Міністерство освіти і науки України
Львівський національний університет імені Івана Франка
Кафедра радіоелектронних і комп'ютерних систем

КУРСОВА РОБОТА

на тему:

«Застосування теорії ґрафів та теорії ігор у розробці відеоігор»

Виконав студент 5 курсу групи ФеІм-14с
Напрям підготовки 122 «Комп’ютерні науки»
Дячков М. В.
Науковий керівник:
Професорка кафедри радіоелектронних і комп'ютерних систем
Клим Галина Іванівна

Національна шкала _____
Кількість балів _____
Оцінка ECTS _____
Члени комісії:

_____ _____
(підпис) (прізвище та ініціали)

_____ _____
(підпис) (прізвище та ініціали)

_____ _____
(підпис) (прізвище та ініціали)

Львів 2023

Зміст

Зміст 2

Вступ 3

1. Теорія ґрафів. Загальні обриси 4

1.1. Основні поняття 4

1.2. Складніші структури ґрафів 6

1.3. Комунікативні мережі 11

1.4. Найчастіші проблеми та їх вирішення 13

2. Теорія ігор та її зв’язок із ґрафами 17

2.1. Основні положення 17

2.2. Застосування в об’єднанні з теорією ґрафів 22

3. Приклади та перспективи використання описаних структур у системах прийняття рішень при розробці відеоігор 24

3.1. Приклади задач та існуючі системи 24

3.2. Перспективи застосування 25

Висновки 29

Джерела 30

Вступ

Комп'ютерні ігри є популярними формами розваги, які привертають увагу мільйонів гравців по всьому світу. З розвитком технологій і зростанням амбіцій розробників, геймплей стає все більш складним і захоплюючим. Проте, забезпечення високої якості ігрового досвіду вимагає глибокого розуміння структури та динаміки гральних процесів.

У цьому контексті, стає все більш очевидним, що традиційні підходи до моделювання геймплею мають свої обмеження. Лінійні структури сюжету не здатні повністю відобразити складність взаємодій та альтернатив, які пропонують сучасні комп'ютерні ігри.

У зв'язку з цим, виникає потреба у нових підходах до моделювання комп'ютерних ігор, які б забезпечували гнучкість, необмежені можливості та інтерактивність. Один із напрямків розвитку цієї області - графова модель гральних процесів, що базується на математичних графах.

Ця робота присвячена дослідженню та розвитку графової моделі комп'ютерних ігор. Ми вивчаємо можливості, які надає використання графів для представлення та керування гральними структурами. Досліджується використання графових правил для опису динаміки геймплею та моделювання взаємодій між різними елементами гри.

Також в роботі розглядаються практичні застосування графової моделі в розробці комп'ютерних ігор, можливості автоматизації тестування та оптимізації гральних структур з використанням формальних методів, що дає розробникам можливість створювати більш ефективні та захоплюючі ігрові досвіди.

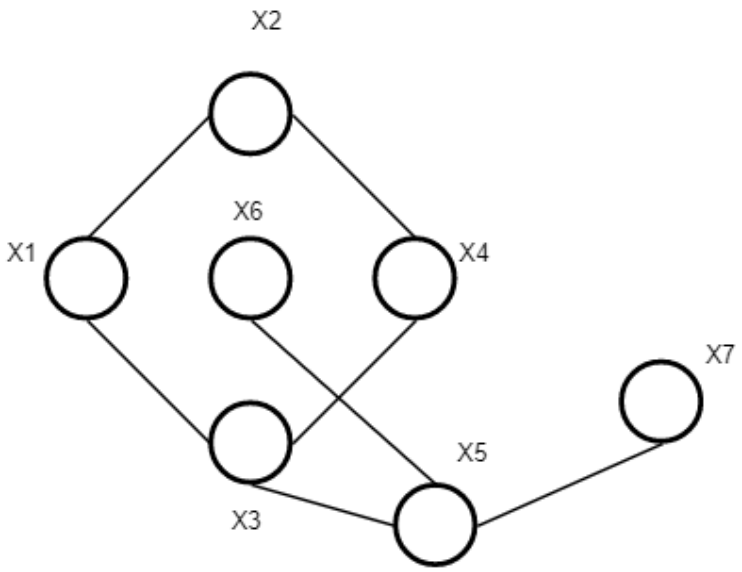
Остаточна мета цієї роботи полягає в тому, щоб показати потенціал графової моделі розробки комп'ютерних ігор і сприяти подальшому розвитку цього напрямку.

1. Теорія ґрафів. Загальні обриси.

1.1. Основні поняття

Теорія ґрафів – розділ математики, що вивчає такі структури як *ґрафи*. Ґрафом G називаються пари множин (V, E), де V – не-порожня множина значень, що називаються *вершинами* або *вузлами* (vertices, nodes), а E – множина двочленних підмножин V, що називаються *ребрами* (edges).

Якщо $x_i, y_j \in E$, тоді ребро $E\{x_i, y_j\}$ вважається *інцидентним*.



Мал. 1. Найпростіший ґраф

Множини елементів для даного ґрафу виглядають наступним чином:

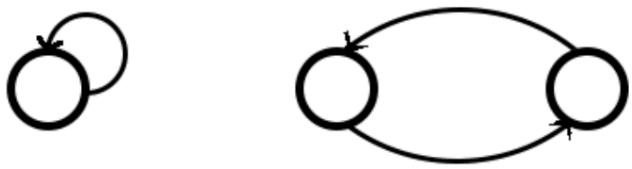
$$V = \{x_1, x_2, \dots, x_7\},$$

$$E = \{\{x_1, x_2\}, \{x_1, x_2\}, \dots, \{x_5, x_6\}, \{x_5, x_7\}\}.$$

Кількість ребер, що приєднані до однієї вершини є *кутом вершини.*

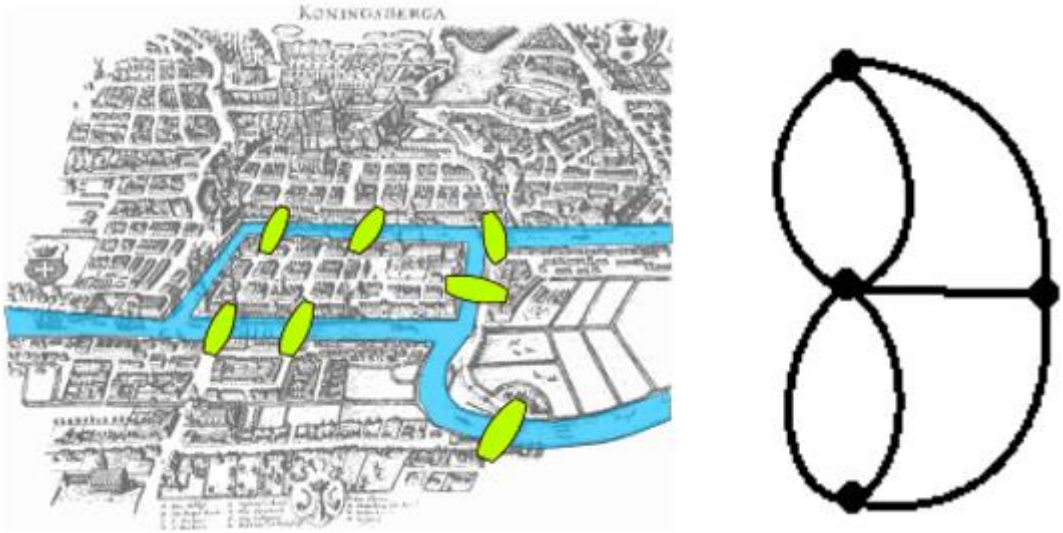
Шляхом називають будь яку скінченну послідовність вершин з’єднаних ребрами.

Ґраф називають *простим*, якщо він не має жодних петель (loops) або кратних ребер (multiedges).



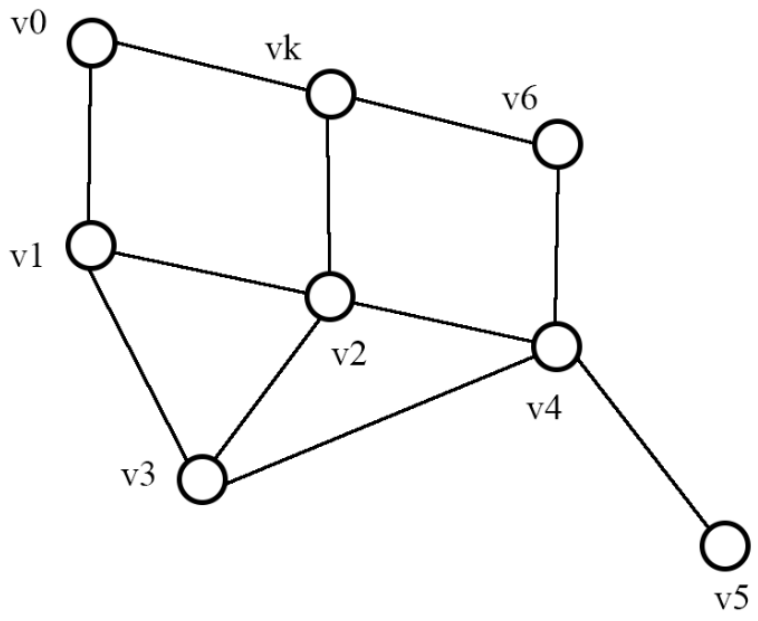
Мал. 2. Петля та кратні ребра

Сама проблема вперше була сформульована у XVIII ст. швейцарським математиком Леонардом Ейлером, який поставив собі задачу – «Чи можна подолати 7 мостів у місті таким чином, щоби по кожному пройти лише раз?». Сьогодні це відомо як задача про Кеніґсберґські мости. Вчений сам сформулював цю задачу і сам її вирішив, заклавши початок розвитку теорії ґрафів.



Мал. 3. Мапа тогочасного Кеніґсберґу та представлення шляху у вигляді мультиграфу

Візьмемо для прикладу певний ґраф, який надалі називатимемо «ґраф G_a».



Мал. 4. Ґраф G_a

Маршрутом називається такий маршрут, у якому всі вершини різні.

Якщо в маршруті всі ребра різні, і при цьому маршрут не замкнений, то маршрут носить назву *ланцюг.* Ланцюгом у ґрафі G_a може бути

$$\{v_0 - v_1 - v_3 - v_2 - v_4 - v_6 - v_k\}$$

Шляхом Ейлера називається такий ланцюг що проходить *усіма* ребрами лише 1 раз.

1.2. Складніші структури ґрафів.

Дерева

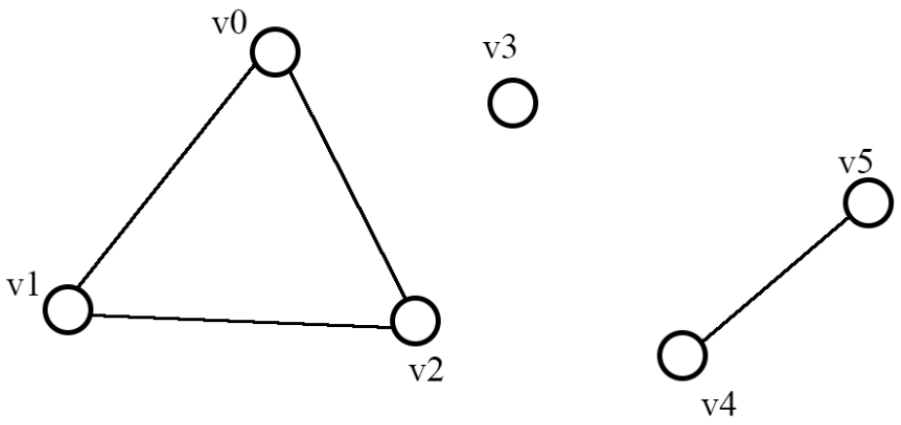
Для опису такої структури як *дерево* нам знадобиться ще декілька визначень.

Вершини u та v є зв’язаними, якщо існує маршрут від u до v.

Ґраф називається *зв’язаним*, коли кожна пара вершин у ньому є зв’язаною.

Відповідно, у не зв'язного графа розрізняються компоненти зв'язності, які є незалежними зв'язними частинами графа. Кожна компонента зв'язності має маршрути між будь-якими вершинами в межах цієї компоненти, і неможливо додати нову вершину до компоненти без порушення зв'язності.

Ребро, що якщо його забрати із ґрафу, збільшує кількість компонент зв’язності називається *мостом.*



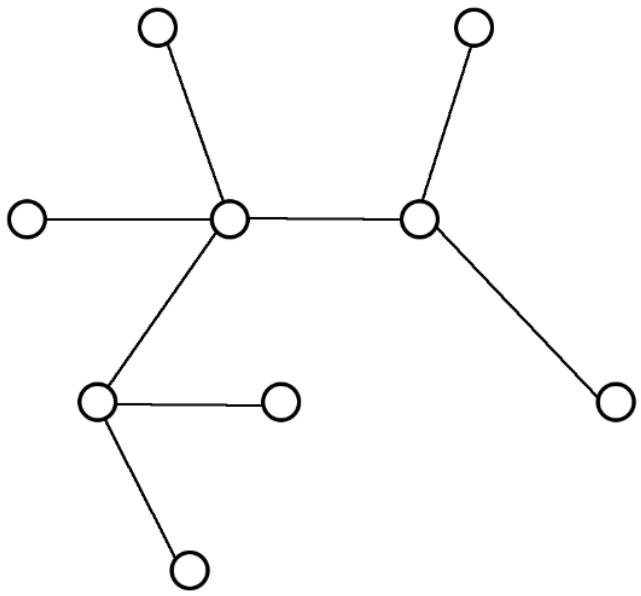
Мал. 5. Три компоненти зв'язності не зв'язаного ґрафа

Замкненим шляхом називається такий шлях, що починається та закінчується на одній і тій самій вершині.

Циклом називається замкнений шлях, де кількість вершин $k \geq 3$, та усі вони, окрім початково-кінцевої, є різні (властивість маршруту).

Маючи усі ці визначення ми нарешті можемо описати дерево.

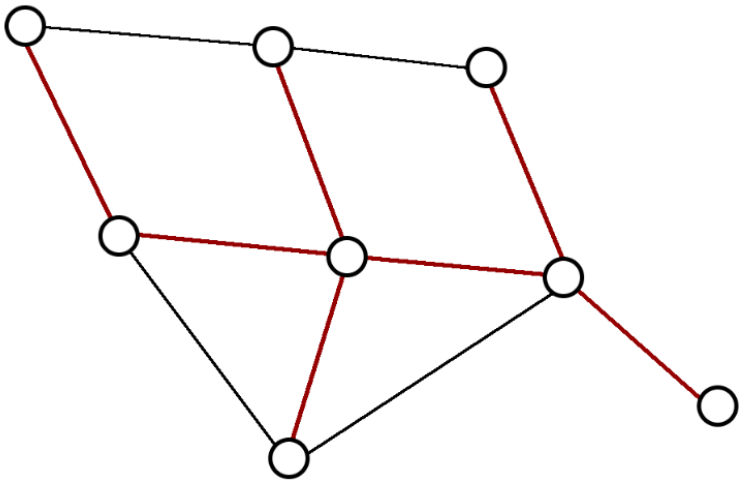
З’єднаний ґраф, що не має циклів, називається *деревом* (tree).



Мал. 6. Дерево

Також, кожна вершина дерева, що має кут 1 називається *листком* (leaf).

Кістяковим деревом з'єднаного ґрафу G є підґраф, що є деревом із тією самою кількістю вершин, що і ґраф G. До того ж, кожен з'єднаний ґраф має принаймні 1 кістякове дерево.



Мал. 7. Червоними товстішими ребрами показано кістякове дерево ґрафа G_a

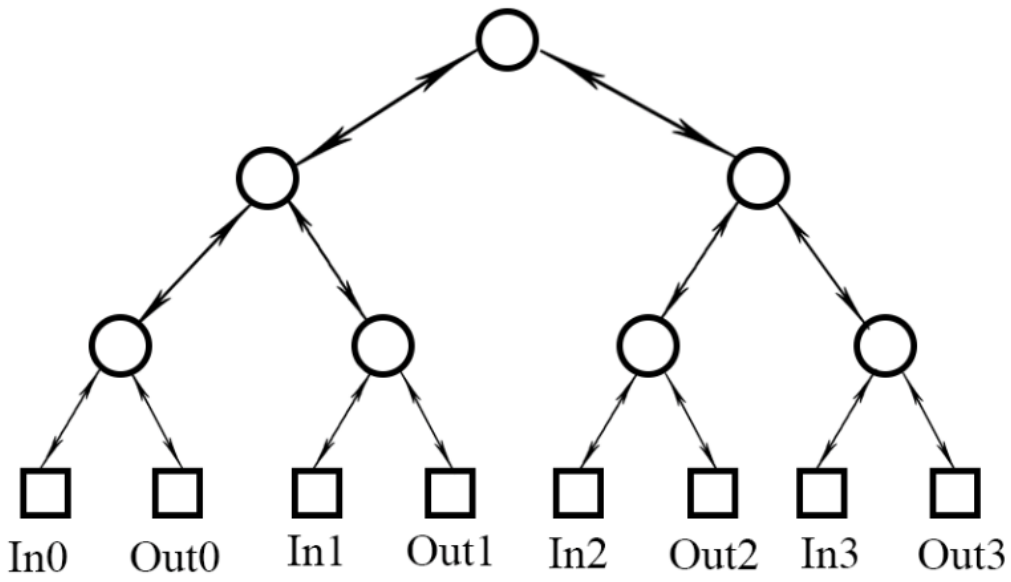
Тут ми репрезентуємо *термінали* (terminals) – листя бінарного дерева, щоявляють собою блок входу (input) або блок виходу (output) – що позначатимуться □ на схемах ґрафів та використовуватимуться у наступному розділі. Вершини $|V|$, ◯ вважатимуться *перемикачами* (switch), що через них проходить інформація із вхідного терміналу до вихідного. Розмір мережі NxM дорівнює кількості input N помножену на кількість output M. В наступних прикладах розглядатимуться рівні мережі, тому розмір виражатиметься NxN.

<u>*Бінарне дерево*</u>

Бінарним деревом називається таке дерево, що має 1 вершину, що зветься *коренем*, яка з'єднана із 2-ма наступними вершинами кратними ребрами. Від кожної із цих вершин може виходити така сама пара і так далі. *Повним бінарним деревом* називається таке бінарне дерево, що відповідає залежності

$$|V| = 2^i, i \geq 1,$$

де i – кількість рівнів дерева.



Мал. 8. Повне бінарне дерево із мережевою властивістю

Діаметром мережі називається довжина найкоротшого маршруту між input та output, що є найбільш віддалені.

Віднині ми будемо зважати на таку характеристику, як *затримка* (latency), що виражається часом, який необхідний пакету переміститись з input до output.

Діаметр бінарного дерева складає $2 * (1 + \log_2 N)$. Значення є логарифмічним, тож результат цілком задовільний. *Розміром перемикача* є його кількість input до кількості output. Не рахуючи кореня бінарне дерево має розмір перемикача 3x3. Це доволі велика, як ми побачимо далі, кількість напрямлених ребер на один перемикач для передачі даних. *Кількість перемикачів* у повному бінарному дереві рівна $2N - 1$.

Перестановкою (permutation) називається функція $\Pi: \{0 \div N - 1\} \rightarrow \{0 \div N - 1\}$, де немає 2-х чисел, що маєть однакове значення.

$$\Pi(i) = \Pi(j) \text{ тоді і тільки тоді, коли } i = j.$$

Проблемою знаходження шляху перестановки називається задача, коли для кожного i потрібно направити пакети із In_i до $Out_{\Pi(i)}$ маршрутом, що записується $P_{i,\Pi(i)}$.

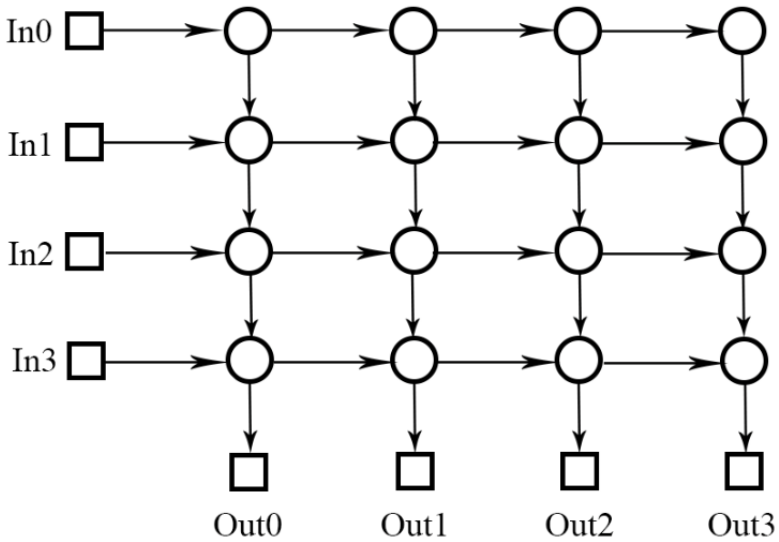
Заторовість (congestion) у шляху $P_{0,\Pi(0)}, \dots, P_{N-1,\Pi(N-1)}$ відповідає найбільшому число маршрутів, що проходять через один перемикач.

Кількість заторів у повному бінарному дереві рівна N, що є неймовірно великим значенням. Дійсно, аби подолати шлях з одніє сторони дерева до іншої ми обов'язково проходимо через кореневий перемикач. Якщо ж з якихось причин кореневий перемикач перестане функціонувати, то ми отримаємо 2 компоненти зв'язності; система розпадається.

Давайте поглянемо на інші приклади мережевих ґрафів, які можливо матимуть кращі характеристики.

Двовимірний масив

Поглянувши на будову мережі представлену двовимірним масивом можемо встановити наступні властивості: діаметр – $2N$, розмір перемикачів – 2x2, кількість перемикачів – N^2, заторовість – 2.



Мал. 9. Ґраф структури двовимірного масиву з мережевою властивістю

Тут ми маємо структуру, що має разючу перевагу у значеннях розміру перемикача та заторовості, натомість ми отримали лінійний діаметр та квадратну залежність кількості перемикачів, що також далеко не завжди є гарним варіантом для застосування. Проте спробуємо об'єднати ці дві концепції в одну.

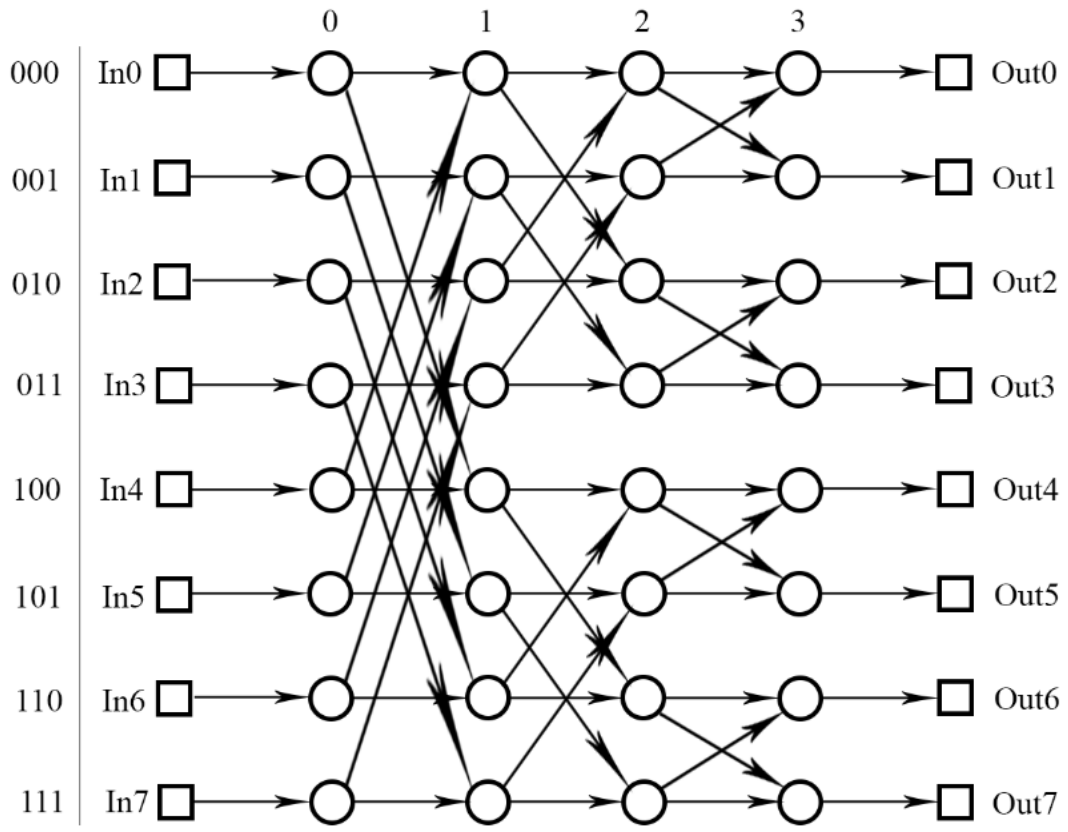
1.3. Комунікативні мережі

Тут ми підійшли до наріжного каменю курсової роботи. Ґрафи, що розглядатимуться в цьому підрозділі, та їх характеристики і будуть предметом подальшого дослідження. Але спершу розглянемо наступні структури.

Мережа-метелик

Об'єднавши підхід подвійного розгалуження, як у деревах та прямокутну структуру, як у двовимірних масивах, ми можемо отримати так звану *мережу-метелика.* Характеристики метелика є наступні: діаметр – $2 + \log_2 N$, розмір перемикачів – 2x2, кількість перемикачів – $N(1 + \log_2 N)$, заторовість – $\sqrt{N}$. Таким чином ми маємо одночасно логарифмічного виду діаметр та кількість перемикачів, а значення заторовості є десь приблизно посередині між повним бінарним деревом та двовимірним масивом. Такий спосіб передачі пакетів є ефективнішим, бо кожен input може претендувати на свій унікальний маршрут.

Перемикачі є «унікально ідентифіковані» відповідно до рядка та стовпчика і можуть бути показані у вигляді бінарного коду, та вибір шляху за допомогою бінарного коду є дуже простим – якщо є перехід від i-input до j-output, $i \neq j$, то номер колонки відповідає номеру $N + 1$ цифри у двійковому числі , і якщо цифри In та Out відрізняються, то шлях іде в бік розгалудження, а не за прямим ребром.

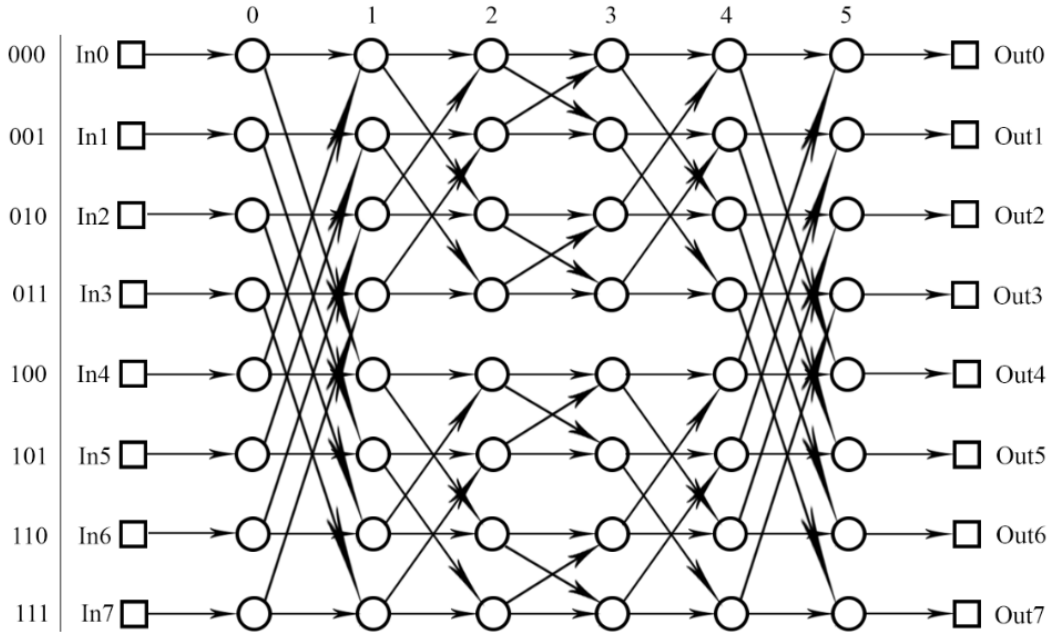


Мал. 10.Мережа-метелик на 8 парвходу-виходу

Тим не менш, ми можемо зробити ще одне покращення мережі, продублювавши розгалудження у зворотньому напрямі.

<u>Мережа-метелик Бенеса</u>

Така мережа матиме наступні характеристики: діаметр – $1 + 2\log_2 N$, розмір перемикачів – 2x2, кількість перемикачів – $2N\log_2 N$, заторовість – 1. Таким дивовоижним чином ми звели показник заторовості до 1, тобто в будь якому варіанті перестановки в нас є унікальний шлях від input до output.

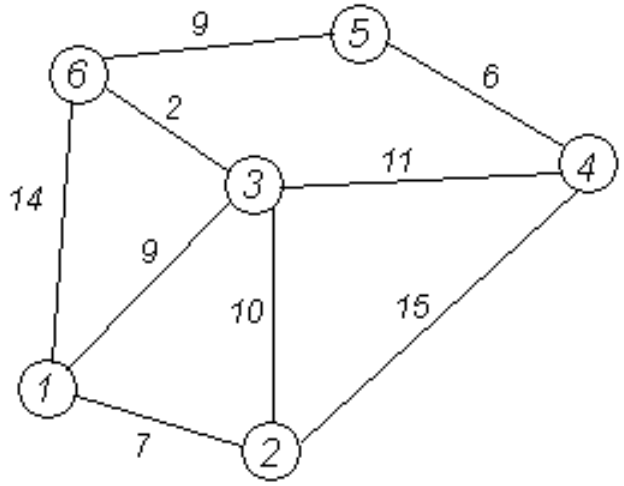


Мал. 11. Мережа-метелик Бенеса на 8 пар входу-виходу

1.4. Найчастіші проблеми та їх вирішення

Алґоритм Дейкстри

Ребра ґрафів можуть мати *довжину/вагу* (weight). Термін означення не потребує, зрозумілий інтуїтивно та краще описується на практиці. Даний алгоритм знаходить найкоротший шлях від однієї вершини графу до всіх інших вершин. Класичний алгоритм Дейкстри працює тільки для графів без ребер від'ємної довжини.



Мал. 12. Приклад ґрафу для проходження ним алґоритму Дейкстри

Необхідно задати початок та кінець шляху. Нехай цим значенням відповідають v_1 та v_5 відповідно.

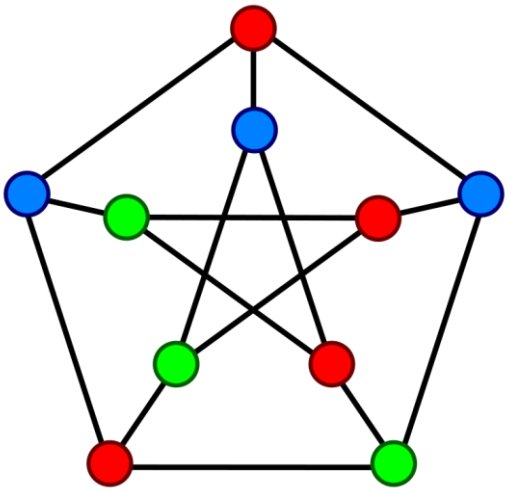
1. Визначаємо відстані до всіх вузлів $|V|$ як нескінченні, а відстань a до початкового вузла - 0.
2. Обираємо початковий вузол v_1 як поточний вузол.
3. Для поточного вузла обчислюємо відстань до всіх сусідніх вузлів шляхом додавання ваги ребра до поточної відстані. Якщо отримана відстань менша за поточну відстань до цього вузла, оновити поточну відстань.
4. Позначаємо поточний вузол як пройдений.
5. Якщо всі вузли були відвідані або немає доступних вершин для оновлення відстані, алгоритм завершує роботу.
6. Вибрати невідвіданий вузол з найменшою відстанню як новий поточний вузол і повернутися до кроку 3.

Розфарбування ґрафів

Проблема розфарбування графів полягає у призначенні кожному вершині графу певного кольору таким чином, щоб жодні дві суміжні вершини не мали одного й того ж кольору. Нижче розпишу пункти алгоритму вирішення цієї проблеми:

1. *Вибір початкової вершини.* Обрати будь-яку вершину графу в якості початкової.
2. *Покриття вершин.* Проходити по кожній вершині графу і призначати їм кольори притримуючись алґоритму розфарбування (жодні дві суміжні вершини не можуть мати одного й того ж кольору).
3. *Вибір кольору.* Вибрати доступний колір, який ще не використовувався для суміжних вершин. Якщо всі доступні кольори вже використані – додати новий у множину кольорів.
4. *Застосування кольору.* Призначити поточній вершині колір.

5. *Перевірка умови розфарбування.* Перевірити, чи задовольняються всі умови розфарбування. Якщо немає вершин, які мають один і той самий колір і є суміжними, переходимо до наступної вершини.
6. *Повторення.* Повторити кроки 3-5 для кожної вершини графу до тих пір, поки не буде призначений колір кожній вершині.
7. *Вивід результату.*

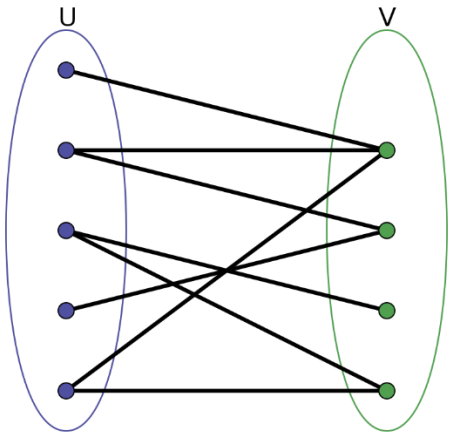


Мал. 13. Простий приклад розфарбованого ґрафу

Мінімальна кількість кольорів, в які може бути забарвлений ґраф називається *хроматичним числом* $\chi(G)$. Якщо кожна вершина у ґрафі G має кут $\leq d, d = |E|$, тоді стандартний алґоритм розфарбування використовує щонайбільше $d + 1$ кольорів для G.

<u>*Проблема стійких шлюбів*</u>

Проблема стійких шлюбів постає тоді, коли ми маємо *дводольний ґраф* (такий, у якому є 2 групи, наприклад чоловіки b та жінки g, що не мають ребер поміж собою, проте мають ребра з іншою групою) та нам необхідно утворити пари згідно з пріорітетом уподобань кожного елемента.



Мал. 14. Ілюстративний дводольний граф

1. *Ініціалізація.* Кожен елемент з кожної групи формує початкове вподобання щодо елементів іншої групи.
2. *Вибір невручністю.* Обираємо одну з груп, наприклад, чоловіків b, і починаємо з першого елемента цієї групи.
3. *Попарне співставлення.* Отримати у поточного елемента b список елементів групи g, відсортованих за їх вподобанням. Перевірити, чи вже є стабільне співставлення для цього елемента. Якщо так, то переходимо до наступного елемента.
4. *Вибір потенційного партнера.* Обираємо перший елемент без пари зі списку вподобань поточного елемента.
5. *Перевірка вподобання.* Перевіряємо, чи має поточний партнер g пару з іншим елементом b. Якщо так, то утворюється пара, і обидва елементи включаються до стабільного співставлення. Якщо ні, перевірити, чи вподобання поточного елементу b краще, ніж у його поточного партнера.
6. *Оновлення стабільного співставлення.* Якщо поточний партнер має пару, а вподобання поточного елемента краще, ніж у його поточного партнера, тоді треба оновити стабільне співставлення, виключивши поточного партнера і додавши обраного елемента.
7. *Повторення.* Повторюємо кроки 3-6 для кожного елемента вибраної групи, поки всі елементи не матимуть пару.
8. *Вивід результату.*

2. Теорія ігор та її зв’язок із ґрафами

2.1. Основні положення

Теорія ігор – теорія математичних моделей прийняття оптимальних рішень в умовах конфлікту. Оскільки сторони, що беруть участь в більшості конфліктів, зацікавлені в тому, щоб приховати від супротивника власні наміри, прийняття рішень в умовах конфлікту, зазвичай, відбувається в умовах невизначеності.

Теорія ігор є захоплюючим інтелектуальним інструментом, який дозволяє нам розкрити складність взаємодії між раціональними гравцями у конфліктних ситуаціях. Вона знаходить своє застосування в економіці, політичній науці, біології та багатьох інших галузях, де аналіз стратегічного прийняття рішень має велике значення.

Вивчення теорії ігор дозволяє нам краще розуміти, як і чому люди вчиняють в певних ситуаціях, як взаємодіють під впливом конкуренції або співпраці, і як можна знайти оптимальні стратегії для досягнення своїх цілей.

Логічною основою теорії ігор є формалізація трьох понять, які входять в її визначення і є фундаментальними для всієї теорії:

- Конфлікт;
- Прийняття рішення в конфлікті;
- Оптимальність прийнятого рішення.

Ці поняття розглядаються в теорії ігор у найширшому сенсі. Їхні формалізації відповідають змістовним уявленням про відповідні об'єкти.

Змістовно, конфліктом можна вважати будь-яке явище, відносно якого можна казати про його учасників, про їхні дії, про результати явищ, до яких призводять ці дії, про сторони, які так чи інакше зацікавлені в таких наслідках, і про сутність цієї зацікавленості.

Ігри можуть бути класифіковані за різними критеріями. Розглянемо ігри за класифікаціями.

Кооперативні та некооперативні ігри

У кооперативних іграх гравці працюють разом з метою досягнення спільного результату. У некооперативних іграх гравці діють самостійно і конкурують один з одним.

Кооперативні ігри:

a) Будь-яка командна гра, в якій група людей спільно працює над досягненням перемоги в матчі. Найбільш популярними відеоіграми на сьогоднішній день є якраз кооперативні змагальницькі CS:GO, League of Legends, Overwatch тощо.

b) Кооперативні ігри, наприклад Payday 2, It Takes Two, Left 4 Dead, в яких збирається команда 2, 3, 4 і більше гравців, які спільними зусиллями мусять проходити рівень знищуючи ворогів та вирішуючи логічні задачі.

Некооперативні ігри:

a) Шахи. Кожен гравець діє самостійно і намагається перемогти опонента шляхом тактики та стратеґії.

b) Режим королівської битви (battle-royal), найяскравішими представниками якого є Fortnite, PUBG, Apex Legends, в яких велика кількість гравців змагається у шутерному жанрі за виживання, перше місце.

c) Симулятори перегонів, де очевидно кожен гравець хоче прийти на фініш першим.

Симетричні та асиметричні ігри

У симетричних іграх всі гравці мають однаковий набір стратегій і однакові виграші. У асиметричних іграх гравці мають різні набори стратегій і можуть мати різні виграші.

Симетричні ігри:

a) У шахах обидва гравці мають однаковий набір фігур і можливостей. Початкові умови і правила однакові для обох гравців.

b) Прикладом симетричної гри можна навести *метаігри* – спідрани (speedrun). Ціль спідрану не в отриманні тих ресурсів та досягнень, що пропонує гра, а в якнайшвидшому проходжені гри чи рівня гри, коли саме змагання відбувається у спільноті гравців на різноманітних інтернет-формуах.

Асиметричні ігри:

a) З міркувань збереження різноманітності у відеоіграх частіше спостерігаються саме асиметричні випадки, бо часто кожному гравцеві надається можливість обирати ігрового персонажа із різним набором характеристик та особливостей на рівнозначній збалансованій ігровій площині, або ж навпаки, коли ігрові персонажі не відрізняються, але є різниця у балансі та стратеґіях відносно навколишнього середовища, ігрової мапи.

b) Настільні ігри жанру «секретні ролі». Найпопулярнішими представниками в настільних іграх є Мафія та Секретний Гітлер, з відеоігор Among Us. Кожен учасник має свою унікальну роль та протидіє команді суперника (або усим іншим гравцям)

Ігри з нульовою сумою та ненульовою сумою

У грах з нульовою сумою загальний виграш всіх гравців дорівнює нулю. Тобто виграш одного гравця втрачається на користь інших. У грах з ненульовою сумою загальний виграш гравців може бути не нульовим, тобто може бути ситуація, коли один гравець отримує більше, ніж інший гравець втрачає.

Ігри з нульовою сумою:

a) У шахах загальний виграш двох гравців завжди дорівнює нулю. Якщо один гравець перемагає, то інший гравець зазнає втрати в тій же самій мірі.

b) Переважна кількість змагальницьких ігор мають нульову суму, бо є команда переможців та команда (або команди) програвших. Також королівські битви мають нульову суму, бо з багатьох гравців лише 1 перемагає.

Ігри з ненульовою сумою:

a) В деяких іграх з нульовою сумою окрім змагання за перше місце присутня *система винагороди* де виногороду отримує кожен гравець у кількості відносній до місця у рейтингу.

b) Економічні стратегії, де гравці мають можливість отримати більший виграш за рахунок інших гравців. Загальна сума виграшів і втрат може бути ненульовою, проте найчастіше переможцем вважають гравця із найбільшою кількістю ресурсів, відповідно такі ігри можуть бути зведені до ігор із нульовою сумою.

c) MMORPG – жанр ігор, в якому глобально немає окремих ігрових сесій, а кількість завдань та викликів постійно оновлюються, що може утворити ефект нескінченної гри. Найяскравішмй приклад – World of Warcraft. Звісно, ігровий процес в такому жанрі може бути розбитий на сегменти ігор із нульовою сумою, проте глобально навіть встановлення умовного «переможця» є майже неможливий, через велику кількість критеріїв, за якими може встановлюватися успішність, та суб'єктивний погляд кожного гравця на вагу цих критеріїв.

Дискретні та неперервні ігри

У дискретних іграх гравці мають обмежений набір можливих стратегій. У неперервних іграх гравці можуть мати безліч варіантів вибору.

Дискретні ігри:

a) Шахи. У шахах гравці здійснюють хід по черзі 1 з 16 фігур.

b) Покрокові ігри та покрокові стратеґії, де в кожного гравця є обмежена кількість дій за 1 крок та обмежена кількість можливостей за 1 крок.

Неперервні ігри:

a) Екшн ігри, в яких дія та подолання тих чи інших перешкод відбувається в реальному часі.

b) Глобальні стратеґії ворґейми (global strategy wargame), в яких керування юнітами відбувається в реальному часі, та найкращий результат приносить одночасне керування багатьма одиницями юнітів.

Одноразові та повторювані ігри

У одноразових іграх гравці взаємодіють один раз і мають можливість вибрати свої стратегії один раз. У повторюваних іграх гравці мають можливість вибирати свої стратегії у послідовних раундах, що дає можливість аналізувати довгострокову стратегію та взаємодію.

Одноразові ігри:

a) З міркувань збереження різноманітності у відеоіграх підхід одноразових ігор не використовується, бо не може забезпечити достатнього рівня задоволення, проте…

b) …з середини 10-х років популярності почали набувати лутбокси – певний віртуальний об'єкт, який можна викупити за ігрову чи реальну валюту, за що є вірогідність отримати один із внутрішньоігрових предметів різної рідкісності. Гравець отримує винагороду, по суті, за аналого до рулетки чи лотереї, де шанс/цінність виграшу напряму залежить від строго визначених да добре відомих вірогідностей.

Повторювані ігри:

a) Шахи є прикладом повторюваної гри, де гравці можуть зіграти багато партій одні за одним і намагатися покращувати свої навички та стратегії з кожною грою.

b) Абсолютно усі відеоігри є повторюваними, як кооперативні так і однокористувацькі, так і змагальницькі. Якщо у змагальницьких іграх ситуація подібна до шахів, то для однокористувацьких ігор є навіть сформульована властивість – реіграбельність, що характеризує можливість проходити гру знову і знову в різні способи та стратеґії, так чи інакше отримуючи певну винагороду.

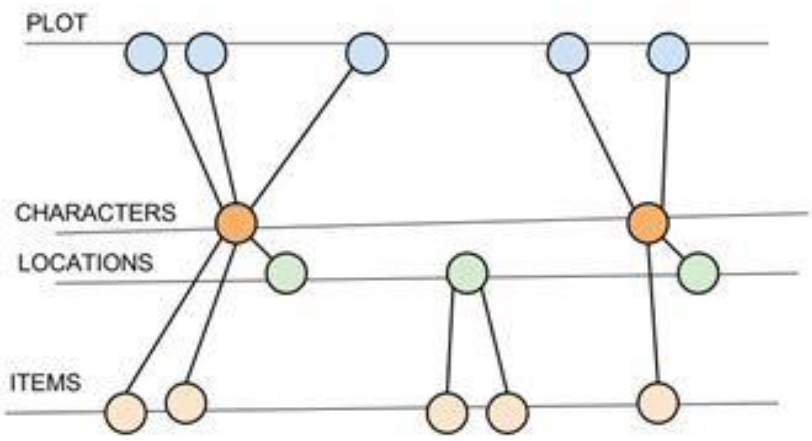
З описаних вище катеґорій можемо зробити наступний висновок: майже кожна відеогра є *повторюваною*, *асиметричною*, та має *нульову суму*, або може бути розділена на сегменти, в яких присутня нульова сума. Відповідно, глобально різницю можна спостерігати за відповідністю до класифікацій *кооперативна чи некооперативна та дискретна чи неперервна.*

2.2. Застосування в об’єднанні з теорією ґрафів

Із описаних вище понять можемо спробуємо встановити зв’язок між двома математичними дисциплінами.

У теорії графів вивчаються взаємозв'язки між вершинами та ребрами графа. Вона досліджує властивості графів, шляхи, цикли, зв'язність, кольорування та інші характеристики графів.

З іншого боку, теорія ігор вивчає стратегічні взаємодії між різними гравцями. Вона аналізує рішення, які гравці приймають в умовах обмежених ресурсів і конкуренції, досліджує різні типи ігор, стратегії гравців.



Мал. 15. Приклад ґрафу, що об'єднує усю структуру гри, а не лише окрему її складову

I. Перший варіант – соціальний – представити вершини як гравців, а ребра як зв’язки між ними. Під таке визначення підпадають лише кооперативні ігри.

II. Другий варіант – економічний – представити вершини як деякий прибуток, збільшення наявних ресурсів, певні цілі; а ребра як вагу рішення, яке треба затвердити для отримання прибутку, набуття ресурсів, досягнення цілі. Найбільш підходящий варіант – ігри з ненульовою сумою

III. Третій варіант – розробницький – представити вершини як певні контрольні точки до загального проєкту, а ребра наділити вагою, що відповідатиме характеристиці затрачених ресурсів та часу. В цьому варіанті застосовується суміш дерева та діаґрами Гассе.

IV. Четвертий варіант – ґрафічний – розробник може використовувати ґрафи для моделювання та відображення геометричних об'єктів, які складають світ гри. Це включає моделювання ландшафту, об'єктів, персонажів та їх взаємодії. Ґрафи можуть використовуватись для визначення з'єднань між об'єктами, розташування об'єктів у просторі та вирішення проблем зі зіткненнями.

Кожен із цих варіантів може бути розглянутий у вигляді окремого ґрафу, та в залежності від поставленої мети ми вже застосовуватимемо той чи інший спосіб отримання найкращого результату.

3. Приклади та перспективи використання описаних структур у системах прийняття рішень при розробці відеоігор

3.1. Приклади задач та існуючі системи

На сьогоднішній день, напевно, усі відеоігри (як і більшість ігор звичайних) у своїй структурі застосовують теорію ґрафів. Системи штучного інтелекту у іграх стратеґіях, розгалудження сюжетних гілок у інтерактивних фільмах чи іграх із нелінійним сюжетом або ж при дизайні ігор-головоломок.

Досить часто персонажем відеогри керує не людина, а штучний інтелект. Поведінка таких персонажів відрізняється від поведінки персонажів якими керує людина. Ігровий штучний інтелект має містити в собі програмні методики, що створюють ілюзію поведінки живої людини. Ці методи включають в себе алґоритми теорії управління, робототехніки, комп'ютерної графіки та інформатики. Поведінка персонажу залежить від основних правил гри, основних принципів та якості процесу штучного міркування. Якість процесу штучного міркування залежить від фактів, що впливають на певну ситуацію та ефективність реалізації.

Головним принципом поведінки персонажа є прийняття рішень щодо подальшої поведінки персонажа. Найяскравіше це реалізується в стратеґіях реального часу, де потрібно аналізувати як поточний стан персонажа, так і повністю стан всієї гри. Прийнявши рішення про те, щоб потрапити до певного об'єкту необхідно оцінити та вибрати оптимальний шлях до цього об'єкту, час, який необхідно витратити на цей шлях та можливі перешкоди. Це може бути реалізовано за допомогою побудови відповідного графу. Аналіз можливих рішень проблеми приводить до вибору методу реалізації подальшої поведінки.

3.2. Перспективи застосування

Наративні елементи в комп'ютерних іграх відіграють важливу роль у створенні захоплюючого та емоційного геймплею. Вони відповідають за розповідь історії, розвиток персонажів, створення атмосфери та передачу гравцеві відчуття присутності у вигаданому світі.

Одним з основних наративних елементів є сюжет. Він визначає хронологію подій, взаємозв'язки між персонажами та розвиток головної історії. Хорошо розроблений сюжет може захопити увагу гравця, викликати емоції та спонукати його до подальшого дослідження гри.

Персонажі також є важливими наративними елементами. Вони мають свої унікальні характеристики, мотивації та історії, що впливають на їхню поведінку у грі. Гравець може взаємодіяти з персонажами, спостерігати за їхніми діями та взаємодіями, а також впливати на подальший розвиток їхніх історій.

Додатково, наративні елементи включають локації та атмосферу гри. Локації визначають місця, де розгортаються події, і можуть бути вигаданими світами, історичними місцями або реальними локаціями. Атмосфера створюється за допомогою музики, звукових ефектів, графіки та дизайну оточуючого середовища, що допомагає поглибити іммерсію гравця в гру.

Наративні елементи в комп'ютерних іграх є важливими для побудови цікавого та незабутнього досвіду геймплею. Вони розширюють можливості гравця, надають значення його діям та сприяють відчуттю присутності у вигаданому світі гри.

Графоцентрична модель обрана для моделювання комп'ютерних ігор через свою здатність ефективно представляти складні залежності та взаємозв'язки між елементами гри. Основні причини використання графоцентричної моделі включають:

Гнучкість: Графи дозволяють представляти складні структури даних, що можуть змінюватися та розвиватися протягом гри. Вони дозволяють описувати

різноманітні взаємозв'язки між персонажами, об'єктами, локаціями та подіями у грі.

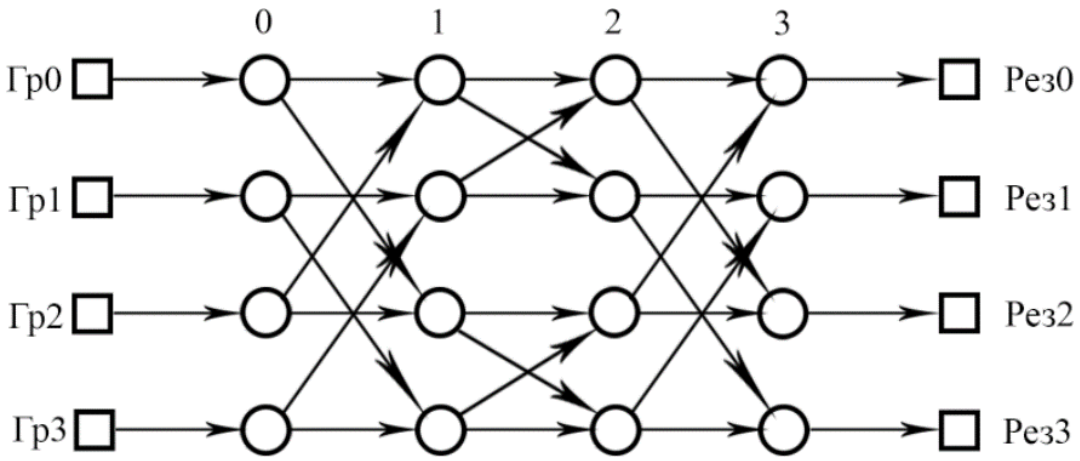
Візуалізація: Графи можуть бути легко візуалізовані, що допомагає розуміти та аналізувати структуру гри. Це дозволяє розробникам та дослідникам легше сприймати та аналізувати взаємодію між елементами гри.

Редагування та модифікація: Графоцентрична модель дозволяє легко вносити зміни до структури гри шляхом додавання, видалення або зміни зв'язків між елементами. Це робить модель гнучкою та легко змінюваною під час розробки ігор.

Аналіз та оптимізація: Графоцентрична модель надає можливість проводити аналіз взаємодії між елементами гри та виявляти можливі проблеми або оптимізаційні можливості. Вона дозволяє виявляти потенційні проблеми з логікою гри, недосяжні об'єкти або непотрібні зв'язки.

Перевикористання: Графоцентрична модель дозволяє перевикористовувати окремі елементи гри, такі як персонажі, об'єкти або локації, в різних сценах або рівнях. Це забезпечує ефективне використання ресурсів та полегшує розширення гри.

Візьмемо за модель ігрового процесу мережу Бенеса. Таким чином ми можемо представити input як гравця, а output як певний результат. Така модель відповідає кооперативній та асиметричній грі. В залежності від мети можемо сприймати модель як гру з нульовою сумою, так і з ненульовою. Дискретність та неперервність залевать від правил гри. Властивість кооперативності залежить від того, чи грають гравці поодинці для досягнення певної мети, чи є в командах одне проти одного, чи взагалі змагаються однією командою проти самої гри, алгоритму її зміни та поведінки.



Мал. 16. Приклад моделі гри у вигляді мережі-метелика Бенеса

Ігровий процес може мати безліч варіацій в такому варіанті. Ця модель обрана виходячи зі значення заторовості 1. В ігровому процесі ми все ще маємо умовну нелінійність (2 вибори) і зменшуємо шанси на те, що гравці заважатимуть одне одному обираючи той, чи інший варіант прокладення шляху.

Один із варіантів гри заснованого на цій моделі полягатиме в проходженні принаймні 1 гравцем до Результату, наприклад, за відведений час, або за обмежену кількість ходів, що залежатиме від характеристики дискретності чи неперервності гри. Тоді вважатиметься, що команда гравців перемогла.

Кожен Гравець має обрати свій вхід, інакшими словами – свого персонажа, що матиме певні характеристики a, b, c ... (що можуть бути як встановлені за замовчуванням, так і можуть бути встановлені учасником), а кожен перемикач матиме умови для входу, що є відомими, та дії над значеннями характеристик Гравця, що є невідомими. Тут постає причина застосовувати теорію ігор для моделювання приємного ігрового процесу.

Залучивши матрицю виграшів в даному прикладі ми отримаємо 4-вимірну матрицю, що може бути складно для опису на папері, проте вкрай просто для описання у програмному коді.

	Гравець 2 приймає стратегію А	Гравець 2 приймає стратегію В
Гравець 1 приймає стратегію А	4 / 4	1 / 3
Гравець 1 приймає стратегію В	3 / 1	2 / 2

Мал. 17. Приклад двовимірної матриці виграшів для 2 гравців та 2 стратегій

Спроєктувавши таку структуру, та маючи характеристики a, b, c ... кожного Гравця, ми змінюємо поведінку перемикачів (тобто як вони впливають на характеристики персонажа) та будемо бачити, які саме дії перемикачів стосовно характеристик персонажа заводитимуть команду в глухий кут. Це той варіант розвитку подій, якого кожен розробник гри бажає уникати. Кожен розробник хоче лишати певний сенс для гравців у своїй грі. Тому такі варіації відкидатимуться на самому етапі моделювання.

Інший варіант – коли Гравцям необхідно самостійно встановлювати характеристики a, b, c ..., проте умови входу до перемикача теж невідомі. В такому випадку потрібно провести симуляцію 4-вимірної матриці і проаналізувати результати. А результати нас можуть цікавити різні. Наприклад, результати кількості успішно пройдених партій, тривалість партій. Або можемо розглянути варіант, в якому результат партії є тимкращий, чим більша кількість Гравців досягла Результату. В такому випадку нас цікавитимуть також дані проходження різної кількості Гравців залежно від обраних характеристик персонажів та значень перемикачів.

Висновки

Як можемо бачити – теорія ґрафів у поєднанні з теорією ігор можуть бути вкрай важливим застосунком не лише у технічній складовій проєкту, а і у дизайнерській. При чому у безлічі варіантів проєктів, що можна концептуально утворити лише перестановками загальних класифікацій ігор, для усих яких ми вже маємо стратеґії вирішення, згідно теорії ігор. Маючи ідею ігрового процесу у вигляду абстракції, можна алґоритмічно знаходити найбільш приємні його для Гравців варіанти.

До загальних недоліків використання теорії графів можна віднести роботу тільки з структурами, що містять графи з невід’ємною довжиною дуг, пошук найкоротшого шляху тільки від однієї вершини, для реалізації методу необхідні повтори алгоритму для кожної вершини, не шукають рефлексивні замикання та мають досить складну в порівнянні з матричними методами програмну реалізацію. До переваг методу можна віднести простоту реалізації, невисоку поліноміальну обчислювальну складність, можливість машинної реалізації, знаходження розв’язку задачі, якщо такий існує.

Джерела

1. Iwona Grabska-Gradzińska, Bartosz Porębski, Wojciech Palacz, Ewa Grabska // Towards a Graph-Based Model of Computer Games – Proc. of Int. Conf. on Advances in Communication and Information Technology 2012 – 6 с.
2. Теорія графів. – Київ: КПІ ім. Ігоря Сікорського, 2020. – 71 с.
3. MIT 6.042J Mathematics for Computer Science [Електронний ресурс]. – Режим доступу: https://youtube.com/playlist?list=PL6MpDZWD2gTF3mz26HSufmsIO-COKKb5j
4. Теорія ігор. – Київ: КПІ ім. Ігоря Сікорського, 2022. – 245 с.