АЛГОРИТМ*A**ДЛЯ ПОШУКУ НАЙКОРОТШОГО ШЛЯХУ МІЖ ТОЧКАМИ НА МАПІ ПЛАТФОРМЕРНОЇ ГРИ

Горчинський Олексій Олександрович
д.т.н., професор каф. ІС Арсірій Олена Олександрівна
Національний університет«Одеська політехніка»,УКРАЇНА

АНОТАЦІЯ. Показано, що при розробці платформної гри часто доводиться вирішувати задачу пошуку найкоротшого шляху між двома точками на мапі. Запропоновано вирішення цієї задачі з використанням алгоритму *A**, який реалізовано за допомогою бібліотеки *Pygame*. Наведено особливості *A** в порівнянні з алгоритмом Дейкстри, а також показані результати застосування *A** в платформерній грі.

Вступ. Пошук шляху є фундаментальною проблемою в ігровій індустрії та в багатьох сферах загалом. Використання правильного алгоритму пошуку шляху в грі може оптимізувати процес гри,зменшити навантаження на систему та забезпечити більш якісний геймплей.Один із найефективніших алгоритмів пошуку шляху - *A**, зазвичай використовується у багатьох відеоіграх та симуляційних додатках для пошуку найкоротшого шляху між двома точками. *A** забезпечує оптимальність та ефективність пошуку шляху завдяки використання евристичних оцінок вартості переходу від поточної точки до кінцевої точки. Дані переваги *A** дозволяють ефективно використовувати його в різноманітних відеоіграх, від класичних платформерів до складних стратегічних ігор з відкритим світом. Крім того, *A** може бути використаний для розв'язання проблеми навігації в багатьох інших областях, таких як робототехніка, планування маршруту, тощо.

Мета роботи. Метою роботи є реалізація ефективного пошуку найкоротшого шляху між двома точками на мапі платформної гри за допомогою алгоритму *A**.

Основна частина роботи. Для вирішення задачі пошуку найкоротшого шляху при створенні платформної грі було використано алгоритм *A**(«*A-star*») – алгоритм обходу графа та пошуку шляхів, який використовується в багатьох галузях інформатики. Найчастіше *A** використовується в розробці ігор. Автори Пітер Харт, Нільс Нільссон і Бертрам Рафаель розглядали його як розширення алгоритму Дейкстри.

*A** досягає кращої продуктивності, використовуючи евристику для керування пошуком. Порівняно з алгоритмом Дейкстри, алгоритм *A** знаходить лише найкоротший шлях від зазначеного джерела до вказаної цілі, а не дерево найкоротших шляхів від зазначеного джерела до всіх можливих цілей.

Алгоритм *A** обчислює вартість для всіх сусідніх вершин і обирає вершинуз мінімальною вартістю. Цей процес повторюється до тих пір, поки не буде неможливим вибір нових сусідніх вершин.Після цього ми розглядаємо сусідню вершину з найкращою вартістю.Вартість вершини обчислюється як:

$$f(n) = g(n) + h(n), \quad (1)$$

де $f(n)$ – загальна вартість проходження через вершину n, $g(n)$ – вартість переходу від однієї вершини до іншої, $h(n)$ – евристична апроксимація значення вершини [1].

Алгоритм A* може підтримувати різні еврестичні функції, для підрахування $h(n)$, наприклад:

1. Манхеттенська відстань – ця евристика використовується, коли можливо рухатися лише в 4 напрямках (вгору, вниз, ліворуч, праворуч) у двомірній сітці.

2. Відстань Чебишова – ця евристика використовується для 8-стороннього руху, коли додається ще діагональне переміщення у двомірній сітці.

1. Евклідова відстань – ця евристика використовується, коли можливо рухатися в будь-якому напрямку. [2]

Наведемо приклад пошуку найкоротчего шляху для графа на рисунку 1, де *N*1 – це початкова вершина, а *N4* – пункт призначення.Початок знаходиться в вершині *N1*,яка має *g* =0 і деяке початкове евристичне значення *h*, наприклад *h* дорівнює 10. Тоді згідно з (1)

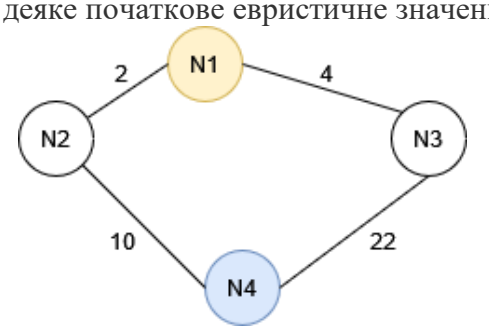

Рисунок1–Приклад графу

f(N1)= g(N1)+ h(N1)=> f(N1) =0 +10 =10.

Розрахуємо шлях до сусідніх вершин:

f(N1->N2)= g(N1->N2)+ h(N1->N2) = 2 +5 =7

f(N1->N3)= g(N1->N3)+ h(N1->N3) = 4 +6 =10

Розрахуємо шлях до місця призначення:

f(N1->N2->N4) = g(N1->N2->N4) + h(N1->N2->N4) = 2 + 10 + 3 = 15

f(N1->N3->N4) = g(N1->N3->N4) + h(N1->N3->N4) = 4 + 22 + 5 = 31

Бачимо, що вибір вершини *N2* з *N1* дає найкращий шлях. Зауважимо, що алгоритм Дейкстри є окремим випадком алгоритму пошуку*A**,де*h*=0для всіх вершин.Тобто відмінність полягає в тому, що алгоритм *A** намагається знайти кращий шлях за допомогою евристичної функції, віддаючи пріоритет вершинам, які мають кращий результат евристичного розрахунку, тоді як алгоритм Дейкстри досліджує всі можливі шляхи.

Таким чином,в алгоритмі*A**замість перевірки*g(n)*для поточної вершини,як у Дейкстри, використовуємо (1) для перевірки поточної вершини. Тобто згідно з *A** ми не відвідуємо всі вершини,а починаємо з початкової вершини,а потім перевіряємо всіх сусідів цієї вершини,потім для кожного сусіда додаємо його значення *g* (відстань від початкової вершини до цього сусіда), потім додаємо значення еврестичної функції *h(n)* цього сусіда.Далі всі ці сусіди додаються до черги пріоритетів відповідно до їхніх значень f(n). Ми використовуємо чергу пріоритетів, щоб шивдше отримати найменше значення. Вершини в черзі пріоритету тепер відкрито для обчислення, а початкову вершину закрито для обчислення (додано до словника відвіданих вершин).Потім ми переходимо до вершини з найменшим значенням *f* (якщо пункт призначення ще не досягнуто), а потім відвідуємо його, лише якщо цієї вершини ще не має доступного коротшого шляху. Якщо до цієї вершини вже є коротший шлях, то поточний шлях не є найкоротшим, і, отже, ми не розширюємо його сусідів, і ми можемо повернутися до наступного найкоротшого шляху в черзі пріоритету. Ми повторюємо це, поки не досягнемо кінцевої вершини. Коли ми досягаємо мети, ми знаходимо найкоротший шлях.

Згідно наведеного опису алгоритм*A**реалізовано з використанням бібліотеки *Pygame* при розробці платформної гри «*Algorithmic Christmas*».

Розроблена гра має декілька режимів:навчальний і аркадний.В арканному режимі гравець і робот повинні зібрати подарунок кожен раунд швидше за один одного.Робот знаходить шляхи до подарунку на ігровій мапі завдяки реалізованого і розглянутого вище алгоритму пошуку шляху *A**. Якщо робот або гравець збирає подарунок швидше то збільшується кількість їх балів відповідно. Кожен раунд гра починається заново з того самого місця але подарунок з’являється випадковим чином.

Навчальний режим відображає додаткову інформацію о ігровій мапі і роботі алгоритму. В ньому гра сповільнена. Оранжеві цифри відповідають за координати на ігровому полі, жовті відповідають за вартість кожної клітинки, яку обробляє алгоритм. Червоні кружечки відображаюь відвідані алгоритмом вершини,сині кружечки відображають найкоротший шлях,за яким йде робот, а білий кружечек відображає початкову клітинку (рис. 2).

Що стосується обчислювальної складності алгоритм *A** – *O* «велике»,то у найгіршому випадку алгоритм *A** проходить через усі ребра графу, щоб досягти пункту призначення від початкової вершини, тобто *O*(*E*) – лінійно залежить від *E* – кількості ребер у графі.

Рисунок2–Вигляд навчального режиму гри

У найгіршому випадку всі ребра будуть всередині словнику відвіданих вершин, тому необхідний додатковий простір у гіршому випадку дорівнює $O(V)$, де V – загальна кількість вершин.

Висновки. Таким чином, алгоритм *A** є більш ефективним, ніж алгоритм Дейкстри. Використовуючи евристичну функцію для надання оцінки вартості шляху між кожною вершиною і кінцевою вершиною, він може зробити кращий вибір щодо обрання наступної вершини та швидше знайти найкоротший шлях. При цьому найбільшою проблемою при використанні*A**є обрання«ефективної»евристичній функції.В такому випадку економія часу в процесі пошуку шляху буде помітною в порівнянні з класичним алгоритмом Дейкстри. Перевагою алгоритму Дейкстри є те, що в будь-якому випадку в результаті буде знайдено найкоротший шлях не тільки між початковою і кінцевою вершиною, але також між початковою вершиною і кожною іншою на графі.Однак алгоритм може бути неефективним,оскільки він «втрачатиме час»на оцінку маршрутів,які можна проігнорувати.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. A* search algorithm. Wikipedia: веб-сайт. URL: https://en.wikipedia.org/wiki/A*_search_algorithm (дата звернення 06.05.2023).
2. 3 distances that every data scientist should know. Towards Data Science: веб-сайт. URL: https://towardsdatascience.com/3-distances-that-every-data-scientist-should-know-59d864e5030a (дата звернення 06.05.2023).
3. XiaoCui,HaoShi«A*-based pathfinding in modern computer games».InternationalJournal of Computer Science and Network Security. 2011; Vol.11 No.1:p.125-130. URL: https://www.researchgate.net/publication/267809499_A-based_Pathfinding_in_Modern_Computer_Games

FINDING THE SHORTEST PATH BETWEEN POINTS ON A PLATFORMER GAME MAP USING THE A* ALGORITHM

Oleksii Horchynskyi

DrSc,Professor of the department of IS Olena Arsirii

Odessa Polytechnic National University, UKRAINE

ANNOTATION. It is known that development of a platformer game often faces the problem of finding the shortest path between two points on the map. The solution to this problem is proposed using the A* algorithm implemented using the Pygamelibrary. The features of A* are presented in comparison with Dijkstra algorithm, and the effectiveness of using A* in a platformer game is also shown.