧНОГО ЗАПОВНЕННЯ ДВОВИМІРНИХ ДИСКРЕТНИХ РЕШІТОК ЗВ'ЯЗАНИМИ ОБЛАСТЯМИ

Запропоновано алгоритм завоювання, який дає змогу швидко заповнювати двовимірні поля неперетинними зв'язними областями заданих площ. Це робить можливим ефективну процедурну генерацію наборів даних. Роботу алгоритму продемонстровано на прикладі створення ігрового поля для модифікованої версії класичної задачі N-Queens. У підсумку набір даних генерується повністю процедурно, а ймовірність отримати дві ідентичні генерації прямує до нуля. Це відкриває можливість використовувати запропонований алгоритм у складніших системах, зокрема, для створення синтетичних зображень, ігрових рівнів або тестових вибірок.

Доведено ефективність розробленого алгоритму.

1. Вступ

При вирішенні багатьох сучасних прикладних задачах виникає потреба у процедурному заповненні двовимірних полів (матриць, решіток) зв'язними блоками різного розміру. Як очікувані результати таких задач можуть розглядатися тестові набори даних, процедурно згенеровані ігрові рівні, синтетичні набори для аугментації моделей машинного навчання, біомедичні мікроматриці тощо. Заповнення полів – це структурування простору, що створює універсальний інструмент для різних типів задач. В геоінформатиці й картографії це може бути генерація зон для карт – полігонів, кластерів, типів місцевості, при обробці зображень – сегментація зображень на зв'язні області. Задача має практичне або теоретичне застосування для багатьох галузей.

Для задоволення зазначених потреб зазвичай використовують алгоритми процедурної генерації та стохастичного моделювання, що поєднують різні підходи до отримання максимально різноманітних випадкових розподілів. Ці алгоритми відрізняються способом представлення даних і такими характеристиками, як трудомісткість та час виконання. Попри схожий напрям, кожен алгоритм формує власний тип результату, а вибір конкретного алгоритму залежить від вимог користувача й обмежень задачі.

Проблема вибору алгоритму процедурної генерації поля із заданими характеристиками є нетривіальною, оскільки жоден із наявних підходів не забезпечує гарантованої відповідності до цільових параметрів, таких як кількість зв'язаних блоків, їхні форми та просторове розташування. Це, у свою чергу, обумовлює потребу у додаткових дослідженнях, спрямованих на аналіз обмежень існуючих алгоритмів, розробку критеріїв порівняння та створення нових методів генерації, які б забезпечували

контрольоване структурування простору.

Актуальність роботи підтверджується наявністю цілого класу задач, які потребують адаптивного структурування двовимірного простору з дотриманням складних умов зв'язності та сегментації, що на сьогодні не можуть бути повноцінно розв'язані з використанням жодного із класичних алгоритмів.

2. Аналіз існуючих алгоритмів і постановка проблеми

Розглянемо модифіковану версію класичної задачі N-Queens (далі N-Queens). У класичному формулюванні мета задачі – розмістити 8 ферзів на звичайній шахівниці так, щоб у кожному рядку, стовпці й діагоналі стояв рівно один ферзь.

У модифікованому варіанті задачі N-Queens кількість ферзів дорівнює N на полі розміром N x N. Саме поле розфарбовано у N кольорів. Треба розставити ферзів таким чином, щоб у кожному кольорі опинився рівно один ферзь. Дозволяється ставити ферзів на одну діагональ, якщо вони не сусідять по діагоналі, але не на один рядок або стовпчик. Клітини поля мають бути розфарбовані з ознакою ортогональної суміжності (клітини вважаються ортогонально суміжними, якщо вони межують по горизонталі або вертикалі). Модифікація задачі N-Queens є активною дослідницькою темою, що передбачає різноманітні модифікації класичної постановки з урахуванням додаткових обмежень, евристик та алгоритмічних стратегій, як, наприклад, в [1].

Постає задача процедурної генерації розфарбованого поля таким чином, щоб задача розміщення ферзів була розв'язною.

Для такого стохастичного розфарбовування поля, яке можна подати у вигляді двовимірної матриці, а кольори – у вигляді цифр, існує кілька відомих алгоритмів та підходів:

– градієнтний шум Перліна;

– клітинні автомати;

– хвильова заливка (queue flood fill);

– випадкове блукання (random walk).

Реалізація градієнтного шуму Перліна полягає в тому, що на полі розміром N x N в кожній клітині генеруються випадкові вектори-градієнти [2]. Значення в інших клітинах отримують за допомогою плавної інтерполяції в залежності від вектора. Після цього вводять поріг P у відсотках. Всі клітини, що мають значення вище порогу можуть бути зафарбовані, наприклад, чорним кольором, а нижче порогу – білим [3].

Алгоритм забезпечує гладкі контури та легко регулює масштаб «зернистості». Проте площа зон цілком залежить від одного-єдиного значення – порогу. Цей алгоритм розфарбовує поле у два кольори. Якщо потрібні N різних кольорових зон, доведеться вводити N–1 поріг, а спрогнозувати, скільки клітин відбере кожен колір, практично неможливо [4].

Отримані надмірно гладкі межі знижують якість згенерованого поля, що віддаляє його від ряду реальних задач.

Роботу алгоритму з використанням клітинних автоматів можна описати у два етапи [5]. На першому етапі створюється поле $N \times N$, де значення клітини задається випадково як 0 або 1 з початковою щільністю відношення кількості одиниць до розміру всього поля – ρ %. На другому етапі алгоритм змінює стан поточної клітини, спираючись на значення ортогональних сусідів. Для кожної клітини обчислюють правило переходу в інший стан в залежності від кількості ортогональних сусідів, значення яких дорівнюють одиниці. При цьому на вхід алгоритму можна подати число ітерацій зміни поточного значення клітини.

Алгоритм використовується для деяких задач стохастичного заповнення квадратного поля, але для вирішення описаної проблеми він не підходить: по-перше, він не гарантує

зв'язності отриманих областей, а по-друге, обмежений зафарбовуванням двома кольорами.

При використанні хвильової заливки (queue flood fill) для кожного кольору вибирається коренева клітина й додається до черги [6]. Крім того, задається ліміт кількості клітин, що розфарбовується одним кольором. Поки черга не порожня і ліміт не вичерпано, з черги виймається клітина, переглядаються значення її ортогональних сусідів. Якщо сусід нерозфарбований, він фарбується тим самим кольором і ставиться в кінець черги. Алгоритм простий в реалізації, забезпечує зв'язність зафарбованих областей і дає можливість задати ліміт.

Застосування цього алгоритму дає достатньо рівномірне розфарбовування, і передбачає багатократне перефарбовування клітин, що в підсумку може привести до нерозв'язності задачі, що розглядається. Крім того, при повному співпадінні стартових даних – ліміт, позиція стартових кольорів, черга розфарбовування – результат буде на 100 % детермінованим, що протирічить вимозі про стохастичність результату алгоритму.

Принцип випадкового блукання (random walk) полягає в такому. Спочатку на полі N x N випадковим чином розставляється m ($m=\overline{(2,N)}$) точок – «мурах» [7]. На кожному кроці «мураха» випадковим чином обирає одне з чотирьох ортогональних напрямків з однаковою ймовірністю й переходить у сусідню клітину, зафарбовуючи її власним кольором.

Цей алгоритм дозволяє розфарбувати поле в m кольорів. При цьому він має декілька критичних для розв'язання зазначеної проблеми недоліків: не гарантує зв'язності, може залишати незаповнені клітини, а «мурахи» здатні перефарбовувати клітини одна одної.

Проведене дослідження показало, що жоден із розглянутих алгоритмів генерації кольорового поля не гарантує створення конфігурації для розв'язання модифікованої задачі N-Queens.

Таким чином, жоден із згаданих підходів не формує поле, що задовольняє вимогам, потрібним для подальшого розв'язання цієї задачі. Це підкреслює актуальність розробки нового алгоритму генерації, здатного керовано створювати ортогонально області з контрольованим числом кольорів і відповідною геометрією.

3. Мета і задачі дослідження

Метою дослідження є розробка алгоритму, який дозволив би розфарбовувати двовимірні поля розміром N x N в N кольорів при виконанні таких вимог:

– повне заповнення (жодних порожніх клітин);

– зв'язність (розфарбована область має бути ортогонально зв'язною);

– різноманітний контур (якомога більш ламані форми);

– стохастичність (можливість одержати дві однакові розфарбовані форми поля при однакових стартових даних наближається до 0);

– фіксовані витрати (час і пам'ять залишаються не гірші за $O(N^2)$).

Для досягнення поставленої мети необхідно розв'язати такі задачі:

– розробити формальний набір метрик, що дозволяє кількісно оцінити виконання вимог до алгоритму, включаючи ступінь зв'язності, геометричну різноманітність, рівень стохастичності та ресурсну ефективність.

– сконструювати та протестувати алгоритм генерації, що забезпечує задані властивості при розфарбуванні поля.

4. Матеріали дослідження

Об'єктом дослідження є процес процедурного розфарбовування двовимірного дискретного простору з топологічними обмеженнями.

Предметом дослідження є алгоритмічні підходи до генерації кольорової конфігурації поля, що забезпечує структурну зв'язність та відповідність до заданих умов задачі.

Основною гіпотезою дослідження є припущення, що застосування модифікованого алгоритму генерації поля з урахуванням ортогональної суміжності та контролю кількості кольорових компонент дозволяє створити конфігурацію, придатну для розв'язання задачі оптимального розміщення елементів з додатковими умовами.

Для розроблюваного алгоритму було обрано назву алгоритм завоювання. Для алгоритму введено поняття «агент» і «завоювання». Агент – це сутність, якій має відповідати окрема територія двовимірного поля, а завоювання – процес почергового заповнення клітин поля різними агентами. Алгоритм має такі властивості:

– агенти можуть пропускати ходи. Ймовірність пропустити хід задається окремою функцією або константою;

– кожний доступний агенту напрям має свою вагу, яка впливає на ймовірність вибору напряму;

– агенти не мають права захоплювати клітини, які вже належать іншим агентам.

Це дозволяє впливати на роботу алгоритму за допомогою різних функцій керування, тобто завдання власних формул для пропуску ходу чи калькуляції ваги напряму. А неможливість захоплювати чужі клітини забезпечує зв'язність територій агентів.

5. Результати досліджень

Розроблений алгоритм представлено у вигляді послідовності таких кроків.

Крок 1. На вхід подається матриця, на якій розташовані агенти, що мають номери від 1 до N. Всі порожні клітини заповнені нулями.

Крок 2. Номери агентів випадково сортуються і додаються до черги, за якою агенти будуть виконувати ходи.

Крок 2.1. Агент отримує право ходу, як, наприклад, у грі з кидком кубика. Визначається мінімальне і максимальне значення для випадкової генерації. Визначається порогове значення, яке буде дорівнювати кількості клітин вже захоплених агентом територій. Якщо отримане випадкове число вище або дорівнює пороговому значенню, агент може отримати хід, а якщо нижче – пропускає хід.

Крок 2.2. Перевіряються сусідні клітини за ортогональними напрямами навколо агенту, що отримав право ходу. При цьому агент не може ходити по чужих територіях, або за межі поля.

Крок 2.3. Опції, доступні агенту (доступні напрями, серед яких буде обирати агент), отримують свою вагу – абстрактну величину, за якою опції сортуються. Опція кроку на вільну клітину отримує вагу 100, а опція руху на вже захоплену тим самим агентом клітину – 50, що дозволяє одразу відсортувати їх за цією величиною. Здійснюється вибір з урахуванням останнього напряму повороту.

Якщо опцій все ще декілька, вищу вагу отримують клітини, що ближче до кордонів, бо це дозволяє швидше захопити їх і уникнути білих плям, які потім перетворяться на фігури з високим рівнем прямокутності.

Крок 3. Якщо опцій для руху по порожніх клітинах нема, агент намагається знайти найближчі незахоплені клітини і рухатись до них по території, яка йому вже належить. Після кожного ходу агент оновлює власні дані про напрям останнього руху, його теперішню позицію і останній поворот.

Крок 4. Кроки 2 і 3 виконуються ітеративно, поки поле не буде заповнено повністю, тобто не залишиться порожніх клітин (нулів).

При виконанні Кроку 1 слід зазначити, що розташування агентів не є частиною роботи алгоритму, оскільки алгоритм передбачає рух вже існуючих агентів. Для поставленої задачі агентів треба розташувати за правилами задачі N-Queens у вигляді її потенційного рішення – тобто по 1 агенту в горизонталі, вертикалі із виключенням суміжності по

діагоналі.

При виконанні Кроку 2 слід зазначити: одна ітерація алгоритму передбачає, що кожен агент або отримає право ходу, або пропускає хід і передає можливість ходу далі по черзі.

При виконанні Кроку 2.1 мінімальне і максимальне значення для випадкової генерації було обрано, відповідно, 1 та N.

При виконанні Кроку 2.3 задля збереження складності задачі краще, щоб фігури мали різноманітні форми і поле не складалось просто з різних зафарбованих прямокутників. Саме тому, крім вибору за вагою, було додано вибір з урахуванням останнього напряму повороту, тобто якщо в агента є більше одного варіанта захопити нічийну клітину, можна відкинути опцію з поворотом у тому самому напряму.

У даній реалізації обрано було додавати до ваги 20 одиниць, що дозволяє посунути цю опцію вище відповідно до пріоритету, але не є достатнім, щоб рухатись по захоплених клітинах, якщо є можливість рухатись по порожніх.

Для поставленої задачі одною з ключових вимог є вимога отримати різноманітний контур розфарбованої фігури, тобто якомога більш ламані форми. Для оцінки цього та інших параметрів були винайдені такі спеціальні метрики, що дають можливість оцінити результати роботи алгоритму:

– середня витягнутість (Avg elongation) – середнє відношення довжини до ширини описаного прямокутника, де 1 означає квадрат.

– середня прямокутність (Avg rectangularity) – частка площі фігури у площі її мінімального описаного прямокутника, де 1 є повністю заповнений.

– середня зубчастість (Avg roughness) – периметр фігури, нормований на корінь площі. Чим більший показник, тим зубчастіший конутр.

– максимальна витягнутість (Max elongation) – максимальна витягнутість серед усіх фігур розфарбованого поля.

– максимальна прямокутність (Max rectangularity) — яка фігура найбільше заповнює прямокутник.

– мінімальна зубчастість (Min roughness) – найгладкіший контур на полі.

– доля витягнутих фігур (Share elongation) > 2,5 – відсоток фігур, що мають витягнутість понад 2,5. Впродовж виконання замірювань параметр (оптимальний поріг) було виявлено експериментально.

– складність (Steps / N^2) – середня кількість кроків алгоритму, поділена на площу поля. Дає нормовану оцінку обчислювальної складності.

Розроблений алгоритм завоювання було реалізовано програмно з використанням мови C#. Крім того, було реалізовано функціонал, який дозволив зібрати оцінки метрик після певної кількості ітерацій.

Для моделювання розфарбованого поля було використано матрицю, заповнену числами, де однакові числа демонструють певний колір. В процесі тестування було отримано можливість візуально оцінити заповнену зв'язними областями матрицю, створену за допомогою реалізованого алгоритму.

Тестові результати на 1000 прогонів трьох рівнів складності 8x8, 12x12 і 20x20 продемонстровано на рис. 1-3.

6. Обговорення результатів дослідження

Для оцінки результатів було зроблено по 1000 прогонів алгоритму для різних розмірів поля, що дало можливість переконатися в коректності роботи алгоритму та отримати уявлення про прогноз поведінки генерацій на довгій дистанції.

Загалом можна сказати, що алгоритм точно відповідає заданим вимогам: отримане поле повністю заповнено, території агентів зв’язні і мають різноманітний контур. Оскільки кожен

крок випадковий, то ймовірність отримати ідентичний результат майже дорівнює нулю.

```
===== BOARD 8x8 - 1000 runs =====

--- Test result: ---
Avg elongation      : 1.75 + 0.28
Avg rectangularity  : 0.74 + 0.06
Avg roughness       : 5.17 + 0.28
Max elongation      : 3.21
Max rectangularity  : 0.98
Min roughness       : 4.18
Share elong > 2.5    : 11.4%
Steps / N^2          : 3.17
```

Рис. 1. Тестові результати для складності 8x8

```
===== BOARD 12x12 - 1000 runs =====

--- Test result: ---
Avg elongation      : 1.77 + 0.26
Avg rectangularity  : 0.65 + 0.05
Avg roughness       : 5.74 + 0.33
Max elongation      : 3.74
Max rectangularity  : 0.96
Min roughness       : 4.27
Share elong > 2.5    : 12.1%
Steps / N^2          : 5.93
```

Рис. 2. Тестові результати для складності 12x12

```
===== BOARD 20x20 - 1000 runs =====

--- Test result: ---
Avg elongation      : 1.78 + 0.22
Avg rectangularity  : 0.56 + 0.04
Avg roughness       : 6.47 + 0.31
Max elongation      : 4.42
Max rectangularity  : 0.93
Min roughness       : 4.36
Share elong > 2.5    : 12.5%
Steps / N^2          : 13.75
```

Рис. 3. Тестові результати для складності 20x20

Приклад точкової генерації поля 8x8 наведено на рис. 4.

```
5 5 2 2 2 6 6 6
7 5 2 4 4 4 4 4
7 5 3 3 3 3 3 4
7 5 3 3 3 3 1 4
7 5 3 3 3 3 1 1
7 5 5 5 8 8 8 1
8 8 8 8 8 8 8 1
8 8 8 8 8 8 1 1
```

Рис. 4. Приклад точкової генерації поля 8x8

Для оцінки результатів генерацій за зазначеними вище метриками використовувався метод Монте-Карло – багаторазове виконання генерації на однакових вхідних параметрах із подальшим збором і статистичним аналізом метрик. Результати вимірів і їхнього аналізу наведено в табл. 1.

З цієї таблиці і поданих коментарів можна зробити висновок, що всі метрики змінюються прогнозовано, отже за описаними вимогами алгоритм є стабільним і ефективним.

Таблиця 1

Результати вимірів метрик

Назва метрики	Розмір поля			Коментар
	8 × 8	12 × 12	20 × 20	
Average elongation (mean ± σ)	1.75 ± 0.28	1.77 ± 0.26	1.78 ± 0.22	Спостерігається помірне зростання. Більші поля дають трохи витягнутіші області
Average rectangularity	0.74 ± 0.06	0.65 ± 0.05	0.56 ± 0.04	Чіткий спад – контур стає більш схожим на дерево
Average roughness	5.17 ± 0.28	5.74 ± 0.33	6.47 ± 0.31	Лінійне зростання – контур стає зубчастішим
Maximum elongation	3.21	3.74	4.42	На великих полях з'являються довші «коридори»
Maximum rectangularity	0.98	0.96	0.93	Прямокутні фігури менш щільні на великих N
Minimum roughness	4.18	4.27	4.36	Мінімум гладкості контурів повільно зростає
Share elongation > 2.5 (%)	11.4	12.1	12.5	Частка довгих вузьких фігур майже стабільна (погрішність близько до 1%)
Steps / N^2	3.17	5.93	13.75	Підтверджує $O(N^2)$

Слід зауважити, що в запропонованому алгоритмі не реалізовано засоби контролю за формою утворюваних фігур. Таким чином, геометрія територій, що захоплюються агентами, визначається виключно правилами завоювання та випадковістю вибору напрямів.

В подальшому дослідженні можна оптимізувати правила вибору опцій агентами для отримання різноманітніше заповненого поля.

7. Висновки

У дослідженні було розглянуто і проаналізовано проблему стохастичного розфарбовування двовимірних полів. Було проаналізовано існуючі алгоритми і причини, через які їх не завжди можна використовувати в деяких задачах, а також згадані приклади таких задач.

За результатами вирішення поставлених задач дослідження отримано алгоритм завоювання, що демонструє стабільну зв'язність територій агентів та гнучкість налаштувань напрямів руху. Сутність отриманого результату полягає у тому, що дії агентів регулюються функціями ймовірності та вагами можливих опцій ходу, що дозволяє моделювати різні сценарії розподілу територій.

Роботу алгоритму було оцінено за допомогою фіксованих математичних метрик, які довели, що алгоритм зберігає свою надійність і має прийнятну складність при зміні величини сторони поля N.

У порівнянні з класичними методами розподілу, запропонований алгоритм забезпечує природніше формування територій без перетину володінь агентів, що сприяє покращенню узгодженості та уникненню конфліктних зон.

Задача заповнення двовимірних полів є достатньо поширеною і може бути застосована для різних областей. В залежності від напряму задачі, можна варіювати деякі параметри розробленого алгоритму, такі як значення випадкової генерації, порогове значення, пріоритет клітини тощо.

Перелік посилань:

1. Sinha, R., Kaur, N., Gupta, S., Thakur, P. (2025). N-Queen Problem Solution Using Modified Genetic Algorithm. In: Bajaj, A., Sreedhar, S., Abraham, A. (eds) Bio-Inspired Computing. IBICA 2023. Lecture Notes in Networks and Systems, vol. 1230. Springer, pp. 201–210.

2. Perlin K. (1985) An Image Synthesizer. ACM SIGGRAPH Computer Graphics, vol. 19, no. 3, pp. 287–296, https://doi.org/10.1145/325334.325247.
3. Кудінов І. П., Ягодкін Д. (2025). Алгоритми процедурної генерації шуму Перліна. Зб. тез наук. доп. здобувачів вищої освіти Бердян. держ. пед. ун-ту, Запоріжжя, Україна, трав. 2025, с. 127-130. https://doi.org/10.5281/zenodo.15548937.
4. Kopel, M., Maciejewski, G. (2020). Comparison of Procedural Noise-Based Environment Generation Methods. In: Nguyen, N.T., Hoang, B.H., Huynh, C.P., Hwang, D., Trawiński, B., Vossen, G. (eds) Computational Collective Intelligence. ICCCI 2020. Lecture Notes in Computer Science, vol 12496. Springer, Cham. pp 878–887.
5. Wu Z., Mao Y., Li Q. (2021). Procedural Game Map Generation using Multi-leveled Cellular Automata by Machine Learning, in Proc. ISAIMS 2021, Chongqing, China, Oct. 2021, https://doi.org/10.1145/3500931.3500962.
6. Fellows M. R., Rosamond F. A., da Silva M. D., Souza U. S. (2021). A Survey on the Complexity of Flood-Filling Games. Discrete Appl. Math., vol. 304, pp. 233–246, Dec. 2021, https://doi.org/10.1016/j.dam.2021.09.029.
7. Popp S., Dornhaus A. (2023). Ants Combine Systematic Meandering and Correlated Random Walks when Searching for Unknown Resources. iScience, vol. 26, no. 2, 2023, Art. no. 105916, https://doi.org/10.1016/j.isci.2022.105916

Надійшла до редколегії 10.06.2025 р.

Красніков Влад Валерійович, старший .Net розробник, компанія Pricer24, м. Харків, Україна, e-mail: krasnikov.vlad.v@gmail.com
Ситнікова Поліна Едуардівна, кандидат технічних наук, доцент, доцент кафедри системотехніки ХНУРЕ, м. Харків, Україна, e-mail: polina.sytnikova@nure.ua, ORCID: https://orcid.org/0000-0002-6688-4641

УДК 004.8:004.9 DOI: 10.30837/0135-1710.2025.185.077

С.Ф. ЧАЛИЙ, Р. В. КРАВЧЕНКО

ГРАФОВА НЕЙРОННА МЕРЕЖА ДЛЯ ТЕМПОРАЛЬНО УПОРЯДКОВАНИХ ДАНИХ В ЗАДАЧІ ПОБУДОВИ ПОЯСНЕНЬ В ІНТЕЛЕКТУАЛЬНІЙ СИСТЕМІ

Об'єктом дослідження є процес побудови пояснень в інтелектуальних інформаційних системах. Предметом дослідження є моделі та методи формування пояснень в інтелектуальних інформаційних системах. Метою роботи є розробка підходу до побудови пояснень в інтелектуальних системах на основі графових нейронних мереж, які враховують темпоральний порядок у вхідних даних. Розроблено модель графової нейронної мережі для темпорально упорядкованих даних; виконано експериментальну перевірку графової мережі в задачі побудови пояснень для системи електронної комерції. Запропонована модель графової нейронної мережі включає функціональні блоки побудови векторних представлень, виявлення темпоральних патернів з використанням мережі LSTM, формування графа мережі для заданих темпоральних інтервалів, підготовки пояснень, прогнозування, генерації пояснень з використанням механізму уваги та результатів агрегації векторного представлення мережі.

1. Вступ

Інтелектуальні інформаційні системи (ІІС) використовують методи машинного навчання для того, щоб апроксимувати складні залежності в даних й використати отримані залежності для формування рішень. Проте внаслідок непрозорості таких процесів користувачі не завжди можуть зрозуміти логіку формування рішень в ІІС [1], [2]. Непрозорість процесу формування рішень призводить до зниження довіри до результатів роботи ІІС і, як наслідок, може привести до обмеження використання отриманих рішень користувачами [3].

Для вирішення проблеми зниження довіри до ІІС в галузі штучного інтелекту розвивається науковий напрямок пояснювального штучного інтелекту (Explainable Artificial Intelligence – XAI) [4]. Напрямок XAI орієнтований на розробку моделей та методів, що забезпечують пояснення процесу та результатів роботи ІІС, забезпечуючи зрозумілість для