Покращення евристичного алгоритму A* згладжуванням кривої маршруту з використанням точного методу Дейкстри на локальних проміжках

В. С. АКСЬОНОВ, Т. Ж. НАДРИГАЙЛО

Дніпродзержинський державній технічний університет

В данной работе предложена модификация метода A* для поиска кратчайшего пути на карте дорог – метод A#. Разработан программный комплекс, реализующий метод A#, который может служить программным обеспечением для устройств с модулем GPS на базе операционной системы Android.

У даній роботі запропонована модифікація методу A* для пошуку найкоротшого шляху на карті доріг – метод A#. Розроблено програмний комплекс, що реалізує метод A#, який може служити програмним забезпеченням для пристроїв з модулем GPS на базі операційної системи Android.

In this paper we propose a modification of the method A* to find the shortest path on the road map – method A#. Developed a software package that implements the method A# which can serve as software for devices with GPS-based operating system Android.

Вступ. Задача пошуку найкоротшого маршруту є досить поширеною та важливою у багатьох сферах життя і виробництва. Для розв'язку цієї задачі створена велика кількість методів умовно розподілених на декілька груп. Для кожної задачі більш за все підходять методи тільки однієї із груп.

Такими групами є: точні методи, жадібні методи, перший–ліпший, ймовірнісні методи, евристичні методи [1].

Для випадків, коли має значення лише точність і доступна велика кількість ресурсів та часу, найкраще використовувати точні методи, такі як метод Дейкстри [2].

Але бувають і інші випадки. Хорошим прикладом є навігаційна система у автомобільному навігаторі або телефоні, що зараз досить розповсюджена. Водій потребує можливості швидко знайти маршрут до місця призначення. У такому випадку на допомогу прийдуть наближені методи. Найбільш поширеним серед навігаційного програмного забезпечення є евристичний метод A*.

Однак жоден метод не є ідеальним і існують початкові умови які призводять до того, що метод A* видає маршрути, слідуючи яким водію доведеться зробити "гак".

В даній роботі проаналізовано причини виникнення таких збоїв в роботі методу A* та обрано спосіб їх виправлення. Таким чином створена модифікація методу A*, що позбавлена цієї особливості. Ця модифікація отримала назву A#. Також створено систему, яка реалізує цей метод. Для цього розроблена зв'язка із клієнтської програми для операційної системи Android та серверної реалізації алгоритму.

В якості джерела картографічної інформації використовується відкрита для користування та публічного наповнення карта OpenStreetMap. Інформація про стан дорожнього покриття отримується автоматично під час збору анонімної статистики з навігаційних пристроїв. Стан покриття характеризує середня швидкість з якою водії пересуваються на певній ділянці дороги. На базі цих даних на сервері методом A# будується маршрут а клієнтська програма забезпечує його відображення та супроводження водія при їзді.

Постановка задачі. Алгоритм A* – один з найпоширеніших алгоритмів пошуку маршруту на картах доріг (топографічних графах). Він використовується в багатьох системах як прикладного напрямку (навігаційні системи, ГІС) так і розважальних (ігри, геоквести).

Кожному розробнику, якому доводилося реалізовувати алгоритм A* у свої системі, доводилося стикатися з такою особливістю методу як виникнення дуг, "гаків". В різних системах ця особливість або ігнорується, або виправляється своїм власним способом.

Таким чином, виникла необхідність проаналізувати метод A*, визначити причину виникнення вказаної особливості, розробити та сформулювати метод, що виправляє похибку методу A*. На основі отриманої модифікації методу розробити програмний комплекс для пристроїв з модулем GPS на базі операційної системи Android.

Результати роботи. Для роботи методу A# знадобиться перераховувати деякі ділянки маршруту точним методом Дейкстри.

Метод Дейкстри не використовується як самостійний модуль для пошуку маршруту, а лише як підпрограма, що викликається під час пошуку маршруту методом A#.

Цей метод полягає у наступному: для знаходження найкоротшого маршруту від вузла u_s до вузла u_f слід перебрати всі можливі маршрути між цими точками.

Алгоритм методу:

1) встановлюємо відстань від початкового вузла до вузла u : $G(u_s) = 0$, та вибираємо вершину u_c як поточну ($u_c = u_s$);

2) для $\forall u \in \nu(u_c)$ та $u \notin l$ визначаємо:

$$G(u) = \mathrm{Len}(\rho(u, u_c)) + G(u_c)$$

та додаємо вершини u до відкритого списку L;

3) включаємо u_c до закритого списку l_c, видаляємо u_c із відкритого списку L_c;

4) якщо $L = \varnothing$ – робота алгоритму закінчена, якщо ні – вибираємо нову вершину u_c таку, що $G(u_c) = \min_{u \in L} G(u)$ і переходимо до пункту 2.

Алгоритм визначений на графах з ребрами додатної довжини [3–5].

Метод A* є основою методу A# і розв'язання задачі цим методом співпадає з першою фазою методу A#.

Метод A* одночасно виступає як база для методу A# і як еталон, з яким будемо порівнювати метод A#.

Цей метод належить до наближених методів евристичної групи. Ця група містить методи, що "здогадуються", "роблять припущення" яка дія є найкращою на даному кроці.

На відміну від методу Дейкстри, що відвідує всі вершини, алгоритм A* в першу чергу відвідує вершини, що ймовірно належать до оптимального шляху. Такі вершини знаходяться за допомогою знаходження для кожної вершини графа значення величини $F(x)$:

$$F(u) = H(u) + G(u),$$

де $G(u)$ – довжина шляху від початку до цієї вершини;

$H(u)$ – евристична функція оцінки довжини шляху від цієї вершини до фінальної вершини.

Алгоритм методу:

1) встановлюємо $G(u_s) = 0$ та додаємо вузол u_s у відкритий список L;

2) вибираємо вузол u_c такий, що

$$F(u_c) = \min_{u \in L} F(u);$$

3) для $\forall u \in \nu(u_c)$ та $u \notin l$ визначаємо

$$G(u) = \mathrm{Len}(\rho(u, u_c)) + G(u_c)$$

$$F(u) = H(u) + G(u)$$

та додаємо вершини u до відкритого списку L;

4) включаємо u_c до закритого списку l, видаляємо u_c із відкритого списку L;

5) якщо $u_c = u_f$ – робота алгоритму закінчена, інакше – переходимо до пункту 2.

Евристичний метод A#. Для останньої фази алгоритму також необхідно знати вектори, що поєднують послідовно всі вузли маршруту, вектор, що поєднує перший та останній вузли маршруту та кути відхилення векторів від базового. Для методу A# було обрано підхід із згладжуванням маршруту вже після його знаходження.

Алгоритм методу: перша частина методу повністю повторює хід базового методу, відмінність виникає в умові закінчення пошуку:

1) встановлюємо $G(u_s) = 0$ та додаємо вузол u_s у відкритий список L;

2) вибираємо вузол u_c такий, що

$$F(u_c) = \min_{u \in L} F(u);$$

3) для $\forall u \in \nu(u_c)$ та $u \notin l$ визначаємо

$$G(u) = \mathrm{Len}(\rho(u, u_c)) + G(u_c)$$

$$F(u) = H(u) + G(u)$$

та додаємо вершини u до відкритого списку L;

4) включаємо u_c до закритого списку l, видаляємо u_c із відкритого списку L;

5) якщо $u_c = u_f$ – закінчена перша фаза методу, інакше – переходимо до пункту 2;

6) після першої фази алгоритму всі вузли, що належать до знайденого маршруту, знаходяться у закритому списку. Для їх знаходження необхідно пройти від фінальної вершини назад, обираючи поміж сусідів тих, хто має найменше значення $G(u)$. Отриманий маршрут позначимо як μ;

7) для $\forall \overline{\upsilon}_i \in \mu$ перевіряємо умови:

$$\arccos(\overline{\upsilon}_i, \overline{\upsilon}_{base}) > \alpha_{max} \quad (1)$$

$$\arccos(\overline{\upsilon}_i, \overline{\upsilon}_{base}) < -\alpha_{max} \quad (2)$$

8) якщо умова 1 або умова 2 виконується, вважаємо, що у вузлі u_i починається дуга і переходимо до пункту 9, інакше – повертаємося до пункту 7;

9) для $\forall \overline{\upsilon}'_{i,j}$ перевіряємо:

$$\arccos(\overline{\upsilon}'_{i,j}, \overline{\upsilon}_{base}) > \alpha_{max} \quad (3)$$

$$\arccos(\overline{\upsilon}'_{i,j}, \overline{\upsilon}_{base}) < -\alpha_{max} \quad (4)$$

$$\arccos(\overline{\upsilon}'_{i,j}, \overline{\upsilon}_{base}) < \alpha_{max} \quad (5)$$

$$\arccos(\overline{\upsilon}'_{i,j}, \overline{\upsilon}_{base}) > -\alpha_{max} \quad (6)$$

10) якщо для $\overline{\upsilon}'_{i,j-1}$ виконується умова 3 або 4 а $\overline{\upsilon}'_{i,j}$ задовольняє умовам 5 та 6 то вважаємо, що дуга закінчилася вузлом u_j;

11) на множині $U'(\mu(i,j))$ проводимо пошук методом Дейкстри, покладаючи вузол u_i початковим, а вузол u_j кінцевим. Отриманим маршрутом $\mu^*(i,j)$ заміняємо фрагмент $\mu(i,j)$, переходимо до пункту 7 продовжуючи виконання алгоритму з вузла u_j.

Порівняння алгоритмів. Важливі показники, що визначають якість та швидкість методів, наведено в таблиці 1

Таблиця 1. Показники методів

Методи:	Дейкстри	A*	A#
Довжина маршруту	83,39	89,84	83,39
Кількість кроків	24	7	24
Кількість вузлів, що не відвідані	0	17	12

Як бачимо, методу Дейкстри та методу A# знадобилося по 24 кроки для того, щоб знайти справді найкоротший маршрут.

Швидшим виявився метод A*, хоча це і не було неочікуваністю. Однак, ціною швидкості стала похибка близько 6,5 одиниць. Ця похибка складає 7,73%

Слід зазначити, що хоча у даному прикладі швидкість методу Дейкстри та методу A# не відрізняються, при збільшенні кількості вузлів, кількість кроків методу Дейкстри буде рости разом з нею, в той час як методу A# – разом з кількістю вузлів включених до маршруту. Це досягається завдяки тому, що метод Дейкстри відвідує всі вузли графа, в той час як методи A* та A# пропускають деякі вузли, що виглядають безперспективними.

Із загальної кількості вузлів (23 вузла) метод A* проігнорував 73% вузлів а метод A# – 52%. І знову слід звернути увагу на той факт, що доля проігнорованих

вузлів зростатиме зі зростанням загальної кількості вузлів.

Навігаційна система на базі методу A#. Для перевірки методу було створено комплекс програмного забезпечення, що реалізує функціонал навігаційної системи для пристроїв на базі операційної системи Android версії 2.1 та новіше.

До комплексу також входить програма мовою PHP, що виконує функцію пошуку найкоротших маршрутів методом A# на сервері.

Клієнтська програма для операційної системи Android. Клієнтське програмне забезпечення має наступні функції:

— *Відображення карти доріг*. В якості джерела картографічної інформації було обрано доступну для загального користування карту проекту OpenStreetMaps. Ці карти покривають всю площу планети Земля, знаходяться у вільному доступі та активно розвиваються і доповнюються всесвітньою спільнотою картографів та простих користувачів Інтернету. Використання карт цього проекту дозволяє мати завжди актуальні дані про стан доріг та об'єктів поряд з дорогами.

— *Відображення POI* (*point of interests, "цікавих точок"*). POI використовуються для позначення різних об'єктів, що можуть зацікавити чи знадобитися людині, що подорожує. Це можуть бути як заправки, станції технічного обслуговування так і будинки, кафетерії, пам'ятники, музеї та інші об'єкти.

— *Пошук POI* (*point of interests, "цікавих точок"*). Є можливість знайти POI певного типу поруч з поточним місцезнаходженням користувача, знайти певну адресу, знайти місце на карті за координатами.

— *Визначення адреси чи координат* певного місця на карті. Ця функція дозволяє знайти адресу та поштовий індекс чи координати певного місці на карті.

— *Шукати маршрути використовуючи метод A#*. Для пошуку маршруту розроблений зручний інтерфейс, що дозволяю шукати маршрут між поточним місцеположенням користувача та певною точкою на карті (місцем призначення) чи адресою, POI, координатами. Або можна прокласти маршрут між двома довільними точками на карті, якщо точка планованого початку маршруту не співпадає з поточною позицією користувача.

Маршрут прокладається засобами серверної програми, що отримує запит з точками початку та кінця маршруту. Після отримання відповіді від серверу клієнтська програма малює маршрут і, якщо було обрано пошук маршруту з поточної локації, забезпечує ведення користувача до місця призначення.

— М*ожливість відображення карти засобами сторонніх серверів*, таких як генератор карт Mapnik спільноти OpenStreetMaps або супутникові знімки компанії Google та Microsoft. Для поточної ділянки карти, що переглядає користувач, з обраного серверу завантажується зображення. Цей режим, як і пошук маршруту, потребує підключення до Інтернету або ділянки мають бути переглянуті заздалегідь і збережені в пам'яті пристрою. Вибір таких серверів досить широкий завдяки API серверів OpenStreetMaps, що дозволяють отримати список доступних сервісів з необхідними параметрами.

— *Профілі* дозволяють вказувати поточний спосіб пересування і відповідно до цього враховувати чи не враховувати ті чи інші шляхи (тропи, велосипедні доріжки) при прокладанні маршруту.

Серверне програмне забезпечення. Для пошуку маршруту використовується серверне програмне забезпечення, яке написано мовою PHP.

Це програмне забезпечення отримує запити від клієнтських програм з початковими та кінцевими координатами та режимом пересування (авто, велосипед, пішохід). Далі відбувається запит до бази даних OpenStreetMaps, яким дістаються з баз всі необхідні вузли та ребра. На базі цієї інформації, методом A#, проводиться пошук маршруту. Останнім етапом клієнту повертається масив з точками та ребрами, що належать до отриманого маршруту.

Висновки. В даній роботі розглянуто точний метод Дейкстри та евристичний метод A* для пошуку найкоротшого шляху на карті доріг (топографічному графі). Аналіз методу A* показав, що цей метод дає викиди в деяких випадках, що виражаються у зайвих дугах в маршруті.

В результаті аналізу причин виникнення викидів, створена модифікація методу A*, яка зменшує або виправляє похибку методу за рахунок незначного зменшення швидкості – метод A#.

Розроблено програмний комплекс, що реалізує метод A#, який може служити навігаційним програмним забезпеченням для пристроїв з модулем GPS на базі операційної системи Android. Цей комплекс дозволяє шукати маршрути, переглядати карту та супутникові знімки по всій площі Земної поверхні.

Програмний комплекс може використовуватися в багатьох галузях таких як: логістика та перевезення вантажу, туризм, велотуризм, повсякденна навігація у легкових автомобілях, таке інше.

Недоліком програмного комплексу є необхідність доступу до мережі Інтернет для прокладання маршруту. Але це дозволяє шукати маршрути між точками віддаленими одна від одної на необмежену відстань та на пристроях будь-якої апаратної конфігурації.

Комплекс впроваджено в компанії "АгроФосПолтава" на вантажних автомобілях з метою покращення логістики перевезень, зменшення витрат пального та амортизації автомобільного парку.

ЛІТЕРАТУРА

1. Левитин А. Алгоритмы: введение в разработку и анализ. Москва. Williams. 2006. — 576 с.
2. Ахо А., Хопкрофт Дж., Ульман Дж. Построение и анализ вычислительных методов. — Москва, "Мир". 1979. — 534 с.
3. Асанов М. О., Баранский В. А., Расин В. В. Дискретная математика: графы, матроиды, алгоритмы. — Ижевск. НІЦ "РХД". 2001. — 288 с.
4. Березина Л. Ю. Графы и их применение. — Москва. "Просвещение". 1979. — 144 с.
5. Евстигнеев В. А. Применение теории графов в программировании. Москва. "Наука". 1985. — 352 с.

пост.12.06.14