ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

Навчально–науковий інститут Інформаційних технологій

Кафедра Інженерії програмного забезпечення

Пояснювальна записка

до магістерської роботи
на ступень вищої освіти магістр

на тему «**МОДЕЛЬ ПОВЕДІНКИ ЛЮДЕЙ В ПРИМІЩЕННІ З ВИКОРИСТАННЯМ КЛІТИННИХ АВТОМАТІВ»**

Виконав: студент 5 курсу, групи ППЗМ - 71
спеціальності
121 Інженерія програмного забезпечення
(шифр і назва спеціальності)
Яцков Андрій Сергійович
(прізвище та ініціали)
Керівник Золотухіна О.А.
(прізвище та ініціали)
Рецензент
(прізвище та ініціали)

КИЇВ – 2021

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

Навчально–науковий інститут Інформаційних технологій

Кафедра Інженерії програмного забезпечення
Ступінь вищої освіти «Магістр»
Спеціальність підготовки 121 Інженерія програмного забезпечення

ЗАТВЕРДЖУЮ
Завідувач кафедри
Інженерії програмного
забезпечення
_____О.В.Негоденко
“____” _____ 2021 року

З А В Д А Н Н Я
НА МАГІСТЕРСЬКУ РОБОТУ СТУДЕНТУ

Яцков Андрій Сергійович
(прізвище, ім’я, по батькові)

1.Тема роботи: «Модель поведінки людей в приміщенні з використанням клітинних автоматів

Керівник роботи Золотухіна Оксана Анатоліївна, к.т.н.,
(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)
затверджені наказом вищого навчального закладу від « 12 » березня 2021 року № 65

2. Строк подання студентом роботи «01 » червня 2021року

3. Вихідні дані до роботи: Матеріали переддипломної практики, методи моделювання, пішохідних потоків, нормативні акти щодо обмежень соціальної поведінки в умовах пандемії COVID-19

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити).
4.1 Огляд предметної області
4.2 Розробка моделей та методів
4.3 Розробка програмного забезпечення моделей
4.4 Проведення моделювання та аналіз отриманих результатів

5. Перелік графічного матеріалу (презентація)
5.1 Мета, об’єкта та предмет дослідження
5.2 Клітинний автомат
5.3 Модель приміщення
5.4 Модель покупця
5.5 Узагальнений алгоритм проведення моделювання

6. Дата видачі завдання «12» квітня 2021року

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів бакалаврської роботи	Строк виконання етапів роботи	Примітка
1	Отримання завдання на магістерську роботу	12.04.2021	
2	Огляд предметної області	17.04.2021	
3	Аналіз методів та засобів клітинних автоматів	20.04.2021	
4	Розробка моделей та методів	22.04.2021	
5	Розробка програмного забезпечення моделей	23.04.2021	
6	Тестування програмного забезпечення	24.04.2021	
7	Моделювання та аналіз результатів	26.04.2021	
8	Написання та оформлення пояснювальної записки	27.04.2021	
9	Розробка графічних та презентаційних матеріалів	11.05.2021	
10	Захист магістерської роботи	01.06.2021	

Студент _____ Яцков А.С.
(підпис) (прізвище та ініціали)

Керівник роботи _____ О.А. Золотухіна
(підпис) (прізвище та ініціали)

РЕФЕРАТ

Текстова частина магістерської роботи: 69с., 26 рис., 2 дод., 15 джерел.

МОДЕЛЬ, КЛІТИННИЙ АВТОМАТ, ПРИМІЩЕННЯ, ПОВЕДІНКА, ДИСТАНЦІЯ, ЕВАКУАЦІЯ, ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ, C++

Об`єкт дослідження – процес побудови моделей поведінки людей.

Предмет дослідження – моделі та закономірності поведінки людей у приміщенні в умовах соціального дистанціювання та екстремальних ситуацій.

Мета роботи – спрощення процесу отримання та аналізу даних про поведінку людей у замкнутому просторі з перешкодами в умовах соціального дистанціювання та у екстремальних ситуаціях за рахунок використання принципів функціонування клітинних автоматів.

Методи дослідження – методи теорії систем та моделей клітинних автоматів, методи теорії інформації, апарат математичної статистики, методи проектування та розробки програмного забезпечення, технології об’єктно-орієнтованого програмування.

У роботі проведено аналіз сучасного стану використання моделей клітинних автоматів та теорії масового обслуговування у сферах моделювання поведінки та переміщення людей, виконано огляд існуючого програмного забезпечення та проведено аналіз використаних моделей, оглянуто питання обмежень та рекомендацій що до поведінки людей та норм утримання приміщень в умовах пандемії коронавірусной інфекції COVID-19. Здійснено розробку моделі та алгоритмів поведінки людей у приміщенні торгових залів з урахуванням необхідності дотримання соціальної дистанції та у випадку екстреної евакуації.

Розроблено програмне забезпечення моделі поведінки людей, яке дозволяє отримувати та зберігати статистичні дані результатів моделювання та надає покрокову візуалізацію процесу переміщення.

Проведено аналіз отриманих результатів моделювання та визначено закономірності поведінки людей у торгових залах, що пов’язані з принципами розташування перешкод та виконанням загального алгоритму поведінки покупців.

ЗМІСТ

Стор.

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ **8**

ВСТУП **9**

1 АНАЛІЗ ПІДХОДІВ ДО МОДЕЛЮВАННЯ ПОВЕДІНКИ ЛЮДЕЙ В ПРОЦЕСІ ЇХ РУХУ В ПРИМІЩЕННЯХ **11**

1.1. Загальна класифікація підходів до моделювання руху людей 11

1.2. Клітинні автомати 13

1.3 Фізико-математичні моделі 16

1.4 Теорія масового обслуговування 18

1.5 Багатоагентні моделі 25

1.6 Аналіз програмних засобів для моделювання руху людей 28

2 РОЗРОБКА МОДЕЛІ ПОВЕДІНКИ ЛЮДЕЙ В ПРИМІЩЕННІ З ВИКОРИСТАННЯМ КЛІТИННИХ АВТОМАТІВ **33**

2.1 Постановка задачі моделювання руху людей з урахуванням протиепідемічних заходів в умовах пандемії коронавірусной інфекції COVID-19 33

2.2 Розробка моделі приміщення 34

2.3 Розробка моделі покупця 37

2.4 Математична модель клітинного автомату 42

2.5 Програмна модель 43

2.6 Програмна реалізація моделі поведінки людей 48

2.6.1 Опис використаних програмних засобів 49

2.6.2 Опис структури проекту 50

2.6.3 Опис інтерфейсу 51

2.6.4 Опис розроблених класів 55

3 ПРОВЕДЕННЯ МОДЕЛЮВАННЯ ТА АНАЛІЗ РЕЗУЛЬТАТІВ **63**

ВИСНОВКИ **67**

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ **68**

Додаток А. ДЕМОНСТРАЦІЙНІ МАТЕРІАЛИ (Презентація) **70**

Додаток Б. ПРОГРАМНИЙ КОД ОСНОВНИХ МОДУЛІВ СИСТЕМИ .. 75

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

IEC	-	International Electrotechnical Commission
ISO	-	International Organization for Standardization
QML	-	Qt Modeling Language
XML	-	Extensible Markup Language
МОЗ	-	Міністерство охорони здоров'я
СМО	-	Система масового обслуговування

ВСТУП

Використання моделей клітинних автоматів - це сучасний напрям, який активно розвивається та використовується у багатьох теоретичних та практичних сферах людського життя.

Застосування клітинних автоматів можна зустріти при моделюванні екосистем, популяційної динаміки, криптографії, при моделюванні фізичних, соціальних та економічних процесів, транспортних та пішохідних потоків, та поведінки людей. Саме побудові та дослідженню моделі поведінки людей в умовах замкнутого приміщення присвячена ця робота.

Встановлення карантинних обмежень, необхідність дотримуватися соціальної дистанції, а подекуди режиму самоізоляції у зв'язку з поточною пандемією коронавірусной інфекції COVID-19, яка викликана коронавірусом SARS-CoV-2 суттєво змінили поведінкові шаблони людей, їх звички та традиційний спосіб життя. Однак уникнення відвідування торгівельних точок повністю неможливе, тому актуальним стає питання забезпечення безпеки людей у приміщеннях торгівельних залів магазинів та супермаркетів. Моделювання поведінки людей в таких умовах надає можливості отримати дані, що до таких зон ризику, як, наприклад, зони швидкого забруднення або зони потенційного порушення соціальної дистанції, проаналізувати їх та застосовувати відповідні заходи що до зменшення ризику зараження.

Таким чином, завдання розробки моделі поведінки людей в приміщенні є сучасним та актуальним.

Об`єкт дослідження – процес побудови моделей поведінки людей.

Предмет дослідження – моделі та закономірності поведінки людей у приміщенні в умовах соціального дистанціювання та екстремальних ситуацій.

Мета роботи – спрощення процесу отримання та аналізу даних про поведінку людей у замкнутому просторі з перешкодами в умовах соціального дистанціювання та у екстремальних ситуаціях за рахунок використання принципів функціонування клітинних автоматів.

Методи дослідження – методи теорії систем та моделей клітинних автоматів, методи теорії інформації, апарат математичної статистики, методи проектування та розробки програмного забезпечення, технології об’єктно-орієнтованого програмування.

Практична значущість результатів полягає в використанні розробленої моделі для отримання статистичних даних що до переміщення людей по приміщенням в загальних та екстремальних умовах, вивчення закономірності поведінки людей у торгових залах, аналізу впливу схем розміщення елементів торгових залів та умов соціального дистанціювання на принципи пересування та взаємодію людей.

Для досягнення мети вирішувалися наступні завдання.

1. Аналіз існуючих підходів до побудови та використання клітинних автоматів у сфері моделювання соціально-поведінкових аспектів життя людини.

2. Дослідження алгоритмів поведінки людей у приміщеннях.

3. Розробка моделі поведінки людей у приміщенні торгового залу в умовах соціального дистанціювання та у екстремальних ситуаціях.

4. Розробка програмного забезпечення моделі.

5. Проведення моделювання поведінки людей у приміщенні та аналіз отриманих результатів.

1 АНАЛІЗ ПІДХОДІВ ДО МОДЕЛЮВАННЯ ПОВЕДІНКИ ЛЮДЕЙ В ПРОЦЕСІ ЇХ РУХУ В ПРИМІЩЕННЯХ

1.1. Загальна класифікація підходів до моделювання руху людей

Підходи до аналізу руху (людей, транспортних засобів тощо) настільки різноманітні по своїй суті, на скільки по різному відносяться до них різноманітні категорії зацікавлених осіб (органи влади, проектувальники, громадськість, професійні спільноти). Відомо достатньо багато апробованих методів аналізу та моделювання транспортних і пішохідних потоків, класифікація моделей зображена на рисунку 1.1.

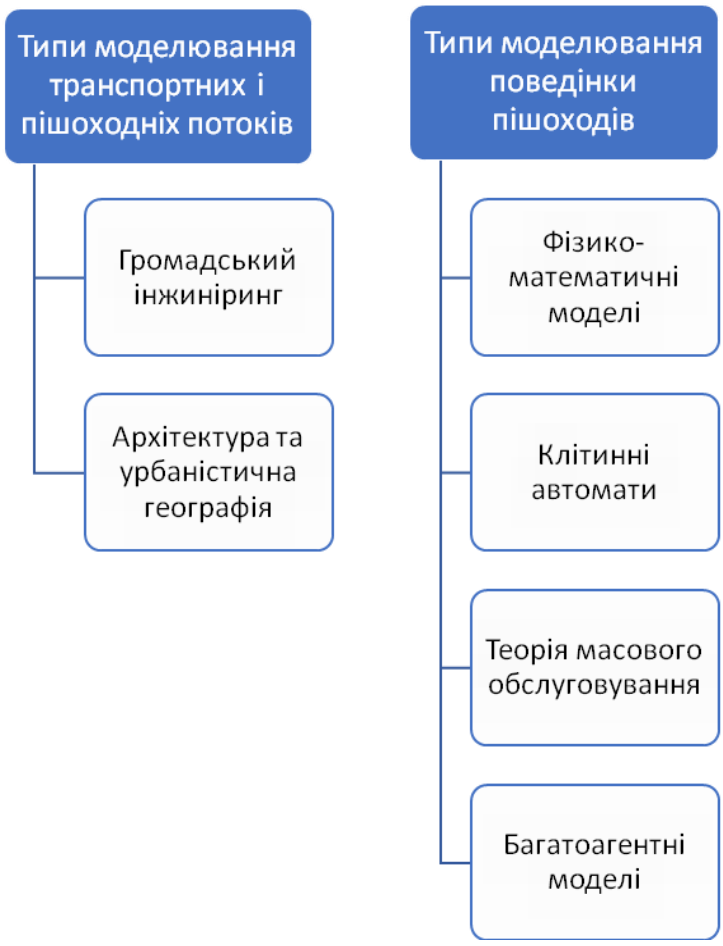


Рисунок 1.1 – Класифікація моделей

В частині моделювання пішохідного руху вони реалізуються на різних принципах, які виділили два основних підходи до дослідження. Перший з них: громадський інжиніринг, який використовують для будови моделі транспортного попиту та служить базою при наступнім конструюванні транспортних моделей у цілому, тобто моделей взаємодії транспортного попиту та транспортної пропозиції. Моделі подібного типу у громадському інжинірингу використовуються для вирішення задач проектування нової транспортної інфраструктури. Результатом побудови моделі транспортного попиту є матриця кореспонденцій і матриця витрат. Осередки матриці визначають кількість людей, які переміщуються по різноманітних маршрутах. Процес моделювання переміщення людей заснований на стандартній чотирьох кроковій транспортній моделі. Такі моделі по характеру – макроскопічні, оскільки елементи нижчого порядку у них – це локація відвідувана індивідами, а також маршрути, які для цього призначені. Другий підхід базується на архітектурі і урбаністичній географії. Прихильник даного підходу виділяє інтерес до того, як розміщення місць тяжіння впливає на пішохідний рух. Такі моделі, як правило є мікроскопічними, з великою кількість деталей. У загальному випадку, вони розроблюються для не великих територій, хоча іноді можуть бути екстрапольовані на території усього міста. Ряд моделей поєднує у собі обидва визначених підходи, що дозволяє їм демонструвати достатню гнучкість у відношенні різноманітних типів територій.

У рамках підходів, що досліджують рольові особливості індивідів на мікрорівні, поведінка кожного окремого пішохода моделюється на основі встановлених загальних правил. Така методика універсальна і підходить для оцінки різноманітних ситуацій: закритий простір або незвичайний поток людей, у яких поведінка окремого індивіда схильні сильному впливу геометрії, хаотичності, соціальних переваг, локальної та колективної поведінки для інших індивідів.

Існує чотири основні підходи до моделювання поведінки пішоходів: клітинні автомати, фізико-математичні моделі, теорія масового обслуговування і багатоагентні моделі.

1.2. Клітинні автомати

Клітинний автомат уявляє собою дискретну динамічну систему, сукупність однакових клітин, з'єднаних між собою, поведінка яких повністю визначається локальними взаємозв’язками клітин [1].

Вперше ідеї клітинних автоматів були використані у 1940-х роках Станіславом Уламом при вивченні зростання кристалів та Джоном фон Нейманом при роботі над проблемою систем, що самоутворюються, а Норберт Вінер і Артуро Розенблют розробили клітинно-автоматну модель середовища, що збуджується з метою побудови математичного опису поширення імпульсу в серцевих нервових вузлах. Певні види клітинних автоматів вивчалися ще біля двох десятиліть, однак вони не мали широко застосування до 1970-х років, коли здобула популярність двовимірна клітинно-автоматна модель з двома станами, розроблена Джоном Конвеем, популяризована Мартіном Гарднером та відома як гра «Життя».

Модель клітинного автомату описується періодичною решіткою, яку утворює набор клітин, та заданим набором правил переходу по клітинам решітці. Під правилами переходу розуміють визначення стану клітини в залежності від поточного стану самої клітини та стану клітин-сусідів. Під клітинами-сусідами розуміють клітини, які знаходяться на певної відстані від певної, не перевищуючи при цьому заданої максимальної відстані. У загальному випадку клітинні автомати мають наступні властивості:

- зміни значень всіх клітин відбуваються одночасно після обчислення нового стану кожної клітини решітки;
- решітка однорідна – неможливо розрізнити будь-які дві області решітки;

− взаємодії локальні, лише клітини околиці здатні вплинути на дану поточну клітину;

− безліч станів клітини кінцева.

Формально клітинний автомат можна визначити як набір

$$\{G, Z, N, f\}, \tag{1.1}$$

де G – метрика поля, на якому діє автомат;

Z – множина станів кожної клітини

N – околиця клітини, яка впливає на стан поточної клітини;

f – правила клітинного автомату.

Математично клітинний автомат визначається як безліч кінцевих автоматів на площині [2], які позначені цілочисельними координатами (i, j) та кожен з яких може перебувати в одному з станів $\sigma_{i,j}$:

$$\sigma_{i,j} \in \sum \equiv \{0,1,2, \dots k-1, k\}. \tag{1.2}$$

Зміна станів автоматів відбувається згідно з правилом переходу:

$$\sigma_{i,j}(t+1) = \varphi(\sigma_{k,l}(t) | (k,l) \in N(i,j), \tag{1.3}$$

де $N(i,j)$ – деяка околиця точки (i,j), ,безліч автоматів, що становлять сусідство. Наприклад, околиця фон Неймана порядку 1 наведена на рисунку 1.1а та визначається наступним чином:

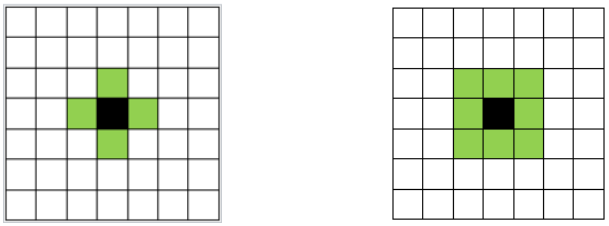
$$N_{FN}^{1}(i,j) = \{(k,l) || i-k| + |j-l| \leq 1\}, \tag{1.4}$$

а околиця Мура порядку 1 на рисунку 1.1б та визначається:

$$N_{M}^{1}(i,j) = \{(k,l) || i-k| \leq 1, |j-l| \leq 1\}. \tag{1.5}$$

Кількість всіх можливих правил переходу визначається кількістю станів σ та кількістю сусідів n і становить:

$$N_r = \sigma^{\sigma^n}. \tag{1.6}$$



а) б)

Рисунок 1.1 – Околиці порядку 1 фон Неймана (а) та Мура (б)

Область застосування моделей клітинного автомату дуже широка: від найпростіших ігор, як «хрестики-нулики», до розробки у сфері штучного інтелекту. Сучасний інтерес до вивчення та використання клітинних автоматів розширився далеко за межі академічної науки, що зумовлено стрімким розвитком інформаційних технологій та підвищенням потужностей обчислювальних систем. Застосування клітинних автоматів можна зустріти при моделюванні екосистем і популяційної динаміки, проектуванні комп’ютерних процесорів: елементи розміщуються на рівномірній сітці однакових осередків, а їх взаємодія реалізується за допомогою електричного струму, магнетизму або вібрації; у криптографії у якості блочного шифру, для генерації випадкових чисел та використання в криптосистемах з відкритим ключем, при моделюванні фізичних процесів та у фундаментальних фізичних дослідженнях нашого світу, який добре описується фізикою елементарних частинок та може виявитися клітинним автоматом на фундаментальному рівні.

Доволі широкою сферою використання клітинних автоматів є економіка, урбаністика та соціологія де різні моделі клітинних автоматів використовуються для симуляції соціально-економічних процесів, транспортних та пішохідних потоків, поведінки людей у певних умовах з метою подальшого вивчення та аналізу. Саме побудові та дослідженню моделі поведінки людей в умовах замкнутого приміщення присвячена ця робота.

1.3 Фізико-математичні моделі

В основі фізико-математичних моделей лежать математичні та фізичні рівняння. Наприклад, одна з них будуються на фізичній формулі руху, де умовні пішоходи мають такі характеристики, як поточне місце розташування і швидкість. Поняття сили використовується для пояснення рухів пішоходів. У ході такі показники сили, як прискорення, ефекти відштовхування від межі і об'єктів, від інших пішоходів, ефекти при тяжіння іншим групам і об'єктам.

Д. Хелбінг [3] використав поняття тяжіння і відштовхування для моделювання мікроповедінки, розробив комплексні рівняння для різноманітних варіантів поведінки пішоходів. Ці розробки отримали широку відомість, як концепція «соціальних сил», у рамках якої при моделюванні враховується ряд внутрішніх передумов визначених дій (рухів) індивіда, а також їх наступний вплив на динамічні атрибути пішоходів (швидкість, прискорення, інтервали). Він помітив, що потоки пішоходів мають властивість збиратися у натовпи, а потів знову розпадатися на окремі потоки.

С.Хугендум і П.Бови [4] застосували ті ж базові формули для створення трьохрівневих моделей, охоплюючий процеси вибору дій, визначення траєкторії слідування та пересування. Ця модель передбачає мінімалізацію витримок переміщення, вона була затребувана при аналізу функцій мультимодальних пересадочних вузлів. Ще одна модель цього ряду опирається на статичні методи оцінки потоків на визначених направленнях.

Математичне моделювання, як елемент навчальної технології, реалізується у змісті курсу фізики, в унаочненні фізичних теорій, законів, у взаємозв'язках між параметрами фізичних теорій. На предметному рівні математичне моделювання виступає і методом, і засобом дослідження фізичного процесу. На дидактичному рівні математичне моделювання є складовою цілісної педагогічної технології як загальнонауковий метод дослідження. Фізико-математичні моделі пішохідних потоків реалізовані у таких програмних продуктів, як PTVVision®, Viswalk, Citilabs Cube Dynasim,Quadstone Paramics, SIMWALK.

Найбільш дослідженими є двовимірніклітинні автомати, наприклад гра «Життя» та мураха Ленгтона.

Клітинні автомати використовуються, коли на умовному клітковому полі кожен пішохід займає одну клітину і переміщується у співвідношенні з рядом простих правил у рамках цього поля. Подібний варіант традиційних моделей кліткових автоматів (приватний випадок матричних систем) припускає, що стан кожної клітки змінюється у залежності від стану навколишніх її клітин, причому відсутність явного руху. У цих моделях використовується матриця, у яких один осередок може займати тільки один індивід, а умовне кодування великих територій потребує матриць високого розміру

Більшість моделей, в основі яких лежить даний принцип, мають на увазі підхід Шрекенберга-Нагеля [5] для моделювання дорожнього потоку з використанням кліткових автоматів у якості базових інструментів. В останній час кліткові автомати часто використовуються при моделюванні поступального руху натовпу, який покидає масові спортивні заходи.

Модельоване середовище представляється у вигляді двох шарів: статистичного, на якому відображаються існуючі виходи з приміщенням і з територій, і динамічного, що містить дані про загальний вектор руху натовпу. Кожен індивід споживає інформацію на рівну своєї особистої клітки, щоб обрати вектор подальшого руху.

Проблема моделей кліткових автоматів полягає у не визначеності рухів у випадку появи натовпу, так як такий стан не представлений у традиційних моделях. Кліткові автомати продемонстрували свою ефективність для дезареєстрованих моделей з мінімальною варіативністю вибору дій. Модель Alp-Sim поєднує підхід кліткових автоматів з сукупністю моделюванням середовищем, щоб отримати переваги за рахунок відображення різноманітних типів даних у вигляді набору карток, у особливості для цілей планування більш високого рівня, де одних клітинних решіток не достатньо.

Ряд дослідників домоглися значних успіхів у цій області. Так В.Блю [6] використав принцип клітинних автоматів для створення моделі пішохідних проходів з рухом у двох напрямках.

Отримані ним результати свідчать про те, що навіть малий набір правил здатен ефективно передати поведінку пішоходів на мікро рівні. Інший спеціаліст, С. Сармаді [7] поєднав поведінкову модель, стимулюючу рухи окремих пішоходів і модель кліткових автоматів, стимулюючу переміщення пішоходів на малих просторах.

1.4 Теорія масового обслуговування

Третій варіант моделювання пішохідного руху опирається на теорії масового обслуговування. Тут прийнято припущення, що усі пішоходи знаходяться під контролем суб'єкта, який вказує їм час і напрямок руху. Такий прийом корисний для загальних моделей, оскільки дозволяє звести воєдино усі необхідні дані. У подібних моделях для умовного представлення середовища беруться основні елементи теорії графів, де можливі коридори руху представлені у вигляді ребер, а точки прийняття рішення – у вигляді вершин.

Теорія масового обслуговування описує процеси, що протікають в системах масового обслуговування (СМО). До СМО відносяться ремонтні майстерні, станції технічного обслуговування, автозаправні станції тощо, система масового обслуговування зображена на рисунку 1.2.

Випадковий характер потоку заявок призводить до того, що в СМО відбувається якийсь випадковий процес. Якщо випадковий процес Марківський, то функціонування СМО можна описати системою диференціальних рівнянь, а в граничному випадку - системою лінійних алгебраїчних рівнянь, рішенням яких визначаються характеристики роботи СМО.

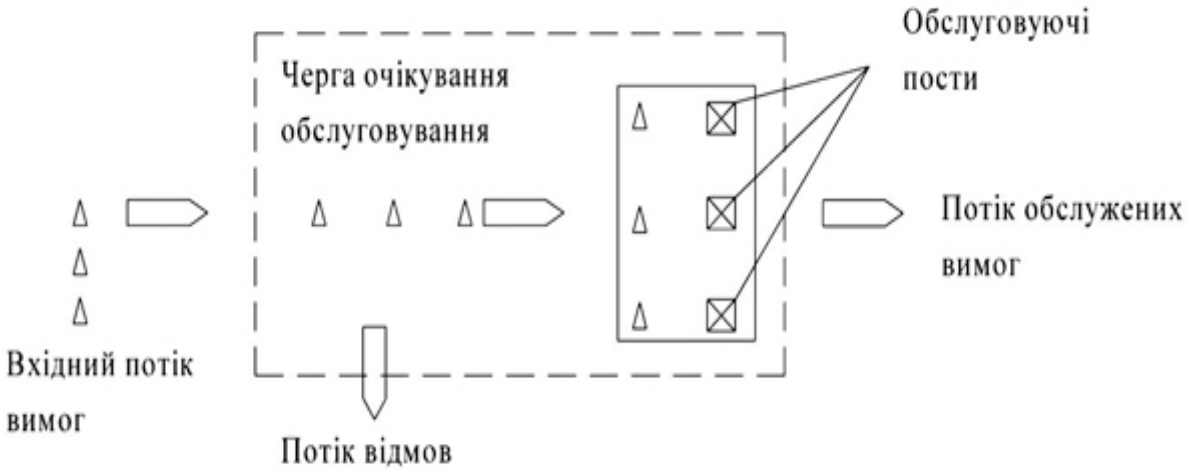


Рисунок 1.2 - Система масового обслуговування

При аналізі роботи СМО необхідно знати її основні вихідні параметри:

- інтенсивність потоку заявок - λ;
- трудомісткість обслуговування однієї заявки - Тр;
- число каналів обслуговування - *n*;
- число місць очікування - *m*;
- кількість операторів на кожному каналі - d;
- умови, що накладаються на освіту черги.

При описі режиму роботи СМО використовуються проміжні параметри:

- час обслуговування однієї заявки - T_0=Тр/d;
- продуктивність кожного каналу обслуговування - $t_0 = \text{Тр}/d$; (середнє число заявок, яке обслуговується каналом в одиницю часу);
- приведена інтенсивність обслуговування $\mu = 1/t_0$;
- коефіцієнт завантаження $\alpha = \lambda/\mu$.

Системи масового обслуговування поділяються по ряду ознак:

1. За часом очікування (T_x):

- без очікування або з втратами вимог (T_x=0);
- з очікуванням або без втрат ($T_x = 0 - \infty$);
- з обмеженим часом очікування ($T_x = 0 - \tau$).

Системи з необмеженим очікуванням початку технічного впливу є найбільш реальним. Це обумовлено тим, що в умовах України перегони (нульові пробіги) до поста обслуговування досить великі. Існує дефіцит запчастин і матеріалів, відсутня оперативний зв'язок між регіональними СТОА, не розвинені фірмові системи сервісу і т.п. Тому клієнт змушений чекати початку обслуговування іноді і поза зони очікування.

2. За кількістю вимог на добу (N_c):

- з обмеженим вхідним потоком, або замкнуті ($N_c = 0...k$);

- з необмеженим потоком, або відкриті ($N_c = 0...\infty$).

Число вимог на СТОА не може бути заздалегідь обмежена, системи є відкритими (розімкненими).

3. За кількістю каналів (обслуговуючих апаратів):

- з обмеженим числом обслуговуючих апаратів ($x = 0...n$);

- з необмеженим числом обслуговуючих апаратів ($x = 0...\infty$).

4. За кількістю фаз (r):

- однофазні (r=1);

- багатофазні (r>1).

Багатофазні СМО використовуються при потоковому вигляді обслуговування (наприклад, діагностика - ТО).

5. За рівнем організації:

- упорядковані (ентропія E=1);

- невпорядковані (E > 1).

Порядок обслуговування встановлюється в результаті аналізу СМО.

При розгляді питань функціонування СМО є два основні види: СМО з відмовами; СМО з очікуванням обслуговування.

У системах з відмовами обслуговуються тільки ті вимоги, які надходять в момент часу, коли хоча б один з каналів обслуговування був вільний. Якщо всі канали зайняті, то заявка покидає СМО не обслуженою.

У СМО з очікуванням заявка, що надійшла в момент, коли всі канали зайняті, чекає звільнення каналу обслуговування, тобто стає в чергу.

Розрізняють одноканальні та багатоканальні СМО з очікуванням, при цьому на довжину черги можуть бути накладені обмеження, що визначають максимальну довжину черги.

Всі показники (характеристики) функціонування СМО можна розділити на чотири групи: імовірнісні; кількісні; тимчасові; якісні.

Імовірнісні показники. Будемо говорити, що вимога знаходиться в СМО, якщо вона очікує обслуговування або знаходиться на обслуговуванні. Позначимо J число вимог, що знаходяться в СМО. Так як з плином часу це число змінюється, то J є випадковою величиною. Припустимо, що вироблено дуже велика кількість спостережень над СМО, в результаті яких встановлено частку випадків, коли в системі спостерігалося рівно J вимог. Ця величина називається ймовірністю p_J того, що в СМО мається J вимог.

Всі ймовірності станів повинні задовольняти умови нормування, згідно з яким сума всіх ймовірностей повинна дорівнювати одиниці, тобто

$$\sum_{j=0}^{m+s} p_j = 1. \tag{1.7}$$

Вимога отримує відмову в тому і тільки в тому випадку, якщо в момент свого вступу до СМО застає зайнятими як всі канали, так і всі місця накопичення. Якщо в системі є *n* апаратів і *m* місць накопичувача, то ймовірність даної події дорівнює

$$P_{отк} = P_{m+n}. \tag{1.8}$$

Ця ймовірність називається ймовірністю відмови.

Імовірність знаходження в СМО не більше і вимог

$$P_{<i} = \sum_{j=0}^{i} p_j. \tag{1.9}$$

Імовірність відсутності черги

$$P_{\text{від. ч}} P_{<n} = \sum_{j=0}^{n} p_j . \tag{1.10}$$

Ймовірність того, що вимогу не доведеться чекати початку обслуговування

$$P_{(\text{Ам}<n)} = 1 - P_{(\text{Ам}-n)} = P_{\text{від.ч}} - P_n . \tag{1.11}$$

Імовірність наявності черги

$$P_{\text{н.оч}} = \sum_{j=m+1}^{m+m} p_j . \tag{1.12}$$

Ймовірність того, що всі обслуговуючі апарати зайняті

$$P_{(\text{Ам}+n)} = \sum p_j = P_{\text{н.оч}} + P_n . \tag{1.13}$$

Ймовірність обслуговування

$$P_{\text{обсл}} = 1 - P_{\text{отк}} . \tag{1.14}$$

Кількісні показники.

Середнє число вільних обслуговуючих апаратів

$$A_c = \sum_{i=0}^{n} (n-i) p_i . \tag{1.15}$$

Середнє число вимог зайнятих обслуговуючих апаратів

$$A_M = \sum_{i=0}^{n} i p_i + n \sum_{i=n+1}^{m+n} p_i = n - A_c . \tag{1.16}$$

Середнє число вимог у накопичувачі

$$A_{\text{Н}} = \sum_{i=s+1}^{m+n} (i-n) p_i . \tag{1.17}$$

Середнє число вимог в обслуговуючій системі

$$A = \sum_{i=0}^{m+n} ip_i = A_{\text{Н}} + A_M. \tag{1.18}$$

Середнє число вимог, які отримують відмову за одиницю часу

$$A_{\text{отк}} = \lambda P_{\text{отк}}. \tag{1.19}$$

Тимчасові показники. Важливим показником є середній час очікування початку обслуговування $T_{\text{оч}}$. Сума середнього часу очікування $T_{\text{оч}}$ і обслуговування $T_{\text{обсл}}$ дорівнює середньому часу перебування вимоги в системі

$$T_{\text{сист}} = T_{\text{оч}} + T_{\text{обсл}}. \tag{1.20}$$

Встановимо корисний зв'язок між середнім числом вимог, що знаходиться в системі, і середнім часом перебування вимоги в системі. Так як за одиницю часу в систему надходить λ вимог, а середня тривалість перебування однієї вимоги в системі є $T_{\text{сист}}$, то сумарна тривалість знаходження всіх вимог у системі за одиницю часу дорівнює ($\lambda T_{\text{сист}}$). Але так як в системі знаходиться в середньому A вимог, то ця величина дорівнює добутку одиниці часу на A, тобто

$$A = \lambda T_{\text{сист}}. \tag{1.21}$$

Аналогічно можна вивести формулу

$$A_{\text{Н}} = \lambda T_{\text{оч}}. \tag{1.22}$$

Якісні показники. Наведені показники характеризують ступінь використання обслуговуючих апаратів і витрати часу на перебування вимог у системі (у черзі і на обслуговування). Зазвичай буває необхідним узгодити ці величини, щоб знайти економічно оптимальне рішення. У зв'язку з цим припустимо, що за одиницю часу перебування вимоги в системі обслуговування мають місце збитки, рівні C. Крім того, експлуатація одного апарату призводить за одиницю часу до витрат,

рівним *К*. Відзначимо, що у величину *К* звичайно входять як експлуатаційні витрати, так і питомі капітальні витрати на один апарат, пов'язані з його придбанням і припадають на одиницю часу. Нехай далі відмова в обслуговуванні однієї вимоги тягне збиток, рівний C_0.

Тоді наведені середні витрати, пов'язані з експлуатацією *n* апаратів, надходженням в систему в одиницю часу λ вимог і збитками від відмов вимогам, в одиницю часу в середньому становлять

$$E = C_0 \lambda P_{отк} + CT_{сист} \lambda + Kn \tag{1.23}$$

У цьому виразі $T_{сист}$ (середній час перебування однієї вимоги в системі), а разом з ним і доданок ($CT_{сист} \lambda$) зменшуються із зростанням числа обслуговуючих апаратів *n*. На противагу цьому доданок *Кn* із зростанням числа апаратів збільшується. Завдання полягає, отже, у виборі такого значення *n*, при якому критерій (Е) буде мінімальний. Такий вибір *n* забезпечує мінімізацію наведених витрат, пов'язаних з придбанням та експлуатацією апаратів і непродуктивним перебуванням вимог у системі обслуговування.

Відзначимо також, що за рахунок додаткових витрат часто є можливість зменшити середню тривалість обслуговування однієї вимоги $T_{обсл}$. При цьому буде зменшуватися і величина $T_{сист}$- середній час перебування вимоги в системі. У цьому випадку для знаходження оптимального рішення в критерій Е слід ввести також зазначені додаткові витрати.

Наступним показником якості функціонування СМО є:

- коефіцієнт завантаження поста:

$$K_з = \lambda \mu / n. \tag{1.24}$$

- коефіцієнт використання апаратів

$$K_{вик} = A_M / n. \tag{1.25}$$

Програма PAXPORT, розроблена як консалтингова фірма Халкроу. Використовувалася для моделювання переміщення людей у аеропортах , на залізничних вокзалах і станціях, у місцях проведення масових спортивних заходів. Вона відображає агреговані дані про потік і рівні організації пішохідного руху у рамках прийомів теорії графів. Варто відзначити, що метод теорії масового обслуговування, як і метод кліткового автомату, не знайшли широкого застосування у комерційних програмних продуктах.

1.5 Багатоагентні моделі

Четвертий вид моделювання руху пішохідних потоків – багато агентні моделі. Для них характерна оцінка взаємодії певних індивідів між собою у заданому середовищі. Ці індивіди у моделі – умовні агенти, які здатні приймати рішення, діяти і отримувати знання із середовища.

Підхід використовується для економічного, соціального, ділового і логістичного моделювання, де індивід постійно взаємодіє з зовнішнім середовищем і іншими індивідами, а також має здатність самостійно приймати важкі рішення. М. Бетти висунув припущення, що багатоагентні моделі з'являються, як серйозна альтернатива більш важким і геометричним підходам до просторового моделювання. Їх поява обумовлена багатьма причинами у тому числі у вигляді підвищення якості маючих даних про існуючі земле використання, різноманітних видах діяльності і потоках пішоходів.

Багатоагентні моделі розглядають пішохода у якості абсолютно незалежного суб'єкта, який має здатність до пізнання і вивченню. Порівняльний огляд різноманітних підходів до комп'ютерного моделювання поведінки мас людей у час евакуації в умовах надзвичайної події представлених у роботах С. Гвайннаі [6] співавтором, а Д.Хелбинг робить акцент на характер таких спостережливих явищ у поведінці людей, як слідування один за одним по коридорам або коливання мас у місцях звужування коридорів («пляшкових

шийок»), при виникненню труднощів і навіть блокування рухів у надзвичайних ситуаціях.

Структура агентно-базованої моделі - система моделюється як набір автономних об’єктів, що приймають рішення, які називають агентами. Кожен агент індивідуально оцінює конкретну ситуацію і приймає рішення на основі встановлених правил. Повторювані взаємодії між агентами є характерною особливістю АМ. Агентна модель складається з системи агентів і відносин між ними. Складніша агентна модель може містити різні навчальні техніки, дозволяючи проводити більш реалістичне вивчення та адаптацію. Розрізняють два основні класи архітектур агента:

• архітектура, яка базується на принципах і методах штучного інтелекту, тобто систем, заснованих на знаннях (deliberative agent architecture, «архітектура розумного агента»);

• архітектура, заснована на поведінці (reactive architecture) або «реактивна архітектура» (заснована на реакції системи на події зовнішнього світу). Наразі серед розроблених архітектур не існує такої, про яку можна було б точно сказати, що вона є виключно поведінковою або заснована тільки на знаннях.

Будь-яка з розроблених архітектур є, по суті, гібридною, і має ті чи інші риси від архітектур обох типів. З іншого боку, незалежно від того, що лежить в основі формалізації парадигми, архітектури агентів класифікуються відносно типу структури, накладеної на функціональні компоненти агента і прийнятих методів організації взаємодії його компонент в процесі роботи. Як правило, архітектура агента організується у вигляді декількох рівнів. Відповідно до роботи, серед багаторівневих архітектур розрізняють горизонтальну організацію взаємодії рівнів і вертикальну.

З класичного погляду архітектура на основі знань - це така архітектура, яка містить символьну модель світу, представлену в явній формі, до того ж в якій прийняття рішень про дії, які повинні бути зроблені агентом, здійснюється на основі міркувань логічного чи псевдологічного типів. Такий агент може розглядатися як спеціальний випадок системи, заснованої на знаннях. Ідея

архітектури агента на основі знань у наш час вже вийшла за межі логічної парадигми подання та обробки знань.

Є архітектури, які сповідують лінгвістичний підхід (на основі формальних граматик), а також такі, що намагаються використовувати наближені знання і правдоподібні міркування. Архітектура на основі планування («плануючий агент») розглядається як альтернатива підходу архітектури, заснованої на знаннях.

У цьому підході планування розглядалося як конструювання послідовності дії, яка, будучи виконаною, призводила б, у результаті, до досягнення бажаної мети. Простим прикладом архітектури такого роду є архітектура, в якій реакція агента на зовнішні події генерується скінченим автоматом. Тільки найпростіші програми агентів можуть бути реалізовані за однорівневою схемою. Як правило, функціональні модулі агента структуруються в кілька рівнів, однак за різними принципами. Зазвичай, рівні представляють різні функціональності: сприйняття зовнішніх подій і прості реакції на них; поведінка, керована цілями; координація поведінки з іншими агентами; оновлення внутрішнього стану агента, тобто переконань про зовнішній світ; прогнозування станів зовнішнього світу; визначення своїх дій на черговому кроці та інше. Найчастіше в архітектурі агента присутні рівні, відповідальні за: сприйняття і виконання дій, реактивну поведінку, локальне планування, кооперативну поведінку, моделювання, формування намірів, навчання агента.

У горизонтально організованій архітектурі всі рівні агента мають доступ до рівня сприйняття і дій (у загальному випадку — всі рівні можуть спілкуватися між собою в стилі «бродкастінг», а у вертикально організованій архітектурі тільки один з рівнів має доступ до рівня сприйняття і дій, а кожний з інших рівнів спілкується тільки з парою безпосередньо суміжних з ним рівнів). Основні проблеми реалізації горизонтально організованої архітектури обумовлені складністю організації узгодженої роботи всіх рівнів. У вертикально організованій архітектурі проблема управління взаємодією рівнів не є настільки складною, оскільки вихідна інформація кожного з рівнів завжди має адресата.

1.6 Аналіз програмних засобів для моделювання руху людей

Програмне рішення від PTV GROUP для вирішення завдань, пов’язаних з мобільністю та транспортуванням [8]. Існують наступні PTV продукти:

− «PTV Vissim» – імітаційне моделювання та візуалізація мультимодальних транспортних потоків з максимальною деталізацією, взаємодія будь-яких видів транспорту в рамках єдиної моделі;.

− «PTV Viswalk» – імітаційне моделювання поведінки пішоходів, дозволяє змоделювати поведінку всіх учасників процесу в існуючій і планованої інфраструктурі на базі науково-обґрунтованих моделей поведінки пішоходів, а також може бути використано для оцінки часу і проаналізу різні сценарії евакуації людей з будівел. [9].

Завдяки інтеграції систем PTV Viswalk і PTV Vissim можна моделювати взаємодію різних учасників процесу в рамках єдиної імітаційної моделі. Скриншот етапу роботи продукту «PTV Vision» наведено на рисунку 1.3.

Рисунок 1.3 - Скріншот етапу роботи продукту «PTV Vision»

«CUBE». Відкрите програмне забезпечення для моделювання від компанії Bentley Systems, розробника програмного забезпечення для професіоналів в сфері будівництва і управління світовою інфраструктурою, включаючи автомобільні дороги, мости, аеропорти, хмарочоси, заводи і електростанції. Основні сфери використання продукту:

− розробка і застосування інтелектуальних, мультимодальних транспортних моделей;

− аналіз впливу нових проектів на транспортну мережу міста, землекористування та населення.

Програмний продукт дозволяє моделювати та визначати яким чином зміни в інфраструктурних об'єктах, їх експлуатації, технологіях і демографії будуть впливати на пересування людей і доступність конкретних районів. Існують наступні розширення та продукти CUBE:

− «CUBE Voyager» – моделювання макроскопічного переміщення людей і транспортних засобів;

− «CUBE Avenue» – мезоскопічне (поєднує макроскопічні методи трафіку із прогресуванням руху та періодами часу, типовими для мікроскопічних моделей) моделювання дорожнього руху;

− «CUBE Cargo» – моделювання вантажних перевезень;

− «CUBE Land» – моделювання землекористування;

− «CUBE Dynasim» – мікроскопічне моделювання дорожнього руху;

− «CUBE Access» – ідентифікація доступності таких важливих місць призначення, як робота, медичні центри, транспортні вузли та розважальні комплекси [10].

Скріншот етапу роботи продукту «Citilabs Cube Dynasim» наведено на рисунку 1.4.

Рисунок 1.4 – Скріншот етапу роботи продукту «CUBE Dynasim»

«Quadstone Paramics». Програмне забезпечення компанії Paramics для імітації мікроскопічного дорожнього руху та симуляції руху пішоходів – використовується фахівцями для проектування ефективної, економічної, зручної для водіїв та пішоходів транспортної інфраструктури, дозволяє виконувати оперативну оцінку поточного та майбутнього стану дорожнього руху та деталізацію високої чіткості [11].

Програмний продукт розроблено на основі кількох моделей, головним чином за статтею «Модель для моделювання дорожнього руху» Ганса-Томаса Фріцше. Paramics використовує модель, що слідує за автомобілем та змінює смугу руху , щоб показати співвідношення числових даних для дорожніх мереж в різних умовах за допомогою комп'ютерної графіки. Модель Paramics представлена комбінацією із вузлів, посилань та інших пов'язаних об'єктів, з метою відтворення обмежень реального життя в геометрії. Після виходу з «зони походження» кожен

транспортний засіб намагається завершити свій шлях до "зони призначення", обмежуючись фізичними та динамічними параметрами транспортного засобу. Використовуючи мікросимуляцію, Paramics дозволяє користувачам моделювати окремі рухи автомобіля, щоб передбачити майбутню поведінку моделі подорожі в результаті зміни обсягу руху або геометричного планування дороги. Кілька планів було розроблено на початку 2000-х років дослідниками з Каліфорнійського університету в Ірваїні для Каліфорнійського департаменту транспорту. Плагіни було розроблено з використанням API, що включали активований сигнал, множину синхронізованих планів синхронізованого сигналу, координацію активованого сигналу, агрегатор даних детектора, контроль вимірювання рампи, контроль перевизначення черги на рампі, контроль вимірювання рампи ALINEA, контроль вимірювання рампи BOTTLENECK, SWARM контроль вимірювання нахилу та МНС автостради, Quadstone Paramics зображена на рисунку 1.5.

Рисунок 1.5 – Скріншот етапу роботи продукту «Quadstone Paramics»

«SIMWALK». Програмний продукт від компанії SAVANNAH-SIMULATIONS успішно застосовуються органами влади та консультантами по всьому світу програмне забезпечення для моделювання переміщення натовпу людей для транспорту, авіації, міського дизайну та архітектури [12]. Скріншот етапу роботи продукту «SIMWALK» зображено на рисунку 1.6.

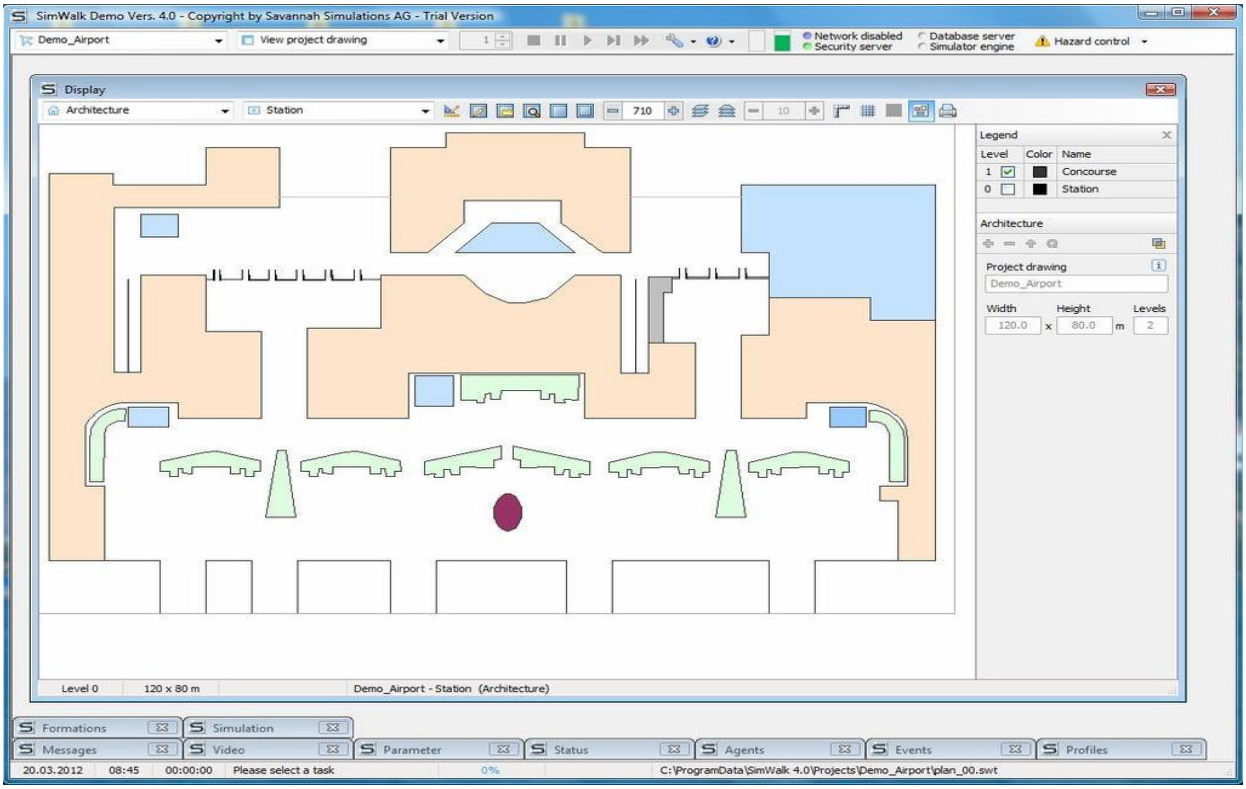


Рисунок 1.6 - Скріншот етапу роботи продукту «SIMWALK»

2 РОЗРОБКА МОДЕЛІ ПОВЕДІНКИ ЛЮДЕЙ В ПРИМІЩЕННІ З ВИКОРИСТАННЯМ КЛІТИННИХ АВТОМАТІВ

2.1 Постановка задачі моделювання руху людей з урахуванням протиепідемічних заходів в умовах пандемії коронавірусной інфекції COVID-19

В умовах пандемії коронавірусной інфекції COVID-19 змінюються шаблони поведінки людей, що обумовлено необхідністю дотримування соціальної дистанції як засобу запобігання зараженню та подальшому розповсюдженню вірусу.

Соціальне дистанціювання – комплекс санітарно-епідеміологічних заходів не медикаментозного характеру, спрямований на зупинку або уповільнення поширення заразної хвороби. Мета соціального дистанціювання – зниження ймовірності контакту між інфікованими і неінфікованими людьми, зменшення рівня передачі інфекції, захворюваності та, врешті, смертності. Соціальне дистанціювання найбільш ефективне, при виникненні інфекції здатної передаватися повітряно-крапельним шляхом (при кашлі або чханні); під час фізичного контакту, непрямого фізичного контакту (наприклад, через дотик до забруднених поверхонь); або при передачі повітряним шляхом (коли мікроорганізм здатний зберегти активність у повітрі протягом тривалого періоду). Соціальне дистанціювання може бути менш ефективним у випадках, коли інфекція передається, здебільшого, водою чи їжею, чи ж переносниками, такими як комахи (комарі), й зрідка — від людини до людини [13]. Тому, у рамках побудови моделі поведінки людей у приміщенні необхідно брати до уваги, що взаємодія людей буде обумовлена необхідністю дотримання соціальної дистанції, та, як наслідок, модель повинна враховувати цей фактор.

Крім того, відповідно до розпорядження МОЗ України № 32 від 18.05.2020 Про проведення громадського обговорення деяких вимог до протиепідемічних

заходів при послабленні карантину [14], кожне підприємство повинно вживати наступні заходи, а саме:

− організувати роботу таким чином, щоб зменшити контакт між особами, забезпечити необхідну відстань між фізичними особами із розрахунку їх робочих місць або запровадити режим віддаленої (дистанційної) роботи;

− забезпечити регулярну дезінфекцію виробничих приміщень;

− підтримувати належну гігієну виробничого середовища та необхідне використання вентиляції в приміщеннях;

− забезпечити необхідними дезинфікуючими засобами (наприклад, мило, дезинфікуючий засіб для рук, схематичних знаків та пам'яток) та заохотити працівників дотримуватись гігієни на робочому місці (наприклад, регулярно застосовувати гігієнічні засоби для рук, уникати дотиків очей/носа/ рота);

− сприяти дотримуватись гігієни дихання (наприклад, забезпечити працівників індивідуальними масками для обличчя та разовими печатками, з метою мінімізувати ризик інфікування, особливо для осіб у групі ризику);

− перегляд правил внутрішнього розпорядку та заходів інфекційного контролю, в тому числі, навчання працівників кращим практикам безпеки та гігієни.

Беручі до уваги все вищезазначене та враховуючи, що модель будується з використанням клітинного автомату, кожен стан якого уявляє певне розміщення людей у приміщенні, виглядає доречним, окрім інформації про переміщення, у якості результатів моделювання отримувати статистичні дані що до зон найчастішого відвідування, як тих, що найшвидше забруднюються, а також зон потенційного порушення дистанції.

2.2 Розробка моделі приміщення

У якості приміщення для моделювання поведінки людей у ньому було обрано торгівельне приміщення, типу магазин. Для виявлення значущих для поведінки людей елементів у даному типу приміщень було проаналізовано

основні шаблони переміщення покупців по магазинах. Загальний прикладом поведінки людей у магазині може бути алгоритм, що описано укрупненою блок схемою та наведено на рисунку 2.1.

Рисунок 2.1 – Укрупнена блок-схема загальної поведінки покупців у приміщенні магазину

З блок-схеми бачимо, що є певний набір умовних об’єктів, відвідування яких неможливо уникнути під час цілеспрямованого відвідування магазину: вхід до магазину, каса та вихід з магазину. Вхід та вихід у переважній більшості випадків відповідають одному й тому же самому об’єкту, тому доречно розглядати їх як єдине ціле. Поміж іншим, відвідування каси може бути уникнено, у разі виникнення екстремальній ситуації, яка передбачає негайне звільнення приміщення з метою збереження людського життя та здоров’я, наприклад, при виникненні пожарної тривоги, загрози вибуху, обвалення будівлі, тощо.

Таким чином, при побудові моделі будемо оперувати наступними об’єктами: приміщення, стелаж, каса, вхід/вихід та покупець та побудуємо моделі

цих об’єктів. Також розглянемо два можливих шаблони поведінки людини у приміщенні магазину: у звичайному та екстремальному режимі.

Модель приміщення уявимо як дискретний простір, фрагментами якого є об’єкти стелаж, каса, вхід/вихід, покупці та вільний простір. Об’єкти стелаж, каса, вхід/вихід не можуть змінювати свого місце розташування у просторі приміщення, об’єкт покупець переміщується по вільному простору поміж інших об’єктів.

Математично, модель приміщення опишемо множиною клітин P, розміщених у вигляді решітки, де фрагменти решітки – клітини $p_{(i,j)}$:

$$p_{(i,j)} \in P. \tag{2.1}$$

Множина клітин P приміщення задається сукупністю множин 5ти типів, які не перетинаються:

$$\begin{gathered} P = P_{io} \cup P_g \cup P_c \cup P_p \cup P_{empty}, \\ P_{io} \cap P_g \cap P_c \cap P_p \cap P_{empty} = \emptyset, \end{gathered} \tag{2.2}$$

де P_{io} – множина клітин входу/виходу;

P_g – множина клітин стелажів;

P_c – множина клітин кас;

P_p – множина клітин покупців;

P_{empty} – множина інших клітин (вільний простір).

Множини P_{io}, P_g та P_c є константними та не змінюють свого вмісту (об’єкти стелажі, каси та вхід/виходи не пересуваються). У множин P_p та P_{empty} можуть змінюватися склад (покупець переміщується) та потужність (покупець з в’явився у магазині, або покупець покинув магазин), однак об’єднання множин є константною множиною (покупець може пересуватися тільки вільним простором, не перетинаючись з іншими об’єктами).

Графічне уявлення моделі приміщення наведено на рисунку 2.2. На рисунку елементи множини P_g позначені чорним кольором (стелажі), елементи множини P_c - червоним (каси), елементи множини P_{io} - зеленим (вхід/виходи), елементи множини P_p – синім (покупці) та елементи множини P_{empty} - білим кольором.

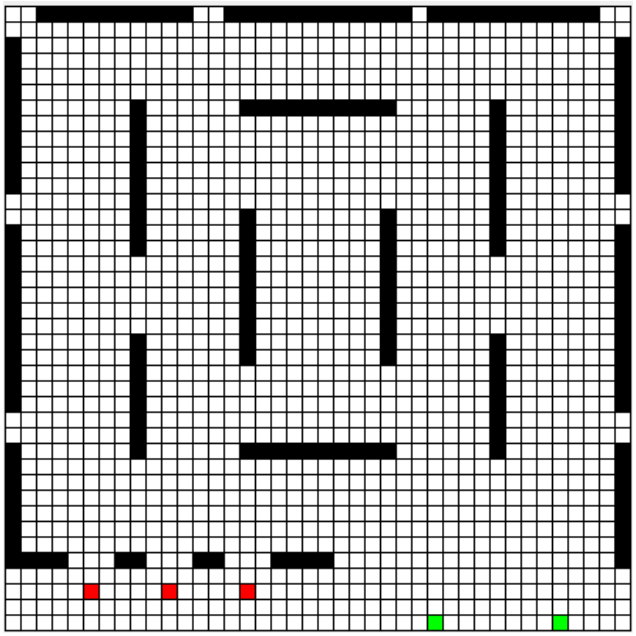
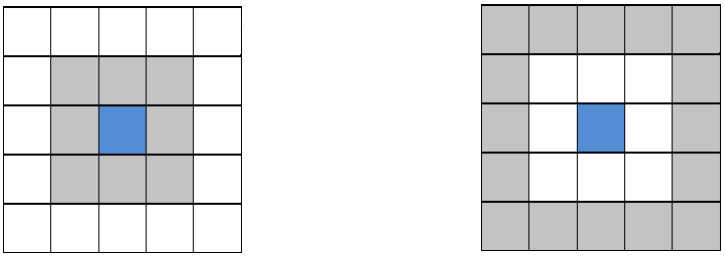


Рисунок 2.2 – Модель приміщення

2.3 Розробка моделі покупця

Модель покупця уявимо множиною властивостей та відповідної поведінки. До властивостей об’єкту віднесемо інформацію щодо поточного місцезнаходження (координати клітини на решітці) та цілі прямування (координати клітини на решітці). Поведінка покупця у загальному розумінні – це переміщення (зміна поточного місцезнаходження) між об’єктами стелажів, кас, вхід/виходів, досягаючи їх (що у реальному світі відповідає входу до магазину, обранню товару, розрахунку на касі та виходу з магазину), та іншими об’єктами покупців. Покупець переміщується постійно, прямуючи від одного до іншого нерухомого об’єкту у певній послідовності, найкоротшим шляхом та відповідно до певних правил, що до близькості інших об’єктів-покупців. Будемо вважати, що

покупець може переміщатися тільки у сусідні клітини решітки околиці Мура 1 від поточного розміщення покупця, а також, що покупець може бачити клітини околиці Мура 2. На рисунку 2.3 наведено околицю переміщення покупця (а) та околицю, яку він може бачити (б). Множина клітин решітки куди може бути переміщено (за певними умовами) покупця на рисунку 2.3 (а) виокремлено сірим, множина клітин решітки, які оглядаються на предмет наявності інших покупців виокремлено сірим на рисунку 2.3 (б).



а) б)

Рисунок 2.3 – Множина клітин решітки переміщення покупця (а) та множина клітин бачення інших покупців (б).

Опишемо алгоритм переміщення покупця.

Покупець переміщується у вільну область приміщення (у клітину решітці околиці 1 покупця) у найкоротшому напрямку до поточної цілі прямування. Якщо у найкоротшому напрямку немає вільної області, або у певній близькості (клітини решітці околиці 2 покупця) є інший об’єкт-покупець, покупець переміщується у будь-яку вільну область навколо нього. Після досягнення цілі покупець обирає наступну для відвідування ціль. Наступна ціль для відвідування обирається випадковим чином, але після відвідування цілі типу каса, наступною ціллю може бути обрано або стелаж, або вихід. Після досягнення цілі, яка є виходом, покупець завершує пересування.

На рисунку 2.4 наведено приклад розміщення об’єктів та доступних областей для переміщення об’єкта. На рисунку чорним позначено ціль

переміщення, синім об’єкти-покупці, центральний синій об’єкт – покупець, що розглядається, рожевим – області, заборонені для переміщення, через близькість інших об’єктів-покупців, білим – області доступні до переміщення, зеленим – наступне місце розташування покупця, як найкоротший шлях до цілі серед доступних.

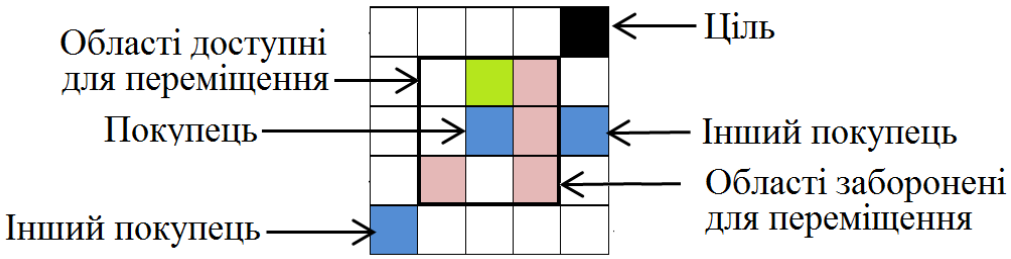


Рисунок 2.4 – Приклад розміщення об’єктів та доступних областей для переміщення об’єкта покупця

У разі екстремальної ситуації покупець обирає у якості цілі об’єкт вихід та переміщується до нього за звичайним правилом переміщення, але за винятком виключення області, поблизу з іншим об’єктом-покупцем.

Математично, модель покупця C буде описуватися 3мя складовими наступним чином:

$$C = \{p_{cur}, p_{purpose}, f\}, \quad (2.3)$$

де p_{cur} – поточне місцезнаходження, $p_{cur} \in P_p$;

$p_{purpose}$ – місцезнаходження цілі, $p_{purpose} \in P_{io} \cup P_g \cup P_c$;

f – правила зміни місцезнаходження.

Опишемо математично правила зміни місцезнаходження:

1) правило переміщення у звичайному режимі:

$$p_{next}(i_{next}, j_{next}) \in random\{(k,l)| \begin{cases} |i_{cur}-k|+|j_{cur}-l| \le 1; \\ (k,l) \ne (i_{next}, j_{next}); \\ (k,l) \in P_{empty}; \\ distance\left((k,l),(i_{purpose}, j_{purpose})\right) \to min; \\ \begin{cases} |k-m|+|l-n| \le 1; \\ (m,n) \notin P_p. \end{cases} \end{cases}\}, \quad (2.4)$$

де: $p_{next}(i_{next}, j_{next})$ – наступне місцезнаходження об'єкту;

$p_{purpose}(i_{purpose}, j_{purpose})$ – місцезнаходження цілі;

2) правило переміщення у екстремальному режимі:

$$p_{nextEx}(i_{next}, j_{next}) \in random\{(k,l)| \begin{cases} |i_{cur}-k|+|j_{cur}-l| \le 1; \\ (k,l) \ne (i_{next}, j_{next}); \\ (k,l) \in P_{empty}; \\ distance\left((k,l),(i_{purpose}, j_{purpose})\right) \to min \end{cases}\};(2.5)$$

3) правило обрання цілі:

$$p_{purpose_{next}}(i_{purpose_{next}}, j_{purpose_{next}}) \in$$
$$\{(k,l)| \begin{cases} (k,l) \in P_{io} \cup P_g \cup P_c, якщо\ p_{purpose}(i_{purpose}, j_{purpose}) \in P_g; \\ (k,l) \in P_{io} \cup P_g, якщо\ якщо\ p_{purpose}(i_{purpose}, j_{purpose}) \in P_c. \end{cases}\}; \quad (2.6)$$

4) правило закінчення переміщення:

$$p_{next}(i_{next}, j_{next}) \in P_{io}. \quad (2.7)$$

Функцію, що знаходить наступну точку місцезнаходження для об'єкта покупець (його переміщення) відповідно до висунутих правил у звичайному режимі назвемо функцією поведінки та позначимо, як $fp_{next}(ip_{cur}, jp_{cur})$. Виконання функції передбачає існування об'єкту, при створені об'єкту визначаються його поточне місце розташування та ціль.

Наведемо формалізований опис алгоритму функції поведінки об'єкту «покупець».

1. Переглянути околицю $N_M^1(icur, jcur)$ точки поточного місцезнаходження об’єкта p_{cur}.

1.1 Якщо серед точок околиці $N_M^1(icur, jcur)$є точка цілі $p_{purpose}$ та поточною ціллю є точка виходу $p_{purpose} \in P_{io}$, то завершити роботу.

1.2 Якщо серед точок околиці $N_M^1(icur, jcur)$є точка цілі $p_{purpose}$, то встановити нову ціль.

2. Серед точок околиці $N_M^1(icur, jcur)$ точки поточного місцезнаходження визначити перелік вільних точок L_{free}.

3. Якщо перелік вільних точок L_{free} не пустий, то перейти до шагу 4, інакше до шагу 6.

4. Серед переліку вільних точек L_{free} визначити точку p_{min} з мінімальною відстанню до точки цілі $p_{purpose}$.

5. Переглянути околиці точки p_{min} $N_M^1(ipmin, jpmin)$.

5.1. Якщо в околиці точки є інший об’єкт-покупець, то видалити точку p_{min} з переліку вільних точок та перейти на шаг 3, інакше обрати знайдену точку наступною для переміщення $p_{next} = p_{min}$.

6. Наступною точкою для переміщення встановити поточну $p_{next} = p_{cur}$.

Також введемо функцію поведінки об’єкту у екстремальному режимі, яка буде знаходити наступну точку місцезнаходження для об’єкта покупець з урахуванням правилу переміщення у екстремальному режимі. Позначмо функцію як $fp_{nextEx}(ip_{cur}, jp_{cur})$, та наведемо формалізований опис її алгоритму.

1. Обрати у якості точки цілі $p_{purpose}$ будь яку точку виходу.

2. Переглянути околицю $N_M^1(icur, jcur)$ точки поточного місцезнаходження об’єкта p_{cur}.

1.1 Якщо серед точок околиці $N_M^1(icur, jcur)$є точка цілі $p_{purpose}$ та поточною ціллю є точка виходу $p_{purpose} \in P_{io}$, то завершити роботу.

2. Серед точок околиці $N_M^1(icur, jcur)$ точки поточного місцезнаходження визначити перелік вільних точок L_{free}.

3. Якщо перелік вільних точок L_{free} не пустий, то перейти до шагу 4, інакше до шагу 5.

4. Серед переліку вільних точок L_{free} визначити точку p_{min} з мінімальною відстанню до точкі цілі $p_{purpose}$.

5. Встановити наступною точкою для переміщення p_{min}: $p_{next} = p_{cur}$.

2.4 Математична модель клітинного автомату

Модель поведінки людей у приміщенні побудуємо за принципами функціонування двовимірного клітинного автомату без пам'яті. Визначимо складові клітинного автомату: метрику поля, стан клітинок, кількість сусідів, що мають впливів та правила роботи автомата.

Простір клітинного автомату задається у вигляді прямокутної решітки розміром MxN . Решітка є однорідною, кожна клітина у певний момент часу може мати лише один активний стан, зміна стану клітин відбувається одночасно після обчислення нового стану кожної клітини решітки.

Клітиною решітки є кінцевий автоматів A з множиною станів $S = \{s_1,s_2,s_3,s_4,s_5\}$, яка також є множиною початкових та кінцевих станів, та множиною можливих переходів: $\{ s_1->s_1,\ s_2->s_2,\ s_3->s_3,\ s_4->s_5,\ s_5->s_4,\ s_4->s_4\}$.

Околиця клітинного автомату – безліч клітин, що становлять сусідство, визначається околицею Мура порядку 2:

$$N_M^1(i,j) = \{(k,l)||i-k| \le 2, |j-l| \le 2\}. \quad (2.8)$$

Правила роботи клітинного автомату базуються на алгоритмі поведінки об'єкту покупця, тобто функції поведінки моделі покупця. Переходи між активними станами кінцевого автомату виконуються наступним чином:

$$s1 \to s1: \forall;$$

$$s2 \to s2: \forall;$$

$$s3 \rightarrow s3: \forall;$$

$$s4 \rightarrow s5: \exists\,(\mathrm{m},\mathrm{n}): \begin{cases} (\mathrm{m},\mathrm{n}) \in \mathrm{N}_{\mathrm{M}}^{1}(\mathrm{i},\mathrm{j}) \\ \mathrm{fp}_{\mathrm{next}}(\mathrm{m},\mathrm{n}) = (\mathrm{i},\mathrm{j}) \end{cases}; \tag{2.9}$$

$$s4 \rightarrow s4: \nexists\,(m,n): \begin{cases} (m,n) \in N_M^1(i,j) \\ fp_{next}(m,n) = (i,j) \end{cases};$$

$$s5 \rightarrow s4: \exists\,(m,n): \begin{cases} (m,n) \in N_M^1(i,j) \\ fp_{next}(i,j) = (m,n) \end{cases}$$

де $fp_{next}(i,j)$ – функція поведінки, що визначення наступне місцезнаходження об’єкту у моделі покупця, у екстремальному режимі замість функції $fp_{next}(i,j)$ будемо використовувати функцію $fp_{next\mathrm{Ex}}(i,j)$.

2.5 Програмна модель

Для опису програмної моделі визначимо вхідні та вихідні дані, структури даних та алгоритм проведення моделювання.

Вхідні дані та обмеження.

1. Множина осередків решітки простору приміщення – цілочисельна прямокутна матриця розміром MxN, де M, N > 0. Значення елементів матриці відповідають розміщенню об’єктів у приміщенні та можуть мати такі значення:

- 0 – вільний простір
- 1 – стелаж;
- 2 – вхід/вихід
- 3 – покупець;
- 4 – каса.

Обмеження: на початку моделювання відсутні елементи матриці зі значенням 3, моделювання починається з відсутності покупців у приміщенні.

Дані матриці зберігають у окремому текстовому файлі цілих чисел, у якому перші два числа є розмірами матриці, інші – позначками розміщення об’єктів у приміщенні. Фрагмент вхідного файлу наведено на рисунку 2.5.

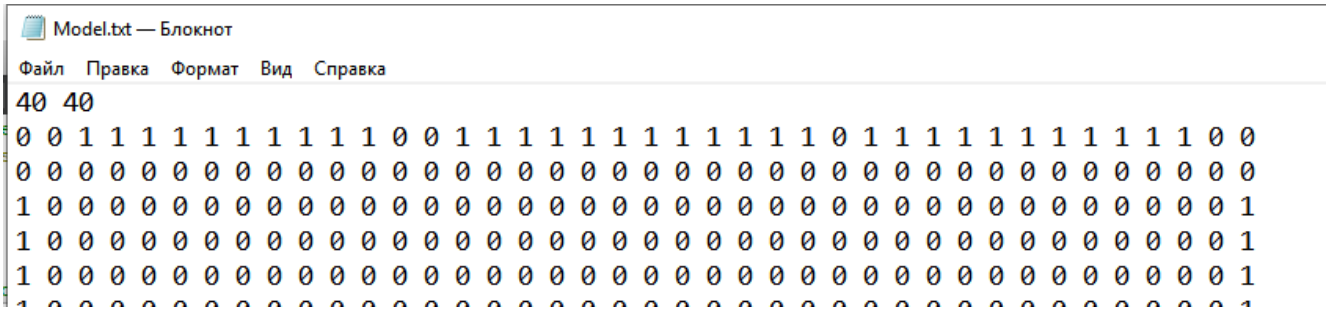


Рисунок 2.5 – Фрагмент вхідного файлу опису приміщення

2. Параметр моделі Pc – не ціле число, $Pc \in [0;1]$, імовірність появи покупця на поточному кроці моделювання.

3. Параметр моделі Pi – не ціле число, $Pi \in [0;1]$, імовірність, що покупець після відвідування каси обере наступною ціллю відвідування вихід (покине систему).

4. Параметр моделі Pd – не ціле число, $Pi \in [0;1]$, імовірність, що на поточному кроці покупець знехтує принципом соціального дистанціювання та обере для переміщення околицю іншого покупця.

5. Параметр моделі T – ціле число, $T>0$, час моделювання (кількість кроків), де один крок – одна зміна стану приміщення (одна зміна стану клітинного автомату).

Вихідні дані та обмеження.

1. Поточний час моделювання Tc – ціле число, $Tc>0$, кількість здійснених кроків (кількість виконаних змін стану клітинного автомату).

2. Загальна кількість покупців Na – ціле число, $Na >0$, кількість покупців, що зайшла до магазину за поточний час моделювання (загальна кількість покупців, згенерованих моделлю).

3. Поточна кількість покупців Nc – ціле число, Nc >0. кількість покупців, що знаходяться у магазині на поточний час (кількість покупців присутніх у системі).

4. Кількість випадків порушення дистанції Nd – ціле число, Nd >0, кількість випадків, коли в околиці 1 одного покупця знаходився інший покупець.

5. Час евакуації Tex – ціле число, Tex>0, час (кількість кроків моделювання) з початку екстреної ситуації до моменту, коли усі покупці покинули приміщення.

6. Dr – матриця забруднення, цілочисельна прямокутна матриця розміром MxN, елементи матриці – кількість станів системи, корда у відповідній клітині простору знаходився покупець.

7. Di – матриця порушення дистанції, цілочисельна прямокутна матриця розміром MxN, елементи матриці – кількість станів системи, коли відповідна клітина простору, будучи у стані 3 (покупець) мала в околиці 1 клітину зі статусом 3 (два покупця знаходяться на сусідніх клітинах простору).

8. Вихідний файл формату .csv, містить результати моделювання: підсумкові статистичні дані, що обуло зібрано, покрокове логування даних на кожному кроці моделювання, підсумкові значення матриць забруднення та порушення дистанції.. Фрагмент вихідного файлу наведено на рисунку 2.6.

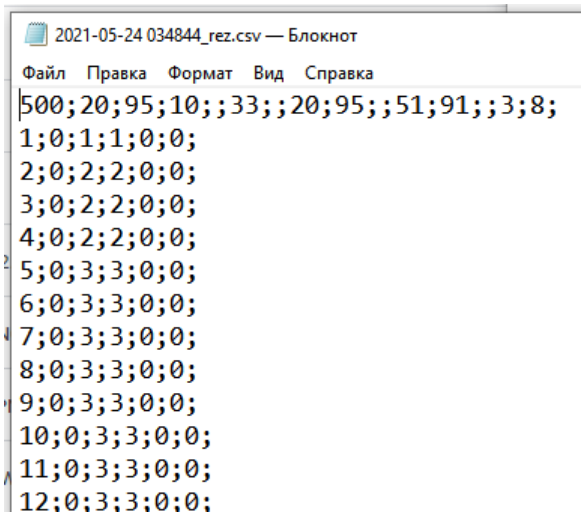


Рисунок 2.6 – Фрагмент вихідного файлу даних

Структури даних.

1. Стан приміщення зберігається у векторі цілих значень:

vector <int> _data.

2. Матриця забруднення зберігається у векторі цілих значень:

vector <int> _mud.

3. Матриця порушення дистанції зберігається у векторі цілих значень:

vector <int> _col.

4. Клітина простору описується структурою Point, де posX – координата клітини за шириною, posY – координата клітини за висотою:

```
structure Point
{   int posX;
    int posY;
    };
```

5. Покупець описується структурою Persone, де pos – клітина простору, де знаходиться покупець, goalPoint – клітина простору де знаходиться поточна ціль:

```
structure Persone
{
    Point pos;
    Point goalPoint;
};
```

6. Множина покупців, що знаходяться на поточний момент у приміщенні зберігається у переліку типу Persone:

vector <Persone> _persones.

Опишемо алгоритм за яким виконується програмне моделювання поведінки людей у приміщенні.

На початку моделювання завантажується матриця приміщення, встановлюються параметри моделювання та виконується запуск моделювання. Моделювання проводиться у двох режимах: у режиму моделювання звичайної поведінки людей та режиму моделювання поведінки людей у екстреній ситуації – при евакуації.

Моделювання розпочинається у режимі звичайної поведінки людей. Проходить перевірки поточного режиму моделювання та, якщо поточний режим – звичайна поведінка, виконується генерація покупців, відповідно то імовірності їх появи. Якщо покупець генерується об’єкт покупця додається до переліку покупців. Для кожного покупця з переліку виконується обчислення та знаходиться наступна клітина приміщення куди слід його перемістити. Пошук клітини виконується відповідно до функції fpnext, що описує правило переміщення об’єкта покупця (см.п.2.3). Якщо наступною клітиною для покупця є клітина виходу – об’єкт покупця видаляється з переліку покупців (покупець покидає приміщення). По закінченню обробки покупців виконується «пересування», обчислюються матриці бруду та порушення дистанції, виконується візуалізація кроку моделювання та виконується перехід до наступного кроку.

Закінчення заданого часу моделювання, умовно, є моментом виникнення екстремальної ситуації. По закінченні встановленого часу моделювання, система переходить у режим моделювання поведінки людей при евакуації з приміщення. Для кожного покупця наступними цілями встановлюються клітини виходу. Якщо перелік покупців у приміщенні не пустий, для кожного покупця з переліку виконується обчислення та знаходиться наступна клітина приміщення куди слід його перемістити. Пошук клітини виконується відповідно до функції fpnextEx, що описує правило переміщення об’єкта покупця при евакуації (см.п.2.3). По закінченню обробки покупців виконується «пересування», обчислюються матриці бруду та порушення дистанції, виконується візуалізація кроку моделювання та виконується перехід до наступного кроку моделювання у режимі евакуації.

Моделювання режиму евакуації закінчується, коли перелік покупців у приміщенні стає пустим – усі покупці покинули приміщення.

Узагальнений алгоритм проведення моделювання наведено на рисунку 2.5.

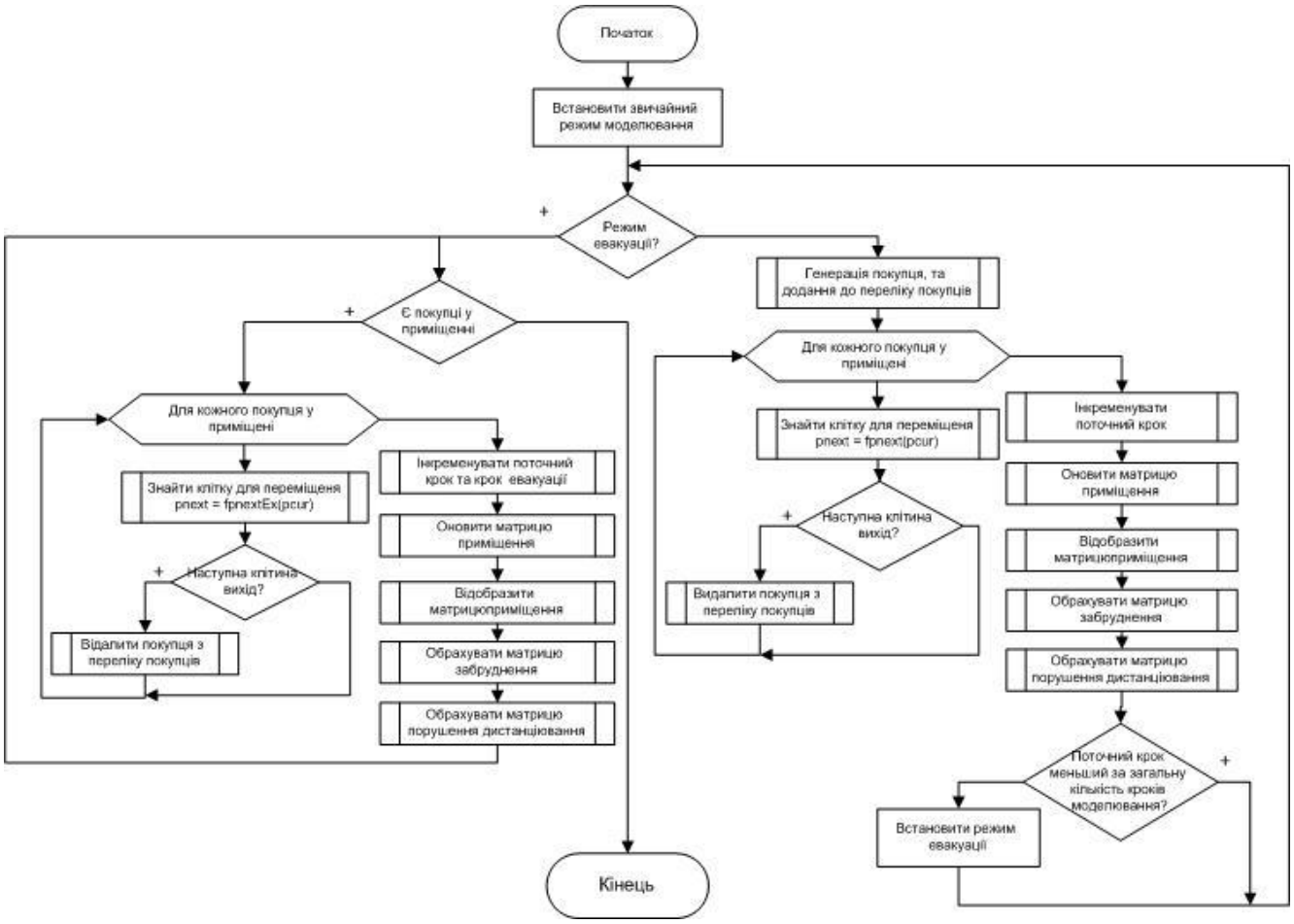


Рисунок 2.7 – Узагальнений алгоритм проведення моделювання

2.6 Програмна реалізація моделі поведінки людей

Програмний продукт було розроблено із використанням ноутбуку HP 250 G3 з наступними характеристиками:

– процесор Intel Pentium N3530 (2.16 - 2.58 ГГц);

– ОЗП 4.00 ГБ (DDR3L 1600 МГц);

– графічний адаптер Intel HD Graphics (інтегрований).

Продукт було розроблено під операційною системою Windows 10, він функціонує на персональних комп’ютерах з операційною системою Microsoft Windows 10 та вище.

2.6.1 Опис використаних програмних засобів

Програмну реалізації розробленої моделі було виконано у середовищі розробки QT Creator мовою програмування C++.

Qt Creator – це кросплатформене інтегроване середовище розробки (IDE) для розробки застосунків з використанням бібліотеки Qt. Qt Creator працює на операційних системах Windows, Linux та macOS для настільних ПК і дозволяє розробникам створювати додатки на настільних, мобільних та вбудованих платформах. Розширений редактор коду Qt Creator дозволяє кодувати мовами програмування C ++, QML, JavaScript, Python та ін., пропонує сервіси завершення коду, виділення синтаксису, рефакторинг та має вбудовану документацію. Qt Creator інтегрується з найпопулярнішими системами контролю версій, включаючи Git, Subversion, Perforce та Mercurial, його оснащено вбудованими візуальними редакторами для створення додатків на основі віджетів C ++ або швидких анімованих інтерфейсів на основі Qt Quick із готовими елементами управління.[15].

C++ - це сучасна потужна мова програмування загального призначення та на теперішній час є однією з найпопулярніших мов програмування у світі, мова програмування високого рівня, яка підтримує різні парадигми програмування, такі як процедурне, об'єктно-орієнтоване, або функціональне. C ++ надає програмам чітку структуру і дозволяє повторно використовувати код, знижуючи витрати на розробку. Мова є портативною і може використовуватися для розробки додатків, що надалі можна адаптувати до декількох платформ. Мовою C ++ розробляють операційні системи, драйвера, браузери, потужні серверні та клієнтські додатки, ігри тощо. C ++ Мову розроблено Б'ярном Страуструпом в AT&T Bell Laboratories у 1979 році на базі мови програмування C з початково назвою «C з класами», а згодом перейменовано розробником у C++. Вперше стандартизована стандартом ISO/IEC 14882:1998, сучасним та найбільш актуальним є стандарт ISO/IEC 14882:2020. C++ зробила суттєвий вплив на інші мови програмування, в першу чергу на такі популярні сьогодні мови програмування як C# та Java. Однак мові властиві і деякі недоліки, серед яких:

використання покажчиків, які є відносно важким поняттям та споживають багато пам’яті, відсутність збирача сміття, відчутність підтримки вбудованих потоків. Тим не менш, наявність недоліків не впливає суттєво на рівень використання мови при розробці сучасних застосунків.

2.6.2 Опис структури проекту

Структуру проекту «People» наведено на рисунку 2.8.

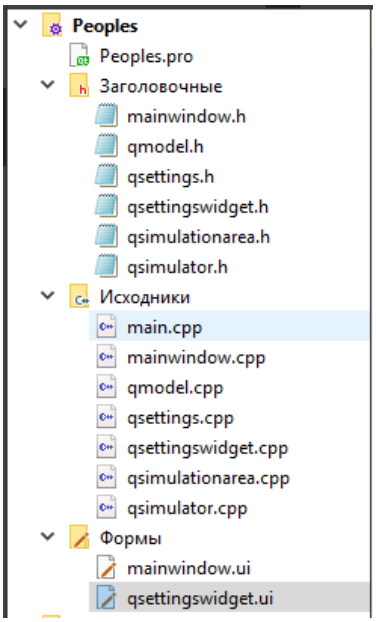


Рисунок 2.6 – Структура проекту People

Структура проекту складається з наступних файлів папок:

- people.pro – файл проекту, містить всю необхідну інформацію для того зборки додатоку;
- тека «Заголовки» – файли заголовків, які містять опис класів;
- тека «Вихідники» – файли, які містять опис класів;
- тека «Форми» – файли Qt Designer, які містять опис форм у вигляді дерева віджетів форми в форматі XML;
- main.cpp – точка входу у програму, виконує створення додатку та запуск головного вікна;

− mainwindow.h – файл опису класу MainWindow (головного вікна проекту);

− qmodel.h – файл опису класу QModel (клас моделі);

− qsettings.h – файл опису класу QSettings (клас налаштувань, проміжний для доступа до полів віджетів);

− qsettingswidget.h – файл опису класу QSettingsWidget (клас віджетів);

− qsimulationarea.h – файл опису класу QSimulationArea (клас візуалізації області моделювання);

− qsimulator.h – файл опису класу QSimulator, структур Point та Person (клас, що забезпечує виконання симуляції);

− mainwindow.cpp – файл реалізація класу MainWindow (головного вікна проекту);

− qmodel.cpp – – файл реалізація класу QModel (клас моделі);

− qsettings.cpp — файл реалізація класу QSettings (клас налаштувань, проміжний для доступа до полів віджетів);

− qsettingswidget.cpp — файл реалізація класу QSettingsWidget (клас віджетів);

− qsimulationarea.cpp — файл реалізація класу QSimulationArea (клас візуалізації області моделювання);

− qsimulator.cpp — файл реалізація класу QSimulator (клас, що забезпечує виконання симуляції);

− mainwindow.ui – опис форми області моделювання у вигляді дерева віджетів форми в форматі XML;

− qsettingswidget.ui – опис форм параметрів та результатів моделювання у вигляді дерева віджетів форми в форматі XML.

2.6.3 Опис інтерфейсу

Інтерфейс програмного продукту уявляє собою вікно, що складається з двох областей:

− область відображення решітки моделі приміщення;

– область налаштувань параметрів моделювання.

Область відображення решітки надає уявлення про поточний стан матриці приміщення, відображаючи значення елементів матриці відповідними кольорами. Додатково область може відображати стан матриці забруднення або матриці порушення дистанції, нормалізованим значенням кольору у відповідних клітках.

Область налаштувань оснащено елементами керування для введення параметрів моделювання та відображення його результатів, а також кнопками для запуску, призупинення процесу моделювання, та кнопкою перемикання процесу моделювання з моделювання звичайної поведінки на моделювання поведінки людей у екстремальній ситуації.

На рисунках 2.9-2.13 наведено скріншоти вікна програми у процесі моделювання.

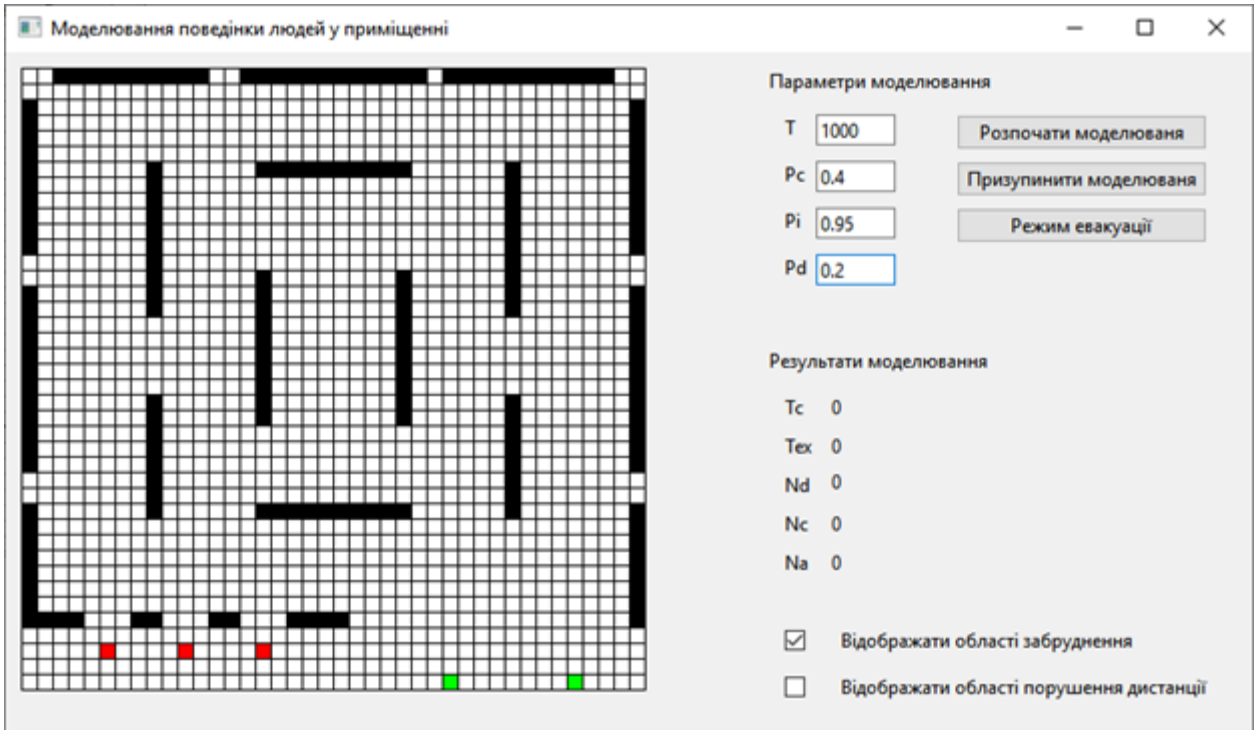


Рисунок 2.9 – Вікно програми перед початком моделювання

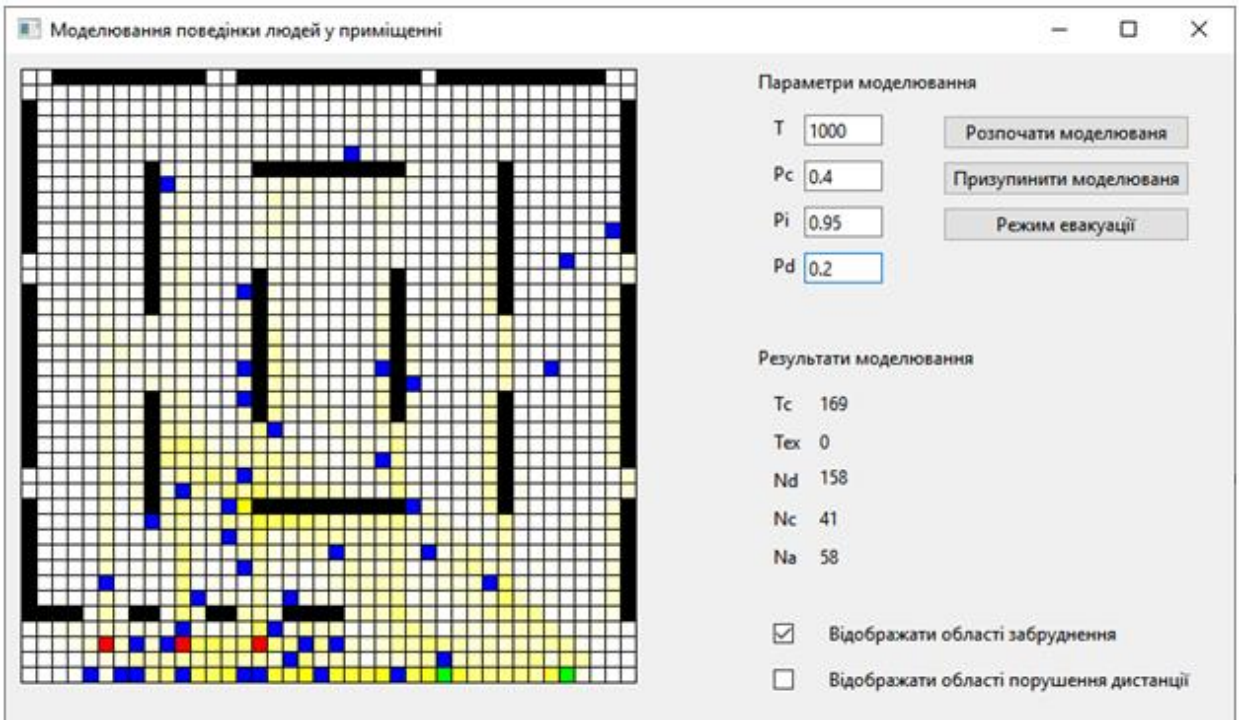


Рисунок 2.10 – Вікно програми у процесі моделювання, з відображення областей забруднення

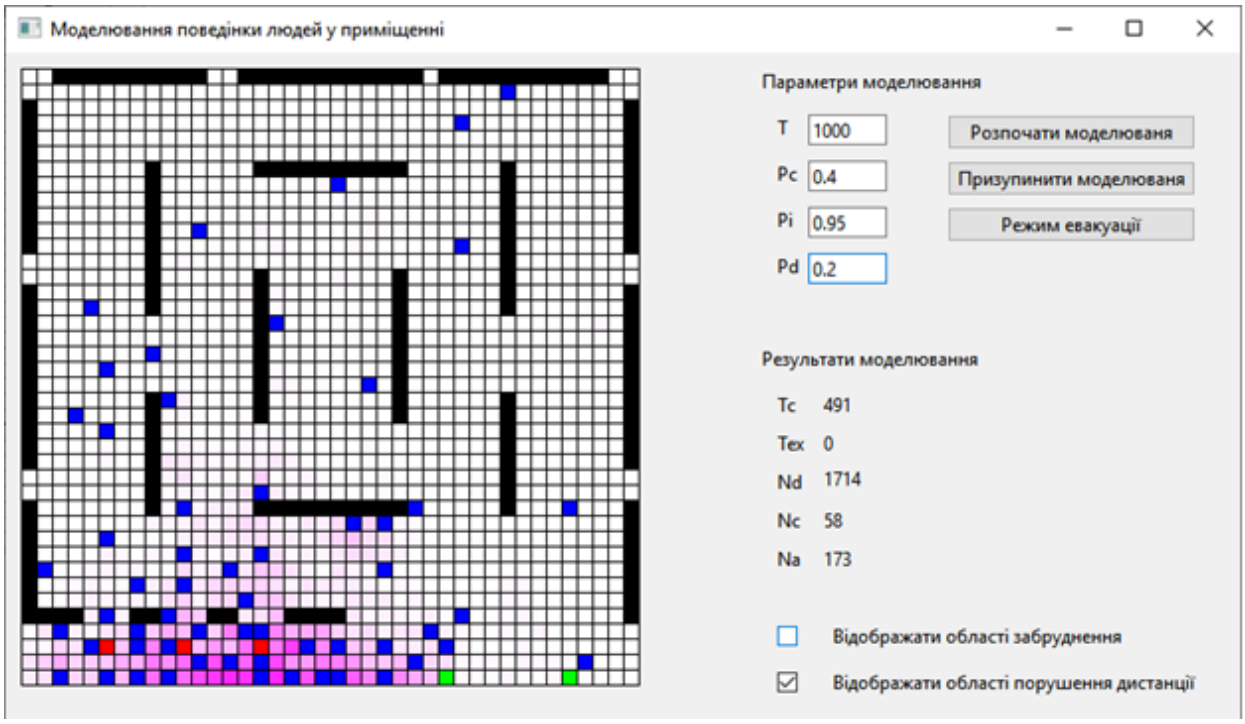


Рисунок 2.11 – Вікно програми у процесі моделювання, з відображенням областей порушення дистанціювання

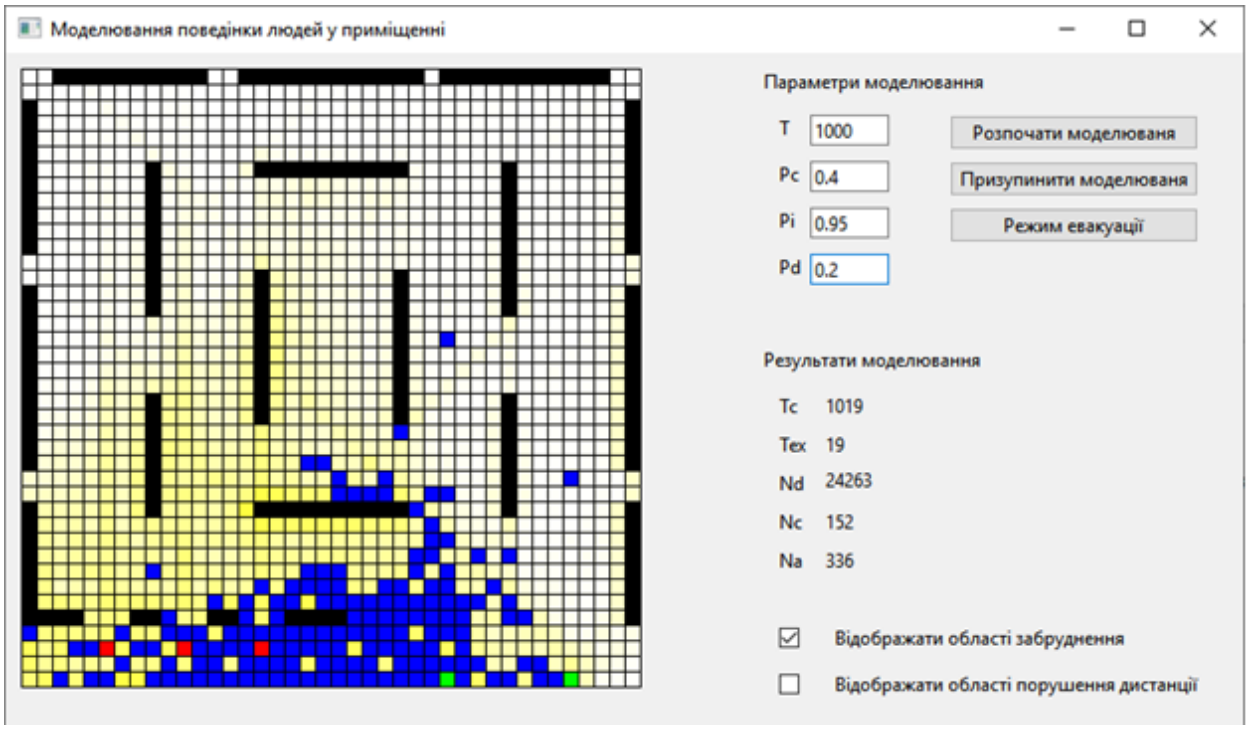


Рисунок 2.12 – Вікно програми у процесі моделювання в режимі евакуації

Рисунок 2.13 – Вікно програми після закінчення моделювання

2.6.4 Опис розроблених класів

При розробці програмного продукту було спроектовано множину класів, діаграму яких наведено на рисунку 2.14.

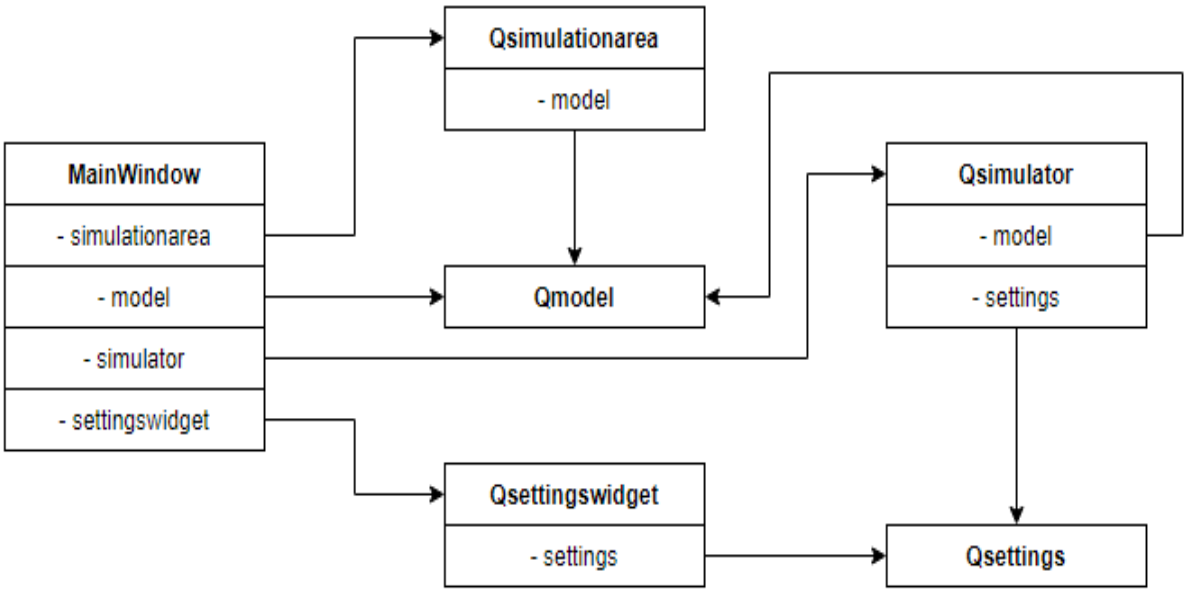


Рисунок 3.14 – Діаграма классів

Наведемо опис розроблених класів.

Клас MainWindow описує сутність «додаток» є успадкованим від класу QMainWindow.

Методи класу:

– public MainWindow(QWidget *parent = nullptr) – конструкток класу, параметр – покажчик на головний відмет;

– public ~MainWindow() – деструктор класу;

– private void onPause() – обробник натиснення кнопки «Призупинити моделювання», запускає на виконання метод pause();

– private void onStart()– обробник натиснення кнопки «Розпочати моделювання», запускає на виконання метод start();

– private void onAlarm() – обробник натиснення кнопки «Режим евакуації», переводить симуляцію у режим евакуації;

– private void stop() – виконує зупинку симуляції, зупиняючи таймер;

– private void start() – виконує запуск симуляції, запускаючи таймер;

– void onTimer() – слот – подія таймера, метод, що виконується при тику таймеру, викликає перерахунок стану області моделювання.

Властивості класу:

– private: Ui::MainWindow *ui – показчик на опис вікна головного вікна;

– private: QTimer* _timer – показчик на об’єкт таймеру;

– private: QSimulationArea* _simulationArea – показчик на екземпляр об’єкту класу QSimulationArea візуалізації області моделювання;

– private: QSettingsWidget* _settingsWidget – показчик на екземпляр об’єкту класу QSettingsWidget віджетів області параметрів та результатів;

– private: QModel* _model - показчик на екземпляр об’єкту класу QModel моделі;

– private: QSimulator* _simulator - показчик на екземпляр об’єкту класу QSimulator логіки поведінки автомату.

Клас QModel описує сутність «область моделювання» або модель «приміщення».

Методи класу:

– public: QModel() – конструктор класу;

– public: virtual ~QModel() – деструктор класу;

– public: void LoadFromFile() – метод завантаження значень клітин решітки моделі приміщення з зовнішнього файлу;

– public: int GetCellStatus(size_t x, size_t y) – метод отримання значення стану клітини решітки моделі приміщення;

– public: int GetMudCellNormalizedValue(size_t x, size_t y) – метод отримання нормалізованого значення рівня забруднення клітини решітки матриці забруднення для формування яскравості відображення клітин решітки моделі приміщення з забрудненням;

– public: int GetBreakCellNormalizedValue(size_t x, size_t y)) – метод отримання нормалізованого значення частоти порушення дистанції у клітині з матриці частот порушення дистанції для формування яскравості відображення клітин решітки моделі приміщення з випадками порушення дистанції між покупцями;

– public: size_t GetWidth() const { return _w; } – метод отримання ширини решітки моделі приміщення;

– public: size_t GetHeight() const { return _h; }– метод отримання висоти решітки моделі приміщення;

– public: void SetStatus(size_t x, size_t y, int status) – метод встановлення значення стану клітини та збільшення значення у відповідної клітини матриці забруднення, параметри: size_t x – координата клітини за шириною, size_t y – координата клітини за висотою, int status – значення стану клітини;

– public: void SetBreakDistance(size_t x, size_t y) – метод збільшення значення у клітиня матриці порушення дистанції, параметри: size_t x - координата координата клітини за шириною, size_t y - координата клітини за висотою.

Властивості класу:

– private: std::vector<std::vector<int>> _data – матриця решітки моделі приміщення;

– private: size_t _w – розмір матриці решітки моделі приміщення за шириною;

– private: size_t _h – розмір матриці решітки моделі приміщення за висотою;

– private: int _maxMudValue = 0 – максимальне значення рівня забруднення у матриці забруднення

– private: int _maxBreakValue = 0 0 – максимальне значення кількості порушень дистанції у матриці порушення дистанції;

– private: std::vector<std::vector<int>> _mud – матриця забруднення;

– private: std::vector<std::vector<int>> _col – матриця порушення дистанції.

Клас QSettings класс налаштувань, проміжний клас для доступа до полів віджета параметрів та результатів моделювання.

Методи класу:

– QSettings() – конструктор класу;

– int GetBrealSocialDistanceValue() – метод отримання значення частоти випадків порушення дистанції;

– int GetGoToExitValue() – метод отримання значення частоти обрання у якості цілі клітину з множини вході/виходів;

– int GetNewPersoneValue()– метод отримання значення частоти додання об’єкту покупця до моделі автомату;

– int GetModellingTimeValue()– метод отримання значення кількості кроків моделювання;

– void SetBreakSocialDistanceValue(int val) - метод встановлення значення частоти порушення дистанції, параметр int val – значення частоти;

– void SetGoToExitValue(int val) – метод встановлення значення частоти обрання у якості цілі клітину з множини вході/виходів, параметр int val – значення частоти;

– void SetNewPersoneValue(int val) – метод встановлення значення частоти додання об’єкту покупця до моделі автомату, параметр int val – значення частоти;

– void SetModellingTimeValue(int val) – метод отримання значення частоти обання у якості цілі клітину з множини вході/виходів;

Властивості класу:

– int _breakSocialDistanceValue = 0.2 – частота порушення дистанції;

– int _goToExitValue = 0.4 – імовірність обрання виходу наступною ціллю;

– int _newPersoneValue = 0.5 – імовірність появи об’єкту покупця у моделі;

– int _modellingTime = 100 – час (кількість кроків)моделювання.

Класс QSimulator поведінка автомату, виконує розрахунок зміну станів клітин решітки моделі приміщення.

Методи класу:

– public: QSimulator(QModel* model, const QSettings* settings) – конструктор класу, параметри QModel* model – модель автомату, const QSettings* settings – параметри моделі;

– public: void next() – метод виконання наступного шагу моделювання, обрахунок нових станів клітин решітки модели приміщення;

– public: void alarm() – метод встановлення параметрів для виконання моделювання у режим евакуації;

– public: void pause() – метод призупинки виконання моделювання;

– private: int getRandom(int min, int max) const; - метод отримання випадкового значення в діапазоні [min, max];

– private: const Point getRandomStall() const – метод отримання випадковим чином клітини решітки з множини клітин стелажів;

– private: const Point getRandomExitDoor() const - метод отримання випадковим чином клітини решітки з множини клітин виходів;

– private: const Point getRandomTerminal() const – метод отримання випадковим чином клітини решітки з множини клітин кас;

– private: const Point getRandomPoint() const 0 – метод отримання випадковим чином клітини решітки;

– private: const Point getRandomPoint(const std::vector<Point>& points) const - метод отримання випадковим чином клітини решітки з набору;

– private: std::vector<Point> getNearestPoints(const Point& point, bool all, bool withSocialDistance) const – метод отримання усіх клітин решітки, що відповідають околиці, параметри const Point& point – клітина – центр околиці, bool all – флак признаку можливості порушення дистанції;

– private: bool isSocialDistanceBreak(const Point& point) – метод перевірки порушення дистанції, для клітини решітки const Point& point у околиці присутні інші покупці;

– private: int getDistance(const Point& point1, const Point& point2) const; - метод розрахунку відстані між двома клітинами решітки;

– private: bool CheckForExit() const – метод визначення, чи треба обрати об’єкту покупцю ціллю клітину виходу;

– private: bool CheckForNew() const - метод визначення, чи треба додати покупця у систему;

– private: bool CheckForBreakSocialDistance() const – метод визначення, чи можна порушити дистанцію.

Властивості класу:

– private:QModel* _model – об’єкт моделі

– private: const QSettings* _settings – об’єкт налаштувань;

– private: std::vector<Persone> _persones – вектор моделей покупців;

– private: std::vector<Point> _stalls – вектор клітин решітки - стелажів;

– private: std::vector<Point> _exitDoors– вектор клітин решітки - виходів;

– private: std::vector<Point> _terminals – вектор клітин решітки - кас;

– private: mutable std::mt19937 _gen – генератор випадкових чисел;

– private: bool _alarm = false – флаг – признак режимі евакуації;

– private: int _brakeSocialDistanceVal = 0 – частота порушення дистанції;

– private: int _goToExitVal = 0 – частота обрання виходу;

– private: int _newPersoneVal = 0 – частота появи покупців у моделі;

– private: int _currentTime = 0 – поточний час моделювання (кількість кроків);

– private: bool _pause = false – флаг-ознака призупинення модулювання;

– private: int _evacTime = 0 – час евакуації покупців;

– private: long _totalPeoplesCount – загальна кількість створених покупців.

Клас QSimulationArea успадкований від QWidget виконує візуалізацію стану решітки моделі приміщення.

Методи класу:

– public QSimulationArea(QWidget *parent, QModel* model) – конструктор класу;

– public void SetMud(bool val) – метод встановлення ознаки відображення матриці забруднення;

– public void SetBreak(bool val) - – метод встановлення ознаки відображення матриці порушення дистанції;

– private virtual void paintEvent(QPaintEvent*) override – метод пере рисовки області;

Властивості класу:

– private QModel* _model – об’єкт моделі;

– private bool _withMood = true – флаг-ознака відображення матриці бруду;

– private bool _withBreak = false - флаг-ознака режиму моделювання, звичайний чи евакуація.

Наведемо опис структур даних, що використовуються у програмній реалізації.

Структура Point описує об’єкт «точка».

Властивості структури:

– int posX = 0 – координата у решітці за шириною;

– int posY = 0 – координата у решітці за висотою;

Методи структури:

– Point() – конструктор структури;

– Point(int x, int y) – конструктор структури з параметрами, параметри: int x – координата точки за шириною решітки, int y – координата точки за висотою решітки;

– bool operator == (const Point& val) const – перевантажений оператор порівнення двох точок «равно»;

– bool operator != (const Point& val) const – перевантажений оператор порівнення двох точок «не равно».

Структура Persone успадковується від структури Point описує сутність «покупець».

Методи структури:

– Persone() {} - конструктор структури;

– Persone(int x, int y) : Point(x, y) - конструктор структури с параметрами, параметри: int x – координата покупця за шириною решітки, int y – координата покупця за висотою решітки;

– void GoToPoint(const Point& point) – метод встановлення точки для досягнення;

– void GoToGoal(const Point& point, bool toStall) – метод встановлення точки стелажа для досягнення;

– void GoToExit(const Point& point) – метод встановлення точки виходу для досягнення;

– bool wasOnPoint(const Point& point) – метод перевірки відвідування покупцем точки;

– void clearHistory() – метод очищення історії відвідувань.

Властивості структури:

– std::vector<Point> _history – вектор точок – історія переміщення;

– bool isGoToGoal = false – флаг-ознака переміщення до цілі;

– bool isGoToExit = false - флаг-ознака переміщення виходу;

– bool goalIsStall = false – флаг-ознака переміщення стелажу;

– Point goalPoint – точка цілі переміщення.

3 ПРОВЕДЕННЯ МОДЕЛЮВАННЯ ТА АНАЛІЗ РЕЗУЛЬТАТІВ

Для отримання даних для аналізу виконаємо ряд експериментів по моделюванню поведінки людей у приміщенні за допомогою побудованої моделі та розробленого програмного забезпечення. Для моделювання будемо використовувати схему розміщення об’єктів у приміщенні, що відповідає схемі на рисунку 2.7.

Отримаємо дані щодо залежності часу, необхідного на евакуацію від кількості людей у приміщенні.

На рисунку 3.1 наведено графіки, де в залежності від імовірності появи покупця зазначено кількість людей у приміщенні на початок евакуації та кількість часу, що знадобилося на звільнення приміщення. Параметри моделювання наведено у заголовках графіків. Значення отримані як середні за 20 моделювань.

На рисунку 3.2 наведено графіки, де в залежності від імовірності появи покупця зазначені кількість людей у приміщенні на початок евакуації та кількість випадків порушення дистанціювання, що трапилися до цього моменту. Параметри моделювання наведено у заголовках графіків. Значення отримані як середні за 20 моделювань.

На рисунку 3.3 наведено графіки залежності часу евакуації від середньої кількості людей у приміщенні, що отримано моделюваннями по 500 та 1000 кроків.

На рисунку 3.4 наведено графіки залежності кількості випадків порушення дистанції усіма покупцями, що відвідали приміщення, за різною імовірністю виникнення порушення – лояльністю покупців до порушення дистанції.

Аналізуючи отримані результати, можна зробити висновок, що отримані статистичні дані носять експоненціальний характер. Збільшення імовірності допущення порушення покупцями дистанції на 0.1 підвищило кількість випадків порушення дистанції на 25% при кількості кроків моделювання T = 500 та на 54% при T = 1000.

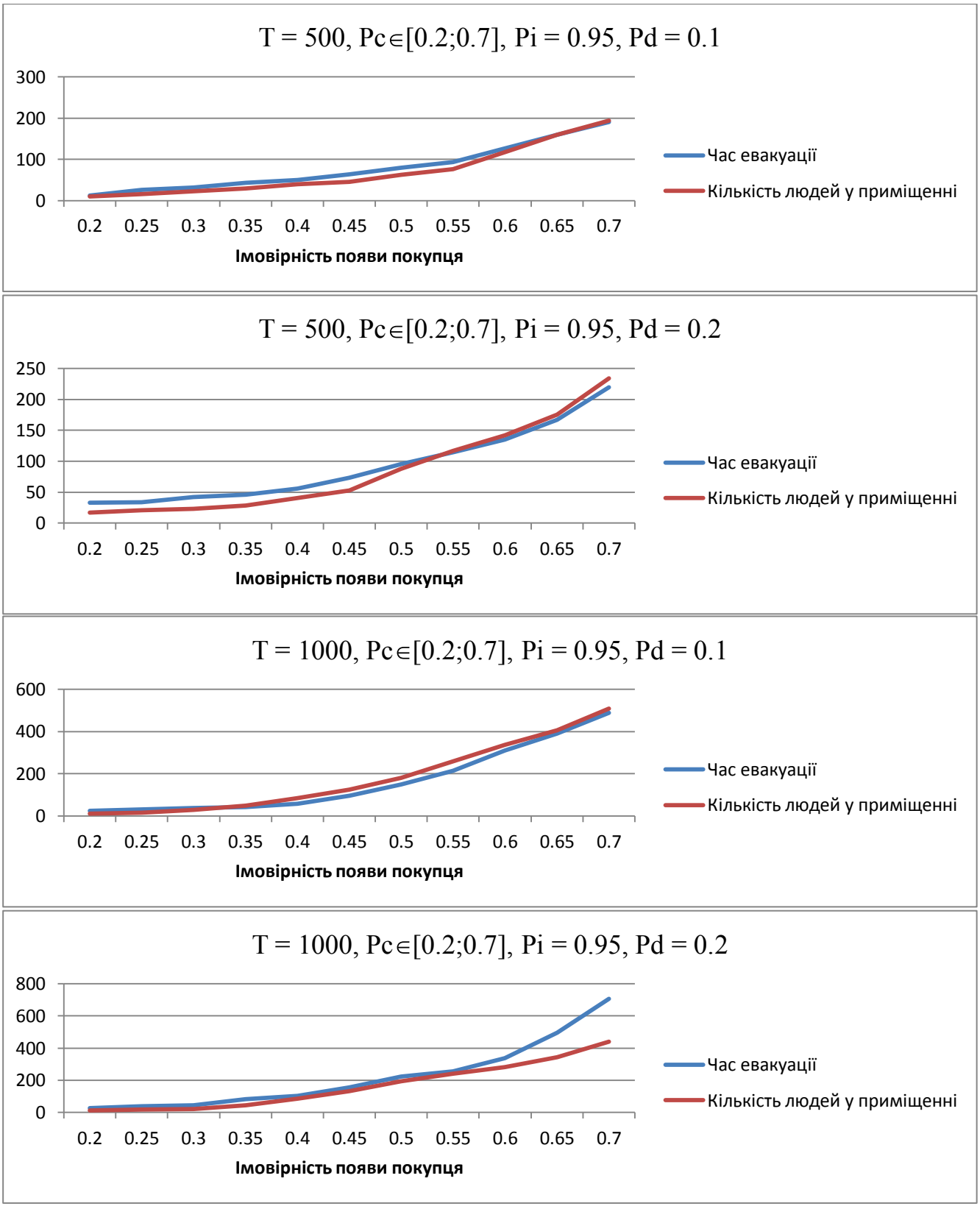


Рисунок 3.1 – Кількість людей та час евакуації

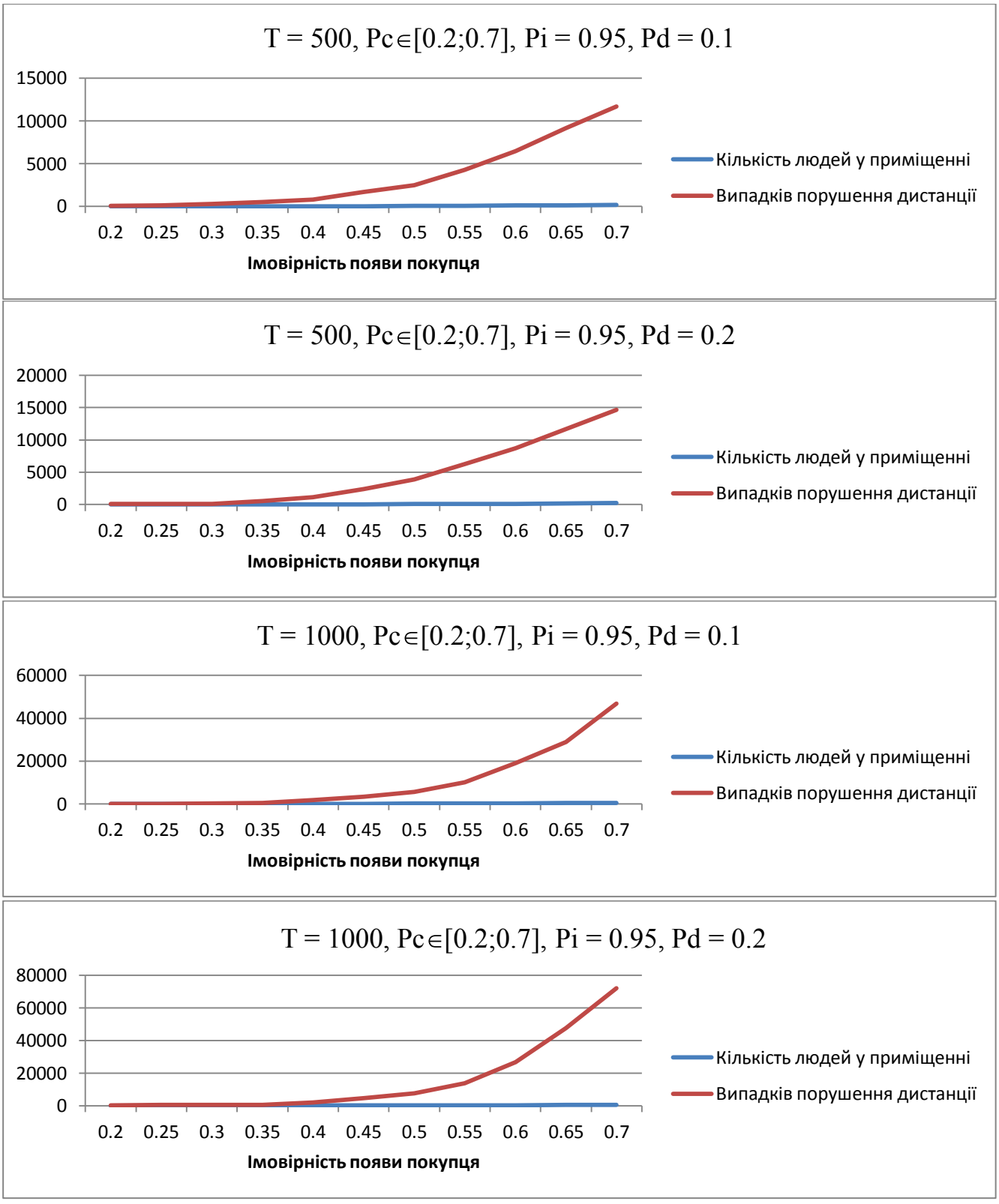


Рисунок 3.2 – Кількість людей та випадки порушення дистанції

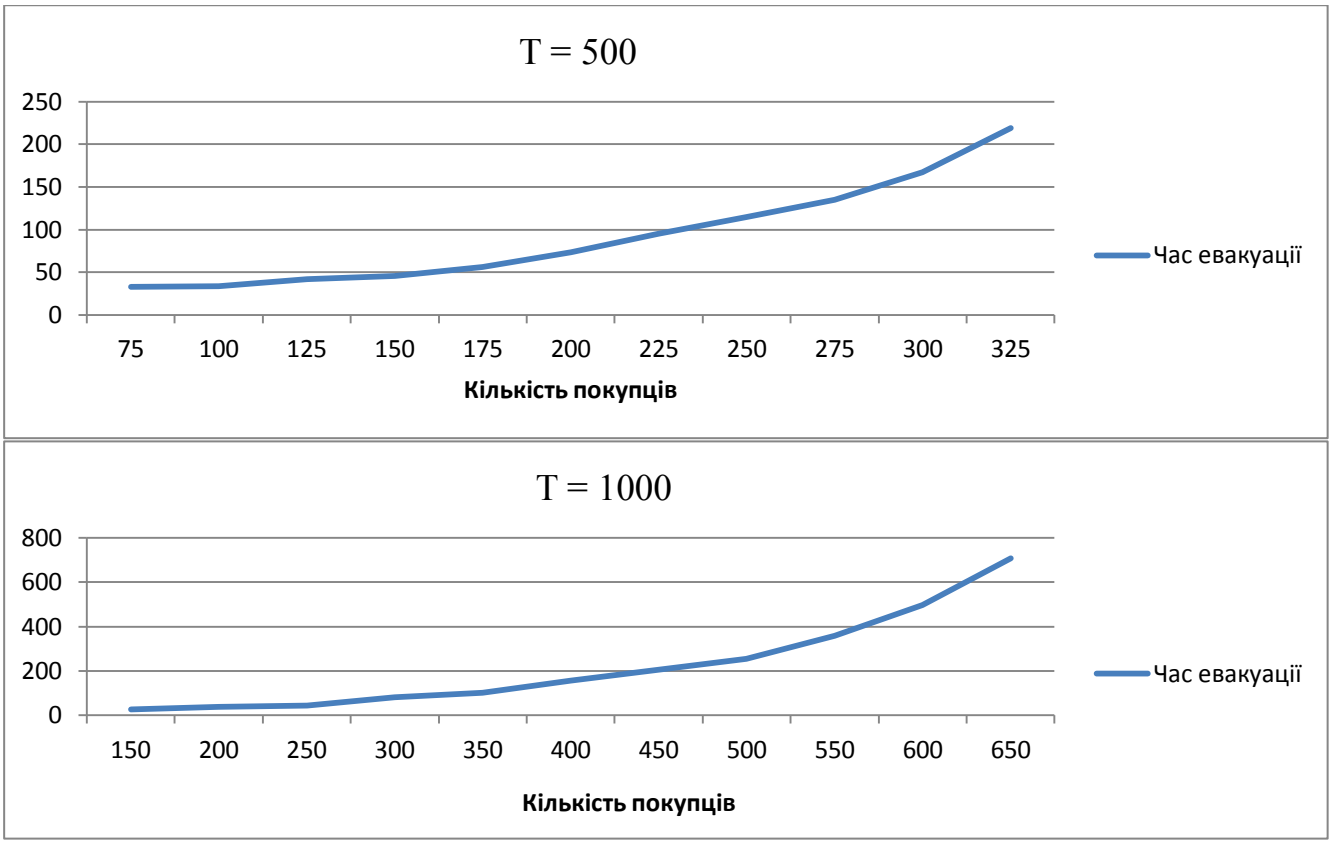


Рисунок 3.3 – Час евакуації за середним значенням кількості покупців

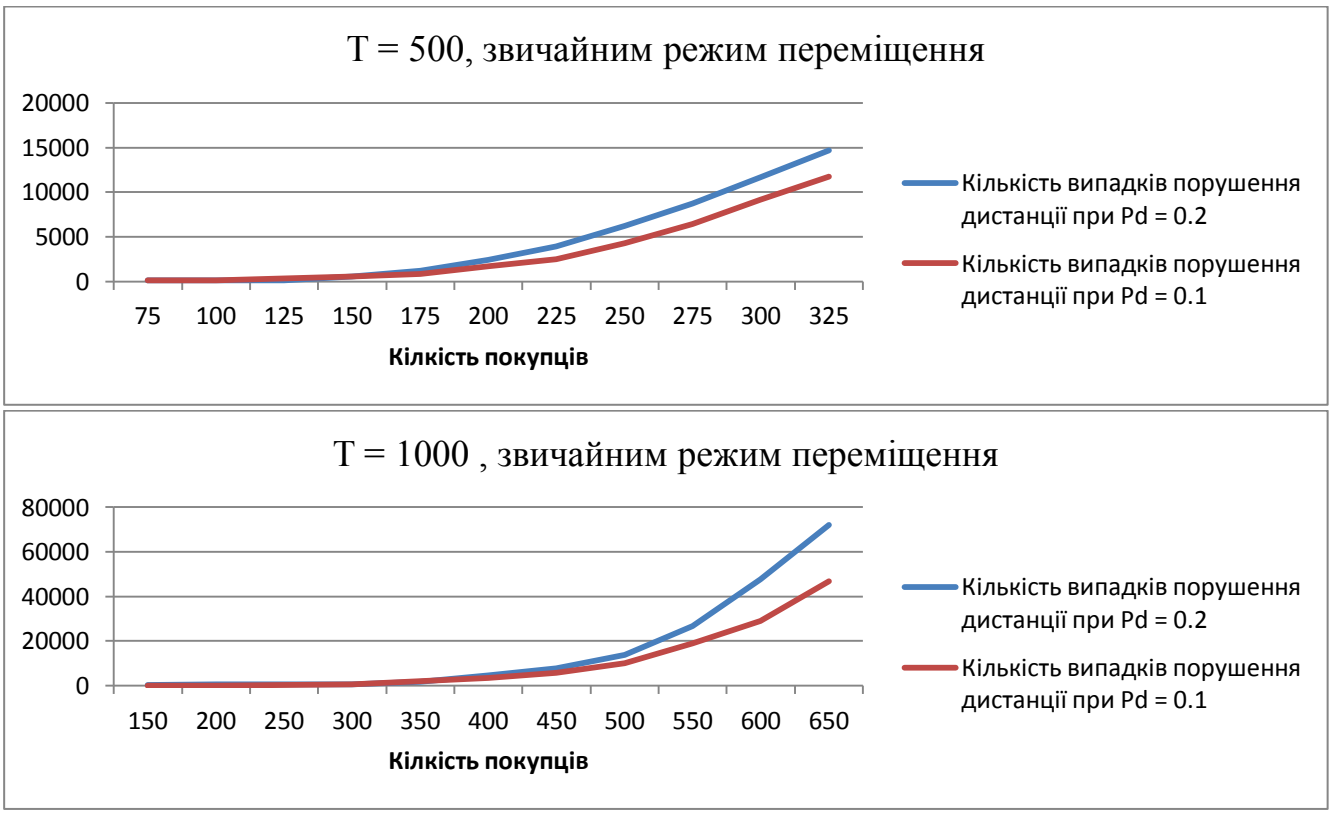


Рисунок 3.4 – Кількість випадків порушення дистанції за різної ймовірністю його виникнення

ВИСНОВКИ

В результаті виконання магістерської роботи було розроблено модель поведінки людей у приміщенні з використанням клітинних автоматів та розроблено програмне забезпечення для реалізації процесу моделювання та збору статистичних даних.

При виконанні роботи було виконано наступні задачі.

1. Проаналізовано існуючих підходи до побудови та використання клітинних автоматів у сфері моделювання соціально-поведінкових аспектів життя людини, зокрема, в процесі їх руху в приміщеннях в умовах соціального дистанціювання.

2. Обрано тип приміщення «магазин», на основі якого розроблено модель приміщення та модель поведінки людей. Було досліджено алгоритми поведінки людей у приміщенні магазину.

3. Розроблено модель поведінки людей у приміщенні торгового залу, що враховує соціального дистанціювання, звичайну поведінку та поведінку при виникненні екстремальної ситуації.

4. Розроблено та проведено тестування програмного забезпечення, що моделює поведінку людей у приміщенні, збирає та зберігає інформацію проміжних даних та результатів моделювання.

6. Проведено моделювання поведінки людей у приміщенні та виконано аналіз отриманих результатів. За результатами аналізу виявлено, що збільшення кількості людей у приміщенні, навіть при спробі дотримуватися дистанції, тягне за собою значне збільшення випадків порушення дистанції, а невдале розташування стелажів, кас та виходів значно підвищують забрудненість певних місць та утворюють зони потенційного порушення дистанції.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Тоффоли Т., Марголус Н. Машины клеточных автоматов / Т. Тоффоли, Пер. С англ. - М.: «Мир», 1991 – 280с , ил

2. Вікіпедія. Вільна енциклопедія. [Електронний ресурс] – Режим доступу: https://uk.wikipedia.org/wiki/Клітинний_автомат (дата звернення 30.04.2021). – Назва с екрану.

3. Helbing, Social force model for pedestrian dynamics, Physical review E, May 1995.

4. Helbing, Simulation of Pedestrian Crowds in Normal and Evacuation Situations, Pedestrian and Evacuation Dynamics Springer-Verlag, Berlin; Heidelberg; New York (2002) pp. 21-58.

5. M. APEL, K. T. WALDEER, Simulation of pedestrian Flows based on the Social Force Model Using the Verlet Link Cell Algorithm, Karl-Scharfenberg-Fakult¨at Salzgitter, Institut fur Simulation und Modellierung.

6. Ramin Mehran, Alexis Oyama, Mubarak Shah, Abnormal Crowd Behavior Detection using Social Force Model, IEEE International Conference on Computer Vision and Pattern Recongition (CVPR), Miami, 2009.

7. Was J., Gudowski B., Matuszyk PJ: Social Distances Model of Pedestrian Dynamics. In: El Yacoubi, Chopard B.,Bandini S. (eds.) ACRI 2006 LNCS, vol.4173, pp.492-501. Springer, Heidelberg (2006).

8. PTV GROUP. Офіційний сайт компанії [Електронний ресурс] – Режим доступу: https://company.ptvgroup.com/ru/produkty (дата звернення 15.05.2021). – Назва с екрану.

9. PTV Имитационное моделирование поведения пешеходов [Електронний ресурс] – Режим доступу: https://www.ptvgroup.com/ru/reshenija/produkty/viswalk/ (дата звернення 15.05.2021). – Назва с екрану.

10. CUBE Моделирование транспортных систем и землепользования пешеходов [Електронний ресурс] – Режим доступу: https://www.bentley.com/ru/products/brands/cub (дата звернення 15.05.2021). – Назва с екрану.

11. Quadstone Paramics.MultimodalTransport [Електронний ресурс] – Режим доступу: https://www.paramics-online.com/ (дата звернення 15.05.2021). – Назва с екрану.

12. SIMWALK Офіційний сайт компанії [Електронний ресурс] – Режим доступу: https://www.simwalk.com (дата звернення 15.05.2021). – Назва с екрану.

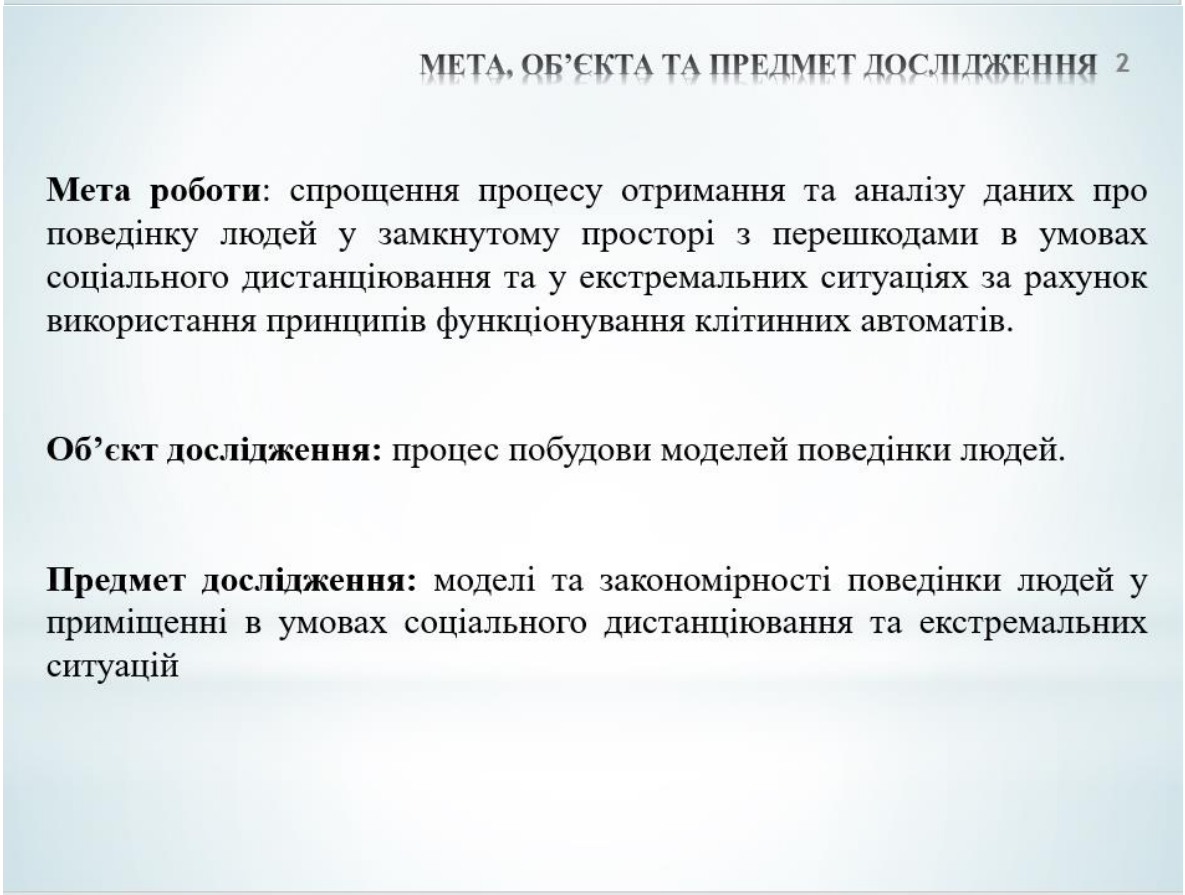
13. Посібник роботодавця з управління робочими місцями під час COVID-19 [Електронний ресурс] – Режим доступу:https://www.ilo.org/wcmsp5/groups/public/---ed_dialogue/---lab_admin/documents/publication/wcms_745603.pdf (дата звернення 15.05.2021). – Назва с екрану.

14. Про проведення громадського обговорення деяких вимог до протиепідемічних заходів при послабленні карантину. Розпорядження МОЗ України № 32 від 18.05.2020 [Електронний ресурс] – Режим доступу: https://zakon.rada.gov.ua/rada/show/v0032488-20#Text (дата звернення 15.05.2021). – Назва с екрану.

15. Qt Creator - A Cross-platform IDE for Application Development [Електронний ресурс] – Режим доступу: https://www.qt.io/product/development-tools (дата звернення 15.05.2021). – Назва с екрану.

Додаток Б

ДЕМОНСТРАЦІЙНІ МАТЕРІАЛИ



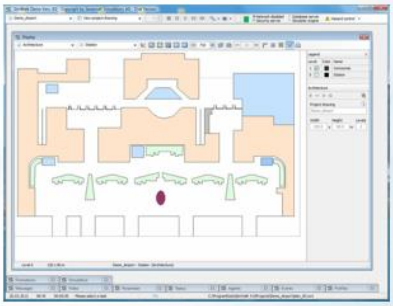
АНАЛОГІЧНІ ПРОГРАМНІ ПРОДУКТИ 3

PTV Vissim

CUBE Dynasim

Quadstone Paramics

SIMWALK



КЛІТИННИЙ АВТОМАТ 4

Клітинний автомат - дискретна динамічна система, сукупність однакових клітин, з'єднаних між собою, поведінка яких повністю визначається локальними взаємозв'язками клітин

Властивості:
– зміни значень всіх клітин відбуваються одночасно після обчислення нового стану кожної клітини решітки;
– решітка однорідна – неможливо розрізнити будь-які дві області решітки;
– взаємодії локальні, лише клітини околиці здатні вплинути на дану поточну клітину;
– безліч станів клітини кінцева.

Математичний опис. КА – це безліч кінцевих автоматів на площині позначених цілочисельними координатами (i, j) та кожен з яких може перебувати в одному з станів $\sigma_{i,j}$:

$$\sigma_{i,j} \in \sum \equiv \{0,1,2, \dots k-1, k\}.$$

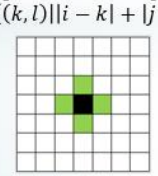
Зміна станів автоматів відбувається згідно з правилом переходу:

$$\sigma_{i,j}(t+1) = \varphi(\sigma_{k,l}(t) | (k,l) \in N(i,j),$$

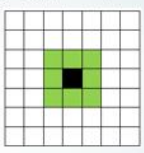
де N(i, j) – деяка околиця точки (i, j), безліч автоматів, що становлять сусідство.

Кількість всіх можливих правил переходу :$N_r = \sigma^{\sigma^n}$.

Околиця фон Неймана порядку 1
$N_{FN}^1(i,j) = \{(k,l) || i-k| + |j-l| \le 1\},$



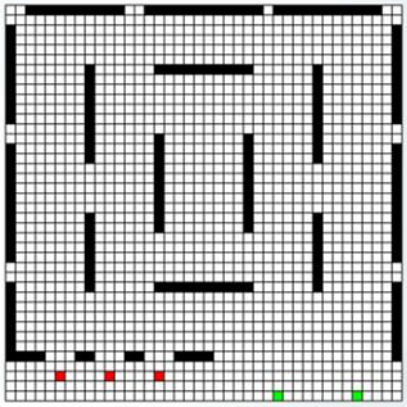
Околиця Мура порядку 1
$N_M^1(i,j) = \{(k,l) || i-k| \le 1, |j-l| \le 1\}.$



МОДЕЛЬ ПРИМІЩЕННЯ 5

Математична модель приміщення – множина клітин P, розміщених у вигляді решітки, де фрагменти решітки – клітини $p_{(i,j)}$:

$$p_{(i,j)} \in P.$$



Множина клітин P приміщення задається сукупністю множин 5ти типів, які не перетинаються:

$$P = P_{io} \cup P_g \cup P_c \cup P_p \cup P_{empty},$$
$$P_{io} \cap P_g \cap P_c \cap P_p \cap P_{empty} = \emptyset,$$

Pio – множина клітин входу/виходу;
Pg – множина клітин стелажів;
Pc – множина клітин кас;
Pp – множина клітин покупців;
Pempty – множина інших клітин (вільний простір).

МОДЕЛЬ ПОКУПЦЯ 6

Математична модель покупця $C = \{p_{cur}, p_{purpose}, f\}$,

p_{cur} – поточне місцезнаходження, $p_{cur} \in P_p$;
$p_{purpose}$ – місцезнаходження цілі,
$p_{purpose} \in P_{io} \cup P_g \cup P_c$;
f – правила зміни місцезнаходження.

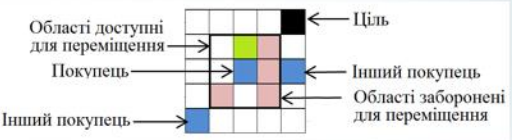
1) правило переміщення у звичайному режимі:

$$p_{next}(i_{next}, j_{next}) \in random\{(k,l)| \begin{cases} |i_{cur} - k| + |j_{cur} - l| \le 1; \\ (k,l) \ne (i_{next}, j_{next}); \\ (k,l) \in P_{empty}; \\ distance\left((k,l),(i_{purpose}, j_{purpose})\right) \to min; \end{cases}\},$$

$p_{next}(i_{next}, j_{next})$ – наступне місцезнаходження об'єкту;
$p_{purpose}(i_{purpose}, j_{purpose})$ – місцезнаходження цілі;

2) правило переміщення у екстремальному режимі:

$$p_{nextEx}(i_{next}, j_{next}) \in random\{(k,l)| \begin{cases} |i_{cur} - k| + |j_{cur} - l| \le 1; \\ (k,l) \ne (i_{next}, j_{next}); \\ (k,l) \in P_{empty}; \\ distance\left((k,l),(i_{purpose}, j_{purpose})\right) \to min \end{cases}\};$$

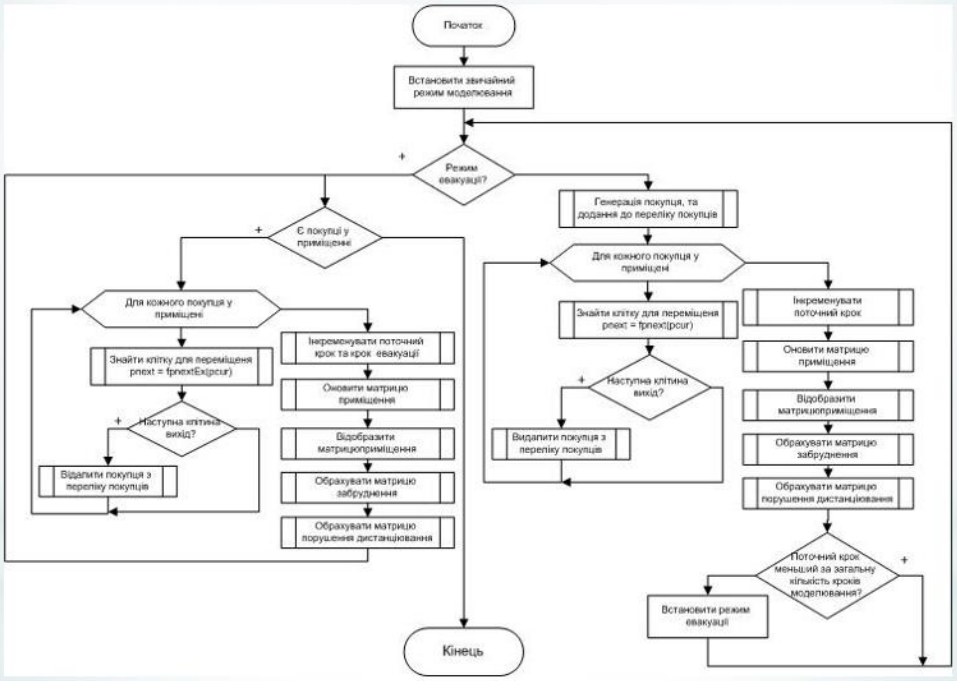


3) правило обрання цілі:

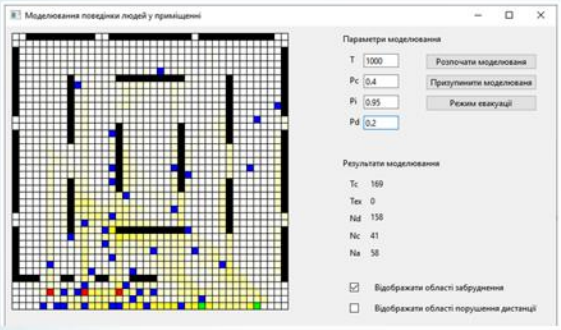
$$p_{purpose_{next}}(i_{purpose_{next}}, j_{purpose_{next}}) \in \{(k,l)| \begin{cases} (k,l) \in P_{io} \cup P_g \cup P_c, \text{якщо } p_{purpose}(i_{purpose}, j_{purpose}) \in P_g; \\ (k,l) \in P_{io} \cup P_g, \text{якщо якщо } p_{purpose}(i_{purpose}, j_{purpose}) \in P_c. \end{cases}\};$$

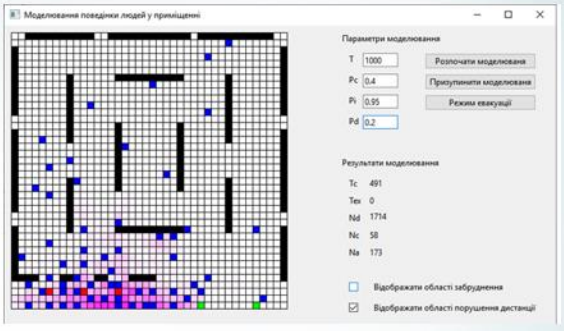
4) правило закінчення переміщення: $p_{next}(i_{next}, j_{next}) \in P_{io}$.

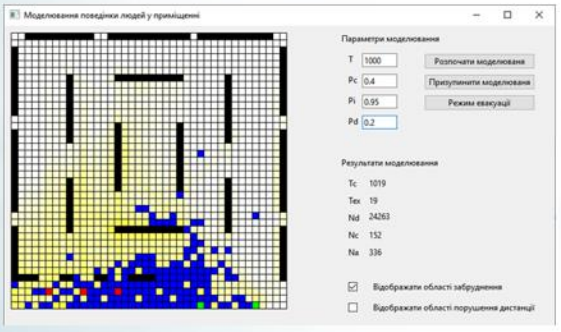
УЗАГАЛЬНЕНИЙ АЛГОРИТМ ПРОВЕДЕННЯ МОДЕЛЮВАННЯ 7

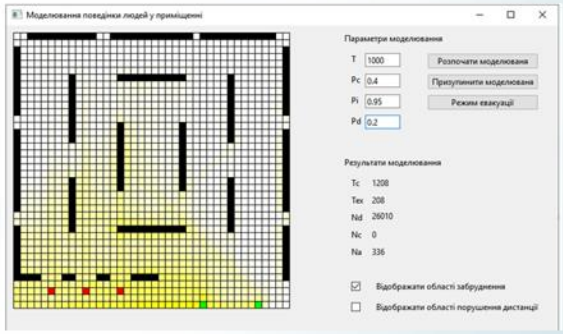


ПРИКЛАДИ ПРОЦЕСУ МОДЕЛЮВАННЯ 8

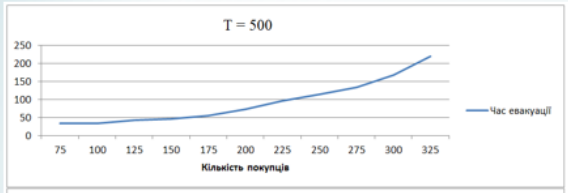


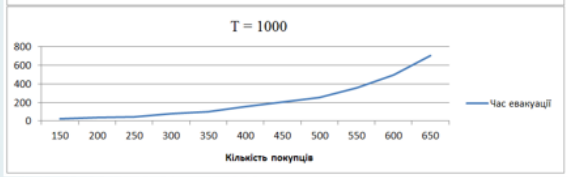


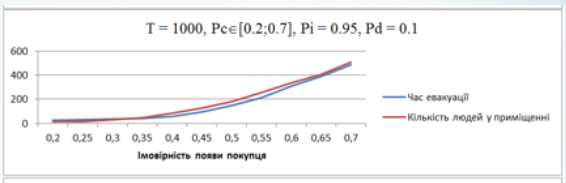


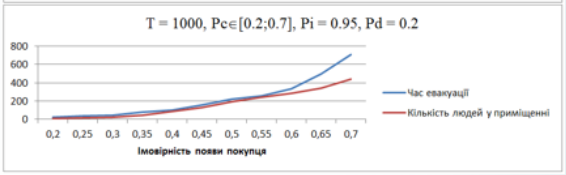


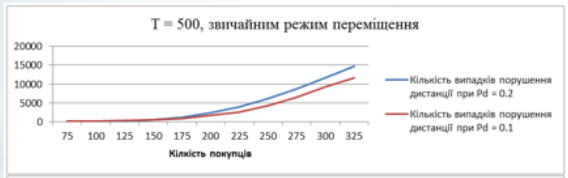
РЕЗУЛЬТАТИ МОДЕЛЮВАННЯ 9

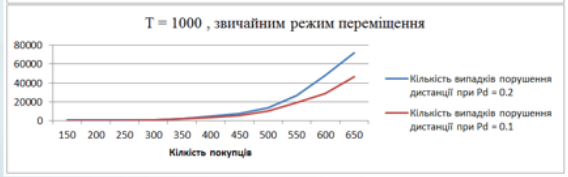


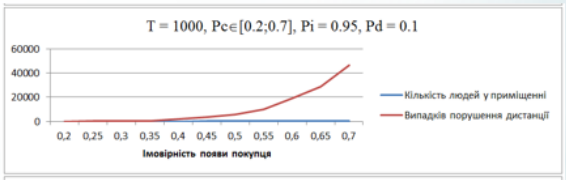


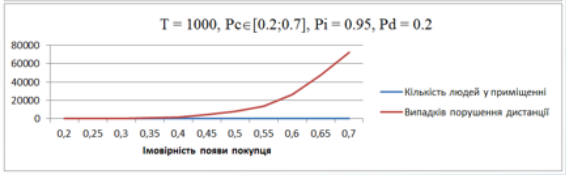












ВИСНОВКИ 10

В результаті виконання магістерської роботи було розроблено модель поведінки людей у приміщенні з використанням клітинних автоматів та розроблено програмне забезпечення для реалізації процесу моделювання та збору статистичних даних.

При виконанні роботи було виконано наступні задачі.

1. Проаналізовано існуючих підходи до побудови та використання клітинних автоматів у сфері моделювання соціально-поведінкових аспектів життя людини.
2. Обрано тип приміщення «магазин», на основі якого розроблено модель приміщення та модель поведінки людей. Було досліджено алгоритми поведінки людей у приміщенні магазину .
3. Розроблено модель поведінки людей у приміщенні торгового залу, що враховує соціального дистанціювання, звичайну поведінку та поведінку при виникненні екстремальної ситуації.
4. Розроблено та проведено тестування програмного забезпечення, що моделює поведінку людей у приміщенні, збирає та зберігає інформацію проміжних даних та результатів моделювання.
5. Проведено моделювання поведінки людей у приміщенні та виконано аналіз отриманих результатів. За результатами аналізу виявлено, що збільшення кількості людей у приміщенні, навіть при спробі дотримуватися дистанції тягне за собою значне збільшення випадків порушення дистанції, а невдале розташування стелажів, кас та виходів значно підвищують забрудненість певних місць та утворюють зони потенційного порушення дистанції.

Додаток Б

ФРАГМЕНТИ ОСНОВНИХ ПРОГРАМНИХ МОДУЛІВ