УДК004.021

СИСТЕМА ПОБУДОВИ ОПТИМАЛЬНИХ МАРШРУТІВ НА ОСНОВІ АЛГОРИТМІВ НАЙКОРОТШИХ ШЛЯХІВ

Яковенко А.В., доцент
yakovenkoalena@ukr.net
Фолькін М.В., студент
markthefolkin@gmail.com
Факультет біомедичної інженерії
Національний технічний університет
«Київський політехнічний інститут імені Ігоря Сікорського»
м. Київ, Україна

Реферат *– На сьогодні з великою різноманітністю транспортних засобів, люди часто не можуть визначити найкращі шляхи пересування, це все спричинено надлишком інформації, яку людині важко запам'ятати. Тому проблема створення маршрутів, за допомогою яких людина змогла б в найкоротший термін дістатися до запланованої цілі постає в найбільшому вигляді. Багато сучасних сервісів дозволяють будувати маршрути від точки А до точки Б, але вони не можуть задовольнити користувача, якому необхідно об'їхати декілька адрес. Для розробки було обрано середовище програмування Eclipse, бібліотеку зв'язку з картографічним сервісом GmapsFX, реалізовану на мові Java. В результаті розроблено програму для пошуку найкоротшого маршруту на карті з використанням алгоритмів оптимальних шляхів, розроблено функціонал сімейного лікаря: перегляд пацієнтів, історія викликів.*

Ключові слова *– алгоритми точного та неточного пошуку, алгоритм Дейкстри, алгоритм A*, задача комівояжера, javafx, GoogleMaps API, дискретна оптимізація, безперервна оптимізація.*

I. ВСТУП

Задача оптимізації полягає в знаходженні найкращого варіанта серед представлених альтернатив. Для того, щоб вибрати оптимальне рішення необхідно визначити залежну величину (функцію). Така функція називається цільова функція або критерій якості. В залежності від задачі шукане значення цільової функції може вибиратись мінімальним (час транспортування пошти, вартість операції) або максимальним (потужність установи, об'єм пам'яті). В залежності від кількості альтернатив, задачі оптимізації поділяються на: дискретну (комбінаторну) та динамічну (безперервну) [1].

Динамічна оптимізація є більш комплексною задачею в порівнянні з комбінаторною. Для нас важливим є не тільки положення критерію оптимальності в певний момент часу, а й його передісторія. Основними елементами динамічної оптимізації є результат, який ми бажаємо отримати, потім певні зміни, які приводять до заданого результату та обмеження, які накладаються на динамічні зміни.

Динамічна оптимізація складається з двох підходів:

1. Згори-вниз – спочатку створюється певний макет (абстракція), потім головні функції і вже в кінці функції останнього рівня. Такий підхід використовується, якщо оптимізація найвищих рівнів дозволить рекурсивно оптимізувати нижчі рівні.
2. Знизу-вгору – вся система розбивається на підсистеми, і так до того часу, поки кожна підсистема не матиме своєї підсистеми (функції). Далі робиться оптимізація кожної функції, які складаються в систему і так до найвищого рівня. Даний підхід є більш зрозумілим для людини, оскільки ми звикли, ділити роботу на певні частини [2].

В комбінаторній оптимізації основним завданням є знаходження кращої комбінації змінних, які приймають тільки дискретні значення. Для більшої кількості завдань рішення розглядається, як розміщення певного набору об'єктів дискретного значення в відповідності до заданих обмежень. Головним мінусом комбінаторної оптимізації є те, що повний перебір даних не є можливим, через те, що на його виконання підуть роки та десятки років. Більшість таких задач називається NP-повні, вони описуються поліноміальною функцією масштабу, тобто найгіршим часом виконання певного

алгоритму.

До завдань дискретної оптимізації, зазвичай, відносять такі завдання:

1. Задача про ранець – NP-повна задача. Основна ціль: вкласти як найбільшу кількість цінних речей при умові, що місткість обмежена.
2. Задача розкрою – основна ідея: поділити певну дискретну величину, на частини різної величини з мінімальними відходами.
3. Мінімальне кістякове дерево – пошук найменшого кістяка графа, який має можливість проходження графа від одного вузла до іншого напряму чи через інші вузли.
4. Задача комівояжера – найбільш відома задача комбінаторної оптимізації. Мається на увазі пошук найбільш вигідного маршруту між містами, який буде проходити через кожне місто хоча б один раз з наступним поверненням в початкове місто. Наше завдання є саме задачею комівояжера [3].

При вирішені задачі комівояжера головними критеріями є вказування вигідності маршруту, це такі значення, як найбільш дешевий, найменший за протяжністю та інші. Та також необхідно вказувати відповідну матрицю відстаней (вартостей), ця матриця показує скільки необхідно витратити ресурсів на пересування з точки А до точки Б.

В симетричній задачі комівояжера цільова функція між тими ж вершинами є однаковою, тобто вартість переходу з вершини А до вершини Б, дорівнює вартості переході з Б до А. Симетрична задача комівояжера не є використовуваною в обраній задачі, тому що, зазвичай, вартість проїзду між різними дорогами відрізняється, це зумовлено появою різних перешкод, використанням на певній ділянці платних доріг.

Асиметрична задача комівояжера використовує різні значення критеріїв оцінювання між однаковими вершинами, це дозволяє обирати оптимальні шляхи в прикладних задачах [4].

Метрична задача комівояжера виникає тоді, коли відносно ребер, які мають вагу (відстань) починає виконуватись нерівність трикутника, тобто в нашій задачі це показує те, що прямий перехід з вершини А до вершини Б буде завжди менше, ніж перехід спочатку до тимчасової вершини Т.

Задача комівояжера є алгоритмічно складною, тому що при кількості вершин після 10 кількість маршрутів збільшується експоненціально, наприклад кількість маршрутів для 15 міст дорівнює 15! = 1,307,674,368,000 [5].

В симетричній задачі комівояжера зустрічається (n-1)!/2 оптимальних маршрутів, в свою чергу для асиметричної – (n-1)!, це спричинено тим, що в симетричній задачі комівояжера цільова функція між сусідніми вершинами рівна, тому немає необхідності рахувати по два рази перехід від точки А до точки Б [6].

Клас P-повних задач включає задачі, які можливо точно вирішити за поліноміальний час, хоча на практиці зустрічаються задачі, які не мають прямого відношення до P-повних, але їх можна вирішити за прийнятний час, і навпаки, задачі які відносяться до P-повних, не завжди можна вирішити за поліноміальний час.

Алгоритмічний час задачі комівояжера тісно пов'язаний з задачею повного перебору або задачі рівності класів P та NP. Опис цієї проблеми можна подати наступним чином: якщо відповідь на задачу можна знайти за прийнятний час (поліноміальний), то чи можливо, використовуючи поліноміальну пам'ять також знайти відповідь на це питання, тобто чи є можливість перевірити сертифікат (певне значення, яке використовується для перевірки) за такий же час, що і знайти його [7].

Алгоритм пошуку в ширину. Для початку роботи алгоритму необхідно створити чергу, до якої будуть послідовно додаватися вже пройдені вершини, також для того щоб знати шлях повернення, якщо були пройдені всі вершини графа з однієї сторони, потрібно створити масив попередників. В ньому будуть позначатись шляхи, до вершини в яку ми потрапили. Основна ідея алгоритму полягає в обході вершин по кроку, тобто для кожної вершини шукається нащадок, якщо він є, відбувається перехід до нього, якщо немає і задача пошуку не була виконана, алгоритм повертається на попередню вершину.

Алгоритм пошуку в глибину. Пошук в глибину є схожим на пошук в ширину, основною відмінністю в цих двох алгоритмах є те, що пошук в глибину спочатку проходить від початкової вершини графа до максимального нащадка. Потім, коли алгоритм дійшов до кінцевої вершини, він аналізує, чи є ще невідомі вершини, якщо такі вершини присутні, алгоритм повертається назад, доки

не будуть відомі нові шляхи вниз.

Алгоритм Дейкстри. Дозволяє знаходити найбільш оптимальні шляхи в графі, в якому спочатку вибирається початкова вершина, а шлях до всіх інших є невідомим. Головним недоліком даного алгоритму є те, що він працює тільки для графів в яких всі ребра мають тільки позитивне значення. В житті бувають випадки, коли певні шляхи для компанії не є такими, що дають прибуток (звичайно від таких шляхів необхідно відмовитись, але якщо це шляхи підвищеної важливості, немає вибору і доводиться користуватися ними), то в такому випадку алгоритм Дейкстри не може бути використаним. Алгоритм Дейкстри чудово підходить для вирішення задач прокладання шляху, так як його використання має на увазі граф, який є зваженим (тобто для кожного ребра встановлюється вага, наприклад, платні автомагістралі) та орієнтованим (кожне ребро має направлення). В процесі роботи алгоритму необхідно використовувати три множини: V1 – множина вершин, до яких відстань вже була підрахована, V2 – множина вершин, до яких відстань ще рахується, V3 – множина вершин, відстань до яких ще не рахувалась [8].

Алгоритм Белмана-Форда. Алгоритм Белмана-Форда є подібним до алгоритму Дейкстри, основною перевагою даного алгоритму є можливість роботи з ребрами, які мають від'ємні ваги. В своїй роботі алгоритм Белмана-Форда використовує метод динамічного програмування (основна задача, розбивається на підзадачі, де для кожної під-задачі шукається вирішення, потім результат підзадач переходить на всю основну задачу.

Динамічний алгоритм, використовується для пошуку найбільш оптимальних шляхів серед всіх вершин в зваженому орієнтованому графі. Коректно працює тільки в випадку, якщо в графі немає циклів з від'ємною вагою, якщо в графі такий цикл все ж присутній, то дозволяє знайти хоча б один від'ємний цикл. Час роботи алгоритму O (n^3), використання пам'яті O(n^2). Алгоритм Флойда-Уоршела є більш загальним в порівнянні з алгоритмом Дейкстри, так як він дозволяє працювати з графами з від'ємними циклами та може знаходити ці цикли під час роботи [9].

*Алгоритм A**. Алгоритм A* (зірка) використовує модель пошуку по першому найкращому співпадінню. Це означає, що алгоритм пошуку, який досліджує граф, розширює найбільш перспективні вузли, обрані за допомогою правила.

На кожному кроці алгоритму необхідно визначити, який зі шляхів розширювати, за допомогою відомої вартості теперішнього шляху та приблизної вартості розширення шляху для точки, яка є цілю. Головною вимогою в алгоритмі A* є допустимість оцінки евристичної функції, вона завжди має обирати оптимальні рішення до обраної вершини, не повинна переоцінювати відстань до цільової точки [10].

II. МЕТА ДОСЛІДЖЕННЯ

Розробка програмного комплексу для обробки та аналізу адрес клієнтів на карті, розрахунку відстаней між ними та виведення оптимального маршруту.

III. МАТЕРІАЛИ ДОСЛІДЖЕНЬ

В якості середовища розробки вибрано Eclipse, оскільки воно має ряд переваг, таких як велику кількість підтримуваних бібліотек, модульність основних компонент та безкоштовність розповсюдження.

Для роботи із картографічним сервісом обрано бібліотеку GmapsFX.

GmapsFX являє собою бібліотеку з відкритим кодом, яка складається з різних методів зв'язку з картами в середовищі Eclipse. В нашій роботі було вибрано Java - ця мова володіє певними перевагами такими як, наприклад, велика розповсюдженість, Java використовується майже на 3 млрд. пристроїв по всьому світу, новітні методи роботи з даними (збирач сміття, автоматичне виділення пам'яті). Також використання Java, дозволяє в майбутньому розширити системний додаток на інші платформи, це досягається кросплатформністю обраної мови програмування.

При побудові найкоротших маршрутів були обрані наступні алгоритми: пошуку в ширину, пошуку в глибину, алгоритм Дейкстри, алгоритм Белмана-Форда, алгоритм Флойда-Уоршела, алгоритм A*, це дозволило проаналізувати їх роботу та обрати найкращі для прикладних задач.

Для зберігання даних була обрана система керування базами даних OracleSQL. Вона має ряд суттєвих переваг над усіма іншими: захищеність даних (дані постійно

шифруються), так як в базі зберігається інформація про пацієнтів, недоступність до персональної інформації є найголовнішою, дана система керування оптимізується корпорацією Sun Entertainment, яка також займається розробкою Java.

IV. РОЗРОБКА ПРОГРАМИ

Програма аналізу зображення працює в послідовності, як показано на рисунку 1.

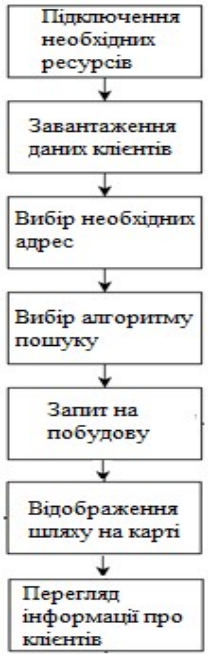

Рис. 1. Спрощена блок-схема роботи програми.

На початку виконуваного файлу підключаються всі необхідні бібліотеки, оголошуються всі глобальні змінні та оголошуються простори імен. Для завантаження даних клієнтів необхідно вибрати їх на комп'ютері (рис.2)

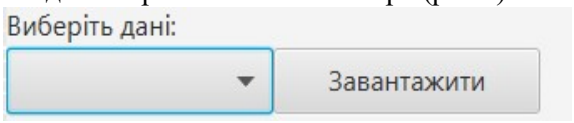

Рис. 2. Вибір даних клієнтів на комп'ютері.

Після завантаження даних клієнтів, необхідно обрати до яких саме клієнтів сьогодні буде прокладено найкоротший маршрут з використанням картографічного сервісу.

Після вибору адрес клієнтів, для більшої зручності та візуальної оптимізації виводиться інформація про широту та довготу вибраних місць (рис.3).

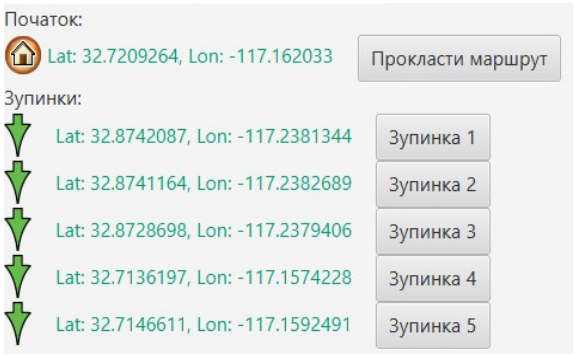

Рис. 3. Обрані адреси клієнтів на карті.

Меню вибору алгоритмів передбачає використання наступних алгоритмів побудови найкоротших шляхів: пошук в ширину, пошук в глибину, алгоритм Дейкстри, алгоритм Белмана-Форда, алгоритм Флойда-Уоршела та алгоритм A* (рис.4).

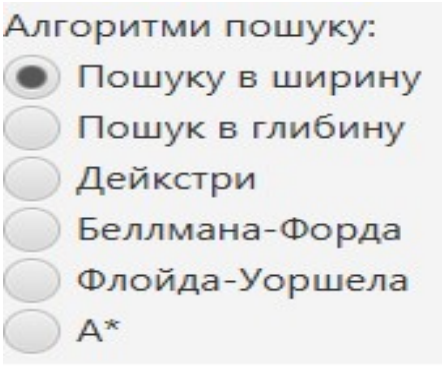

Рис. 4. Обрані алгоритми

Після натискання клавіші «Прокласти маршрут» відбувається оцінка відстаней від початкової адреси. Головною вимогою створюваного програмного додатку є повернення в початок, так як цього вимагає обрана модель задачі комівояжера. Для зручності, було обрано синій колір (такий же як і в GoogleMaps) при відображенні найкоротшого шляху на карті.

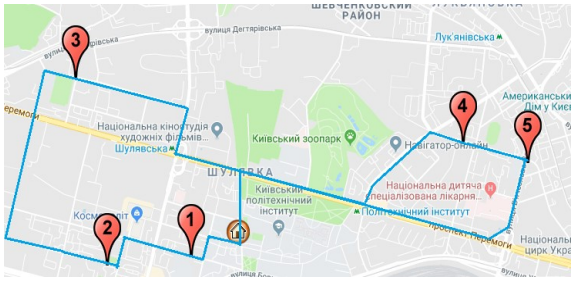

Рис. 5. Створений найкоротший маршрут.

Після побудови оптимального шляху на карті, користувач програмного додатку може перевірити основну інформацію про створений маршрут: відстань маршруту в кілометрах та час об’їзду в хвилинах (рис.5).

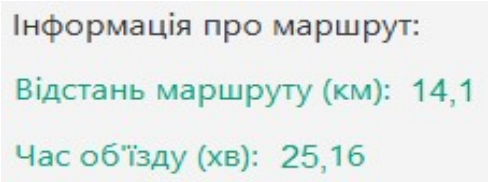

Рис. 5. Інформація про прокладений маршрут.

Програмний додаток має функціонал сімейного лікаря, де він може дивитись інформацію про пацієнтів (рис.6).

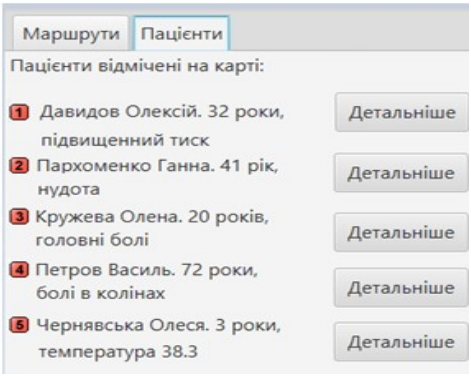

Рис. 6. Інформація про пацієнтів сімейного лікаря.

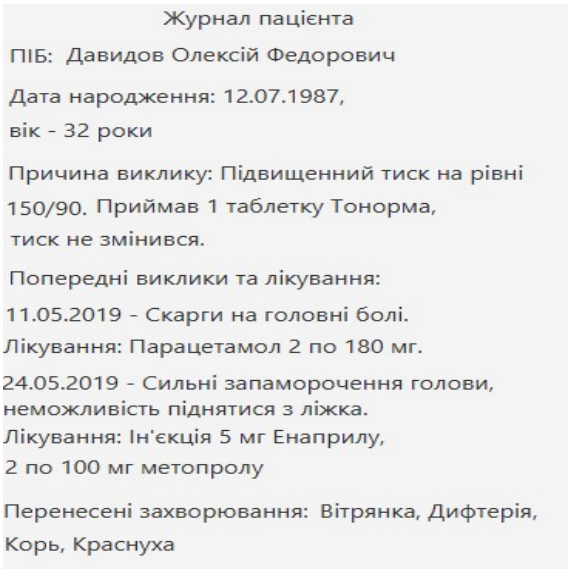

Рис.7. Історія викликів хворого.

V. ВИСНОВКИ

Розроблено програму, яка будує найкоротші маршрути на картографічному сервісі за допомоги алгоритмів оптимальних шляхів.

Для зручної роботи із программою лікарь може переглядати інформацію про пацієнтів, які його викликали та історію викликів певного хворого.

В ході тестування програми виявилось, що найкраще себе показали алгоритм Дейкстри та алгоритм A*, інші алгоритми працювали занадто повільно при збільшені точок на карті.

ПЕРЕЛІК ПОСИЛАНЬ

[1] Jonker, Roy; Volgenant, Ton. "Transforming asymmetric into symmetric traveling salesman problems". Operations Research Letters. 2 (161–163): 1983.

[2] Костевич Л. С. Математическое программирование: Информ. Технологи оптимальныхрешений: Учеб. пособие / Л. С. Костевич. — Мн.: Новоезнание, 2003. ил., стр. 150, ISBN 985-6516-83-8

[3] Haider A Abdulkarim, Ibrahim F Alshammari. "Comparison of Algorithms for Solving Traveling Salesman Problem". International Journa lof Engineering and Advanced Technology (IJEAT) ISSN: 2249 – 8958, Volume-4 Issue-6, August 2015

[4] Jayakumar S. Rigidflatfoot. / S. Jayakumar, H. Cowell. // Clin Orthop Relat Res.. – 2017. – №22. – С. 77–84.

[5] Angel E., Zissimopoulos V. On the classification of NP-complete problems in termsof their correlation coefficient // Discrete Appl. Math. 2. V. 9. P. 261–277.

[6] Yannakakis M. Computationa complexity. In: Aarts E., Lenstra J. (Eds.) Local search in combinatorial optimization. NY:Wiley, 1997.

[7] Kleinberg, Jon; Tardos, Éva (2006). Algorithm Design (2nd ed.). Addison-Wesley. p. 464. ISBN 0-321-37291-3.

[8] Korte B, Vugen J. Combinatorial optimization. Berlin: Springer, 2007.

[9] Yannakakis M. Computational complexity. In: Aarts E., Lenstra J. (Eds.) Local search in combinatorial optimization. NY:Wiley, 1997.

[10] Ежов А., Шумский С. Нейрокомпьютинг и его применения в экономике и бизнесе . — М.: МИФИ, 1998. — С. 216

УДК 004.021

СИСТЕМА ПОСТРОЕНИЯ ОПТИМАЛЬНЫХ МАРШРУТОВ НА ОСНОВЕ АЛГОРИТМОВ КРАТЧАЙШИХ ПУТЕЙ

Яковенко А.В., доцент
yakovenkoalena@ukr.net
Фолькин М.В., студент
markthefolkin@gmail.com
Факультет биомедицинской инженерии
Национальный технический университет
«Киевский политехнический институт шимени Игоря Сикорского»
м. Киев, Украина

Реферат *- На сегодняшний день с большим разнообразием транспортных средств, люди часто не могут определить лучшие пути передвижения, это все вызвано избытком информации, которую человеку трудно запомнить. Поэтому проблема создания маршрутов, с помощью которых человек смог бы в кратчайшие сроки добраться до запланированной цели предстает в наибольшем виде. Многие современные сервисы позволяют строить маршруты от точки А до точки Б, но они не могут удовлетворить пользователя, которому необходимо объехать несколько адресов. Для разработки была выбрана среда программирования Eclipse, библиотеку связи с картографическим сервисом GmapsFX, реализованную на языке Java. В результате разработана программа для поиска кратчайшего маршрута на карте с использованием алгоритмов оптимальных путей, разработан функционал семейного врача: просмотр пациентов, история вызовов.*

Ключевые слова - *алгоритмы точного и неточного поиска, алгоритм Дейкстры, алгоритм А *, задача коммивояжера, javafx, GoogleMaps API, дискретная оптимизация, непрерывная оптимизация.*

UDC 004.021

THE SYSTEM OF CONSTRUCTING OPTIMAL ROUTES BASED ON THE SHORTEST PATH ALGORITHMS

Yakovenko A.,docent
yakovenkoalena@ukr.net
Folkin M., student
markthefolkin@gmail.com
Faculty of Biomedical Engineering
National Technical University of Ukraine
“Igor Sikorsky Kyiv Polytechnic Institute”
Kyiv, Ukraine

Abstract - *To date, with a large variety of vehicles, people often can not solve the best ways of moving, this is all due to an excess of information that it is difficult for a person to memorize. Therefore, the problem of creating routes, with which a person could quickly reach the planned goal appears in the largest form. Many modern services allow you to build routes from point A to point B, but they can not satisfy a user who needs to travel multiple addresses. For development, the Eclipse programming environment, a library of communication with the mapping service GmapsFX, was implemented in Java. As a result, a program was developed for finding the shortest route on the map using the algorithms of optimal paths, and the functional of the family doctor was developed: patient viewing, history of calls.*

Keywords - *accurate and inaccurate search algorithms, Dijkstra algorithm, A * algorithm, traveling salesman problem, javafx, Google Maps API, discrete optimization, continuous optimization.*