ВІСНИК ЛЬВІВ. УН-ТУ *VISNYK LVIV UNIV.*
Серія екон. 2010. Вип. 43. С.25-31 *Ser. econ . 2010. Vol. 43. P.25-31*

УДК 519.857:004.051

ВІДШУКАННЯ НАЙКОРОТШИХ ШЛЯХІВ У ТРАНСПОРТНІЙ МЕРЕЖІ МЕТОДОМ ДИНАМІЧНОГО ПРОГРАМУВАННЯ

М. Квик[1], Г. Цегелик[1], Я. Романчук[2]

[1]Львівський національний університет ім. Івана Франка
79000 м. Львів, Університетська, 1
E-mail: kafmmsep@franko.lviv.ua
[2]Національний університет "Львівська політехніка"
79013 м. Львів, Ст. Бандери, 22/806
E-mail: romasu@polynet.lviv.ua

Проведено адаптацію методу динамічного програмування для розв'язування задачі визначення найкоротших відстаней від будь-якого пункту до всіх інших в заданій транспортній мережі. Алгоритм методу формалізовано і може бути програмно реалізований. Приводиться оцінка складності алгоритму і його переваги над алгоритмом Дейкстри.
Ключові слова: метод динамічного програмування, транспортна мережа, алгоритм визначення найкоротших відстаней.

Розглянемо задачу про визначення найкоротшої відстані від будь-якого пункту (вершини) до інших у заданій транспортній мережі. Вона знайшла своє широке відображення, наприклад, як у науковій [1, 6], навчальній [2, 3, 7, 9], так і спеціальній літературі [6 – 8]. Незважаючи на наявність досить ґрунтовної і різноманітної літератури з даного питання, проблема покращення існуючих алгоритмів розв'язування такої задачі, особливо із застосуванням методу динамічного програмування продовжує бути актуальною [4, 5, 10].

Нехай на деякій поверхні задана скінченна кількість точок $P_1, P_2, ..., P_n$, які з'єднані дугами (зв'язками) (P_i, P_j), що не перетинаються. Сукупність точок і дуг, які їх з'єднують, називатимемо мережею. Мережу будемо називати достатньо зв'язною, якщо для будь-яких двох точок існує шлях (сукупність вершин і дуг, що їх з'єднують), по якому можна пройти з однієї точки в іншу. При цьому, дві точки мережі називатимемо сусідніми, якщо існує дуга, що їх з'єднує.

Постановка задачі. Нехай задана достатньо зв'язна мережа, кожній дузі якої, що виходить із точки P_i і входить у точку P_j, поставлене у відповідність деяке дійсне невід'ємне число l_{ij} – її довжину, причому $l_{ij} = l_{ji}$. Треба визначити найкоротші шляхи в мережі від довільної точки до всіх інших і вказати відповідні їм відстані.

Для розв'язування задачі використаємо метод динамічного програмування [1, 10], згідно з яким будемо відшукувати найкоротші шляхи не від фіксованої точки до всіх інших, а найкоротші шляхи від усіх інших точок до фіксованої (заданої) через сусідні точки. Дугу, що міститься в найкоротшому шляху, позначатимемо стрілкою в напрямку до фіксованої точки. Вершини мережі позначатимемо кружечками (цифра всередині кружечка вказує номер точки), а в дужках біля них записуватимемо найкоротші відстані від цих точок до фіксованої точки. Відповідні найкоротші відстані називатимемо характеристиками точок.

Нехай

P_i – фіксована точка, до якої необхідно визначити найкоротші відстані та відповідні шляхи від усіх інших точок;

p_{ij} – кількість дуг, із яких складається найкоротший шлях від фіксованої точки P_i до поточної точки P_j;

k_s *(s = 1, 2, ..., m)* – кількість точок, найменша кількість дуг до яких від фіксованої точки P_i складає *s*, причому $m = \max\limits_{j \neq i} p_{ij}$;

$P_1^{(s)}$, $P_2^{(s)}$, …, $P_{k_s}^{(s)}$ *(s = 1, 2, ..., m)* – точки, найменша кількість дуг до яких від фіксованої точки P_i складає *s*.

Алгоритм розв'язування задачі. Алгоритм складається з початкового кроку та загального, що повторюється *m*-1 разів.

Початковий крок. Біля кружечка, що позначає точку P_i, записуємо нуль – характеристику цієї точки, оскільки відстань від точки P_i до неї самої дорівнює нулю. Визначаємо сусідні з P_i точки і біля кружечків, якими позначені ці точки, записуємо їх характеристики, тобто $0 + l_{ij}$, якщо P_j є сусідньою точкою, а на дугах ставимо стрілки, спрямовані в сторону точки P_i. Після цього відмічаємо точку P_i символом +, який означатиме, що операція над цією вершиною завершена.

Загальний крок. Припустимо, що ми вже виконали *r* кроків: знайшли характеристики всіх точок, найменша кількість дуг до яких від фіксованої точки P_i є не більшою за *r*; точки $P_1^{(s)}$, $P_2^{(s)}$, …, $P_{k_s}^{(s)}$ *(s = 1, 2, ..., r-1),* помічені символом +; характеристики точок $P_1^{(r)}$, $P_2^{(r)}$, …, $P_{k_r}^{(r)}$ відповідно дорівнюють $C_1^{(r)}$, $C_2^{(r)}$, …, $C_{k_r}^{(r)}$ і від цих точок проведені стрілки в напряму до точки P_i.

Тоді на *(r+1)*-му кроці беремо будь-яку точку $P_i^{(r)}$ і визначаємо характеристики всіх сусідніх до неї точок (крім точки, до якої спрямована стрілка від $P_i^{(r)}$) як суму $C_i^{(r)}$ і відстані від точки $P_i^{(r)}$ до кожної з них. Після цього точку $P_i^{(r)}$ відмічаємо символом +.

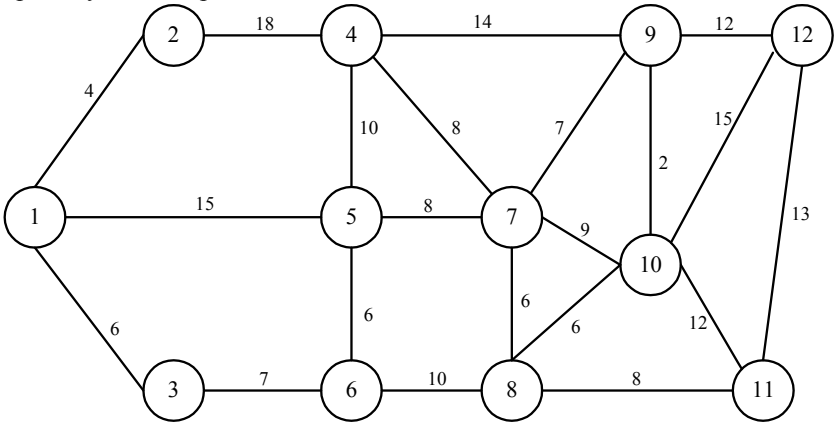
Зауважимо, що при визначенні характеристик сусідніх із $P_i^{(r)}$ точок може виявитися, що характеристика сусідньої точки вже була обчислена раніше. Нехай це буде точка P_q, стара характеристика якої C_q, а наново обчислена характеристика цієї точки C'_q. У такому випадку порівнюємо між собою величини C'_q із C_q. Якщо $C'_q \geq C_q$, то характеристику C_q залишаємо без зміни. Якщо ж $C'_q < C_q$, то старе C_q замінюємо на C'_q. Відповідно зміниться зв'язок, через який проходить найкоротший шлях до точки P_q, а стрілка на дузі, що виходить із точки P_q, заміниться на стрілку на дузі $(P_q, P_i^{(r)})$, що виходить із цієї точки.

Якщо при цьому змінилася характеристика якоїсь конкретної точки P_q, яка раніше вже була відмічена символом +, то перераховуємо характеристики сусідніх із нею точок (крім точки $P_i^{(r)}$) і змінюємо при потребі їхні характеристики і напрям стрілок.

Якщо ж при визначенні характеристик сусідніх із $P_i^{(r)}$ точок виявиться, що характеристика якоїсь сусідньої точки раніше не обчислювалася, то на відповідній дузі, що виходить із цієї точки, ставимо стрілку в напрямку до точки $P_i^{(r)}$.

Нарешті, відзначимо, що *(r+1)*-ий крок виконуватимемо доти, доки послідовно не будуть перебрані всі вершини мережі, тобто точки $P_i^{(r)}$ $(i = 1,2,...,k_r)$.

Приклад. Відшукаємо найкоротші шляхи від усіх точок до точки з номером 1 у такій мережі:



Розв'язування. *Початковий крок.* Точці 1 ставимо у відповідність характеристику нуль і визначаємо характеристики точок 2, 5, 3. Ці характеристики відповідно дорівнюють 4, 15, 6. На дугах (1, 2), (1, 5), (1, 3) ставимо стрілки, спрямовані в сторону точки 1, а саму точку 1 відмічаємо символом +.

Перший крок. Беремо точку 2 і визначаємо характеристику сусідньої до неї точки 4. Вона дорівнюватиме 4 + 18 = 22. Тому на дузі (2, 4) ставимо стрілку, спрямовану до точки 2, а точку 2 відмічаємо символом +.

Беремо точку 5 і визначаємо характеристики її сусідніх точок 4, 6, 7. Вони відповідно складатимуть 25, 21, 23. Нова характеристика точки 4 дорівнює 25. Оскільки 25 > 22, то характеристика точки 4 залишається рівною 22. На дугах (5, 6) і (5, 7) ставимо стрілки, спрямовані до точки 5, а точку 5 відмічаємо символом +.

Далі беремо точку 3 і визначаємо характеристику її сусідньої точки 6. Вона становить 13, але для точки 6 раніше вже була обчислена характеристика, яка дорівнює числу 21. Тому, оскільки 13 < 21, за характеристику точки 6 приймемо 13, а стрілку на дузі (5, 6) замінюємо на стрілку на дузі (3, 6), спрямовану до точки 3, після чого точку 3 відмічаємо символом +.

Другий крок. Беремо точку 4 і визначаємо характеристики її сусідніх точок 5, 7, 9. Вони відповідно рівні 32, 30, 36. Раніше обчислені характеристики точок 5 і 7 становлять, відповідно, 15 і 23. Оскільки 32 > 15 і 30 > 23, то характеристики точок 5 і 7 залишаємо без змін, тобто 15 і 23 відповідно. Після цього на дузі (4, 9) ставимо стрілку, спрямовану до точки 4, яку відмічаємо символом +.

Беремо точку 7 і визначаємо характеристики її сусідніх точок 4, 8, 9, 10. Вони відповідно дорівнюватимуть 31, 29, 30, 32. Але характеристики точок 4 і 9, які були обчислені раніше, є 22 і 36 відповідно. Оскільки 31 > 22, а 30 < 36, то характеристика точки 4 залишається рівною 22, а характеристику точки 9 замінюємо на 30. Відповідно до цього стрілку на дузі (4, 9) замінюємо стрілкою на дузі (7, 9), спрямованою до точки 7. Далі на дугах (7, 8) і (7, 10) ставимо стрілки, спрямовані до точки 7, а саму точку 7 відмічаємо символом +.

Далі беремо точку 6 і визначаємо характеристики її сусідніх точок 5 і 8. Вони, відповідно, становлять 19 і 23. Раніше обчислені характеристики цих точок були, відповідно, 15 і 29. Оскільки 19 > 15, а 23 < 29, то характеристика 15 точки 5 залишається без зміни, а характеристику точки 8 замінюємо на 23. Тому стрілку на дузі (7, 8) замінюємо стрілкою на дузі (6, 8), спрямованою до точки 6. Точку 6 відмічаємо символом +.

Третій крок. Беремо точку 9 і визначаємо характеристики її сусідніх точок 4, 10 і 12. Вони, відповідно, дорівнюють 44, 32 і 42. Оскільки раніше обчислені характеристики точок 4 і 10 – це значення 22 і 32 відповідно, то маємо: 44 > 22 і 32 = 32. Тому характеристики точок 4 і 10 залишаємо без змін. На дузі (9, 12) ставимо стрілку в напрямку до точки 9, після чого точку 9 відмічаємо символом +.

Розглянемо далі точку 10 і визначимо характеристики її сусідніх точок 8, 9, 11, 12. Вони, відповідно, будуть 38, 34, 44, 47. Для точок 8, 9, 12 раніше вже були обчислені відповідні характеристики, а саме: 23, 30, 42. Тому, оскільки 38 > 23, 34 > 30 і 47 > 42, характеристики точок 8, 9, 12 залишаємо без змін, а на дузі (10, 11) ставимо стрілку в напрямку до точки 10. Точку 10 відмічаємо знаком +.

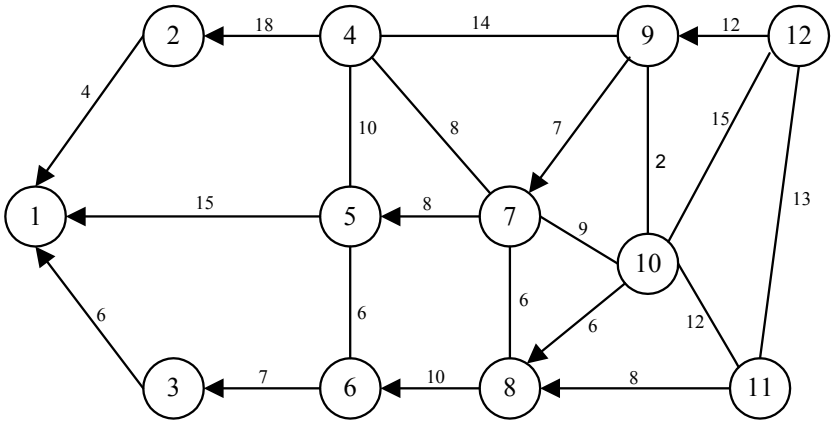
Беремо точку 8 і визначаємо характеристики її сусідніх точок 7, 10 і 11. Вони, відповідно, дорівнюють 29, 29 і 31. Раніше для цих точок уже були обчислені характеристики: 23, 32 і 44 відповідно. Оскільки 29 > 23, а 29 < 32 і 31 < 44, то характеристика точки 7 залишається без зміни, характеристики точок 10 і 11 заміняться, відповідно, на 29 і 31, а стрілки на дугах (7, 10) і (10, 11) заміняться стрілками на дугах (8, 10) і (8, 11), спрямованими до точки 8. Точку 8 відмічаємо символом +. Але, оскільки при цьому змінилася характеристика точки 10, яка вже

була відмічена символом +, то перераховуємо характеристики її сусідніх точок 9, 11, 12. Одержимо, відповідно, значення 31, 41, 44. Однак, для цих точок раніше вже були обчислені відповідні характеристики 30, 31 і 42. Оскільки 31 > 30, 41 > 31 і 44 > 42, то характеристики точок 9, 11 і 12 залишаємо без змін.

Четвертий крок. Розглянемо точку 11 і визначимо характеристики її сусідніх точок 10 і 12. Вони дорівнюватимуть, відповідно, 43 і 44. Але раніше обчислені характеристики цих точок, відповідно, дорівнюють 29 і 42. Оскільки 43 > 29 і 44 > 42, то старі характеристики цих точок залишаємо без змін. Точку 11 відмічаємо символом +.

Нарешті, беремо точку 12, для її сусідніх точок 10 і 11 характеристики, відповідно, дорівнюють 57 і 55. Раніше обчислені для них характеристики, відповідно, становлять 29 і 31. Оскільки 57 > 29 і 55 > 31, то обчислені раніше характеристики точок 10 і 11 залишаємо без змін. Точку 12 відмічаємо символом +.

Отже, ми одержали розв'язок задачі, який показаний нижче на графі стрілками.



Оптимальні шляхи можна зобразити, вказуючи спочатку значення характеристики вершини (відстань до точки 1) і відповідний шлях:

4: 2→1;
6: 3→1;
22: 4→2→1;
15: 5→1;
13: 6→3→1;
23: 7→5→1;
23: 8→6→3→1;
30: 9→7→5→1;
29: 10→8→6→3→1;
31: 11→8→6→3→1;
42: 12→9→7→5→1.

Запропонований тут алгоритм методу динамічного програмування для задачі визначення в мережі найкоротших відстаней від будь-якої вершини до заданої був програмно реалізований. Результати, одержані для розглянутого прикладу на основі комп'ютерної програми, збігаються із наведеними тут.

Перевагою цього алгоритму є те, що він у порівнянні з іншими алгоритмами має більшу швидкодію: необхідний час T роботи складає $T=O(n^2+3n)$ проти $T=O(n^3)$ за алгоритмом Дейкстри [8] чи, наприклад, проти $T=O(n^3b^2)$ і необхідної пам'яті $П=o(n^2b)$, де n – кількість вершин мережі, b – максимальний обсяг попиту за [4]. Якщо мережа є ланцюгом, то останній алгоритм має оцінки $T=O(n^3)$, $П=o(n)$.

Висновки. Метод динамічного програмування сьогодні широко застосовують до розв'язування багатокрокових задач теорії керування, мікро- та макроекономіки, соціальної сфери. Запропонований нами алгоритм є економічнішим у порівнянні з відомими в літературі, оскільки час його роботи майже на порядок менший за час роботи відомих алгоритмів.

---

1. Беллман Р., Дрейфус С. Прикладные задачи динамического программирования [Текст] : Пер. с англ. под ред. А.А. Первозванского. – М.: Наука, 1965. – 457 с.
2. Бурдук Е.Л. Исследование операций [Текст] : учеб. пособ. / Е.Л. Бурдук, И.Н. Кравченя : М-во обр. Респ. Беларусь, Белор. гос. ун-т трансп. – Гомель: БелГУТ, 2008. – 74 с.
3. Вентцель Е.С. Исследование операций. Задачи, принципы, методология : [Текст] : учеб. пособ. для вузов. – 3-е изд. – М.: Дрофа, 2004. – 208 с.
4. Вознюк И.П. Задача размещения на сети с ограниченными пропускными способностями коммуникаций // Дискретный анализ и исследование операций. – 1999. – Серия 2, том 6. – № 1. – С. 3 – 11.
5. Камышан А. Динамическое программирование // Дискретная математика: Алгоритмы [Електронний ресурс]: Режим доступу : http://rain.ifmo.ru/ cat/view.php/theory/algorithm-analysis/dynamic-programming-2004
6. Кристофидес Н. Теория графов. Алгоритмический подход [Текст] : – М.: Мир, 1978. – 432 с.
7. Лежнев А.В. Динамическое программирование в экономических задачах [Текст] : учебн. пособ. / А.В. Лежнев. – М.: Бином. Лаборатория знаний, 2006. – 176 с.
8. Липский В. Комбинаторика для программистов [Текст] : Пер. с польск. – М.: Мир. – 1988. – 213 с.
9. Черноусько Ф.Л. Динамическое программирование // Математика. – 1998 [Електронний ресурс]: Режим доступу : http://www.pereplet.ru/obrazovanie /stsoros/501.html.
10. Щербина О.А. Методологические аспекты динамического программи-рования // Динамические системы, вып. 22. – 2007. С. 21 – 36. [Електронний ресурс] : Режим доступу : http://dynsys.crimea.edu/issue/22/ dynsys_22_Shcherbina.pdf.

SEARCHING FOR THE SHORTEST WAYS IN A TRANSPORT NETWORK BY METHOD OF DYNAMIC PROGRAMMING

M. Kvyk[1], H. Tseheluk[1], Ya. Romanchuk[2]

[1]I. Franko Lviv National University, 1 Universytets'ka St., Lviv-00, 79000, Ukraine
E-mail: kafmmsep@franko.lviv.ua
[2]Lviv Polytechnic National University, 22/806 Bandera St., Lviv-13, 79013, Ukraine
E-mail: romasu@polynet.lviv.ua

Adaptation of method of the dynamic programming is conducted for acomplishing of the task of estimating the shortest distances from any point to all the rest in the set transport network. The algorithm of the method is formalized and can be programmaticly realized. An estimation of complication of the algorithm and its advantages above the algorithm of Deykstri are brought.

Keywords: method of the dynamic programming, transport network, algorithm of estimating the shortest distances.

ОТЫСКИВАНИЕ КРАТЧАЙШИХ ПУТЕЙ В ТРАНСПОРТНОЙ СЕТИ МЕТОДОМ ДИНАМИЧЕСКОГО ПРОГРАММИРОВАНИЯ

М. Квык[1], Г. Цегелык[1], Я. Романчук[2]

[1]Львовский национальный университет им. Ивана Франка
79000 г. Львов, Университетская, 1
E-mail: kafmmsep@franko.lviv.ua
[2]Национальный университет "Львовская политехника"
79013 г. Львов, Ст. Бандеры, 22/806
E-mail: romasu@polynet.lviv.ua

Проведена адаптация метода динамического программирования для решения задачи определения кратчайших расстояний от любого пункта ко всем другим в заданной транспортной сети. Алгоритм метода формализирован и может быть программно реализован. Приводится оценка сложности алгоритма и его преимущества над алгоритмом Дейкстры.

Ключевые слова: метод динамического программирования, транспортная сеть, алгоритм определения кратчайших расстояний.