УДК 004.02+519.17

ПОБУДОВА МАКСИМАЛЬНОГО ПРОСТОГО ШЛЯХУ ГРАФА

В. В. Черняхівський

Львівський національний університет ім. І. Франка

E-mail: v_chrn@franko.lviv.ua

Розглянуто задачу пошуку простого шляху графа для побудови максимального ланцюга. Базова постановка задачі не накладає жодних умов на процедуру пошуку максимального шляху. Розширена постановка задачі дає можливість виконувати керований пошук на основі визначеного поняття серединної умови ланцюга графа. Серединні умови поділені за типами на ізольовані і зв'язані. Введено поняття конструктивної повноти серединних умов і сформульовані твердження про їхні властивості.

Ключові слова: *граф, простий ланцюг, максимальний шлях, серединна умова, властивість.*

CONSTRUCTION OF THE MAXIMAL SIMPLE PATH OF A GRAPH

V. V. Chernyakhivskij

Ivan Franko National University of Lviv

The problem of a simple graph path searching is considered for the case of maximal chain construction. The investigation deals with expansion of domain of search tasks on graphs. A recursive algorithm of searching the maximal simple path is described. It is based on the backtracking method. In the first case we solve the problem without any additional restrictions. In the second one we find a maximal simple path across the given vertexes and/or bypasses of another given vertexes. The concept of mid condition of a graph chain is introduced. The isolated mid conditions of types 1, 2, 3 and bounded ones of type 4, 5 are considered. The constructive completeness concept of mid conditions is introduced and a way of its interpretation is described. The results of the investigation of all types of mid conditions constructive completeness are given. The use of mid conditions allows us to perform the controlled search with account of specific criteria derived from the formulation of an external problem. Separately the important in practice case of a simple chain of a circular path with coinciding bounding conditions at the start and the end points of the path is considered. The example of calculation of a circular transport traffic based on the road map is given.

Keywords: *graph, simple chain, longest path, middle condition, property.*

Базова пошукова задача. Моделі предметних областей багатьох практичних задач можна подати орієнтованим або неорієнтованим графом. Наприклад, відшукання маршруту руху між двома заданими містами, проектування схеми водопостачання через задані пункти, планування перевезень товарів на основі карти розташування клієнтів і схеми доріг тощо. Розглядаємо клас задач, модель яких можна подати у вигляді неповного графа. Потрібно відшукати максимальний простий ланцюг між двома заданими вершинами V_a і V_b. Така постановка задачі є симетричною до відомої задачі побудови мінімального ланцюга графа. Переважна більшість наукових публікацій, наприклад [1, 2], пов'язана із задачами мінімального або оптимального пошуку відповідно до заданих критеріїв. Дослідження пошукових задач на графах орієнтовані в основному на досягнення різноманітних критеріїв оптимальності [3–5].

Дослідження цієї статті пов'язані з розширенням кола пошукових задач на графах – обчислення максимального шляху графа між заданими вершинами. Цю задачу пов'язуватимемо зі сучасними дослідженнями методів і задач граничного пошуку [6] як ідеї загального підходу до постановки проблеми.

Отже, маємо:

1. Модель деякої предметної області у вигляді неорієнтованого неповного графа $G=(V, E)$ з кількістю вершин $n=|V|$ і з ребрами однакової ваги або незваженими. Вершини і ребра зафіксовані наперед з точністю до ізоморфізму.

2. Ніяких умов на окремі ребра і вершини не накладаємо.

3. Умов взаємного зв'язку ребер і вершин немає.

4. Граф може бути зв'язним або незв'язним, проте за незв'язності задача переходить на окрему компоненту зв'язності.

5. Планарності графа не вимагаємо. З практичних міркувань розглядаємо як приклади планарні графи.

6. З обчислювальних міркувань приймаємо зображення графа матрицею суміжності $A[n,n]$, де $n=|V|$.

Потрібно отримати:

1. Метод і алгоритм побудови простого ланцюга максимальної довжини, який з'єднує дві будь-які наперед задані вершини $V_a, V_b : (V_a, V_1, V_2, ..., V_k, V_b)$, $k \to \max$, де $V_i \neq V_j$ при $i \neq j$, крім, можливо, V_a і V_b. В останньому випадку маємо циклічний простий ланцюг. При $V_a = V_b$, $k = n-1$ отримаємо гамільтонів цикл.

2. Програмне обчислення простого максимального ланцюга.

Схема методу і алгоритм побудови. Рекурсивний алгоритм P_0^{r} відшукання максимального простого ланцюга побудовано на підставі методу пошуку з поверненнями і опубліковано раніше [7]. Алгоритм виконує перегляд усіх можливих простих ланцюгів, які починаються у вершині V_a, і використовує дві головні структури даних: матрицю суміжності $A[n,n]$ і список L_+ вже відвіданих на цей момент вершин. Усі вершини графа нумерують натуральними числами 1, 2, ..., n у довільному порядку. За один крок рекурсії алгоритм $P_0^{\mathrm{r}} = P_0^{\mathrm{r}}(i)$ будує одну наступну ланку простого ланцюга, починаючи з вершини i: $P_0^{\mathrm{r}}(i) \to j$, якщо це можливо, і виконує рекурсивно $P_0^{\mathrm{r}}(j)$. У цьому разі додаємо вершину i до списку L_+. Якщо ж нову ланку неможливо побудувати, то крок алгоритму не виконує ніяких операцій. Отже, $P_0^{\mathrm{r}} = P_0^{\mathrm{r}}(P_0^{\mathrm{r}}(P_0^{\mathrm{r}}(\ldots (P_0^{\mathrm{r}}(a)) \ldots)))$. Побудова чергової ланки $P_0^{\mathrm{r}}(i) \to j$ супроводжується викресленням вершини i та всіх інцидентних їй ребер з графа. Вершина i викреслена, якщо $i \in L_+$, а наступну вершину j для простого ланцюга треба шукати в множині $j \in \{V\} \setminus L_+$.

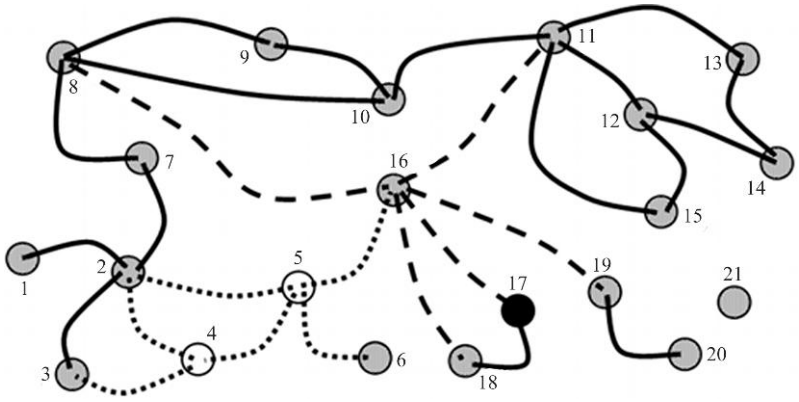

Рис. 1. Поточний стан графа на шляху (4–5–16).

Зауважимо, що множина $\{V\} \setminus L_+$ враховує вершину V_b, або V_a і V_b при $V_a = V_b$, отже, на кожному кроці алгоритму існує можливість отримати гамільтонів шлях або гамільтонів цикл – якщо це можливо за структурою графа.

На рис. 1 показано поточний стан графа після побудови ланцюга (4–5–16). Пунктирними лініями показано викреслені ребра, а викреслені вершини незафарбовані. Штриховою лінією показано можливі продовження ланцюга з вершини 16.

За описаним алгоритмом знаходимо, наприклад, для $V_a = 4$, $V_b = 17$ такий ланцюг: 4–3–2–7–8–9–10–11–16–18–17.

Зауваження 1. Якщо виявиться декілька найдовших ланцюгів однакової довжини, то за наведеним алгоритмом результатом буде перший зі знайдених ланцюгів. Алгоритм можна модифікувати для знаходження і запам'ятовування всіх найдовших ланцюгів рівної довжини.

Зауваження 2. Алгоритм допускає відсутність ланцюга $V_a \rightarrow ... \rightarrow V_b$, якщо граф незв'язний і вершини V_a і V_b належать різним зв'язним компонентам. У цьому разі поведінка алгоритму залишається коректною.

Керований пошук. ***Загальні серединні умови.*** Задача побудови максимального простого ланцюга має два випадки:

1) для будь-якої заданої початкової V_a і кінцевої V_b вершин знаходити максимально можливий простий ланцюг без додаткових умов – розглянуто вище;

2) шукати максимальний простий ланцюг з умовою, що деякі зазначені вершини мають бути в знайденому ланцюзі, або/і деякі вершини не повинні бути в знайденому ланцюзі.

Розв'язок другого випадку побудовано на визначеному понятті *серединної умови ланцюга графа* [8, 9].

Позначимо множину $M = \{V\} \setminus \{V_a, V_b\}$ вершин, які не є граничними для ланцюга. Для кожної вершини $m_i \in M$ визначимо деякий предикат $r_i = r_i(M)$. Предикат може набувати одного з трьох значень: {–1; 0; +1}. При r_i = +1 вершина i повинна бути включена до ланцюга L, при r_i = –1 не має бути в ланцюзі L, а при r_i = 0 присутність чи відсутність вершини в ланцюзі не визначено. У загальному випадку предикат $r_i(M)$ є функцією від усіх вершин множини M, у частковому випадку $r_i = r_i(V_i)$ – якщо потрібно розглядати всі вершини як взаємно незалежні.

Означення. Узагальненою серединною умовою задачі побудови простого ланцюга визначимо вектор $Z = \{r_i(M)\}, i = 1, 2, ..., |M|$.

Серединні умови поділяємо залежно від функціонального вигляду предиката r_i на ізольовані і зв'язані.

Щоб мати змогу враховувати серединні умови під час побудови ланцюгів, модифікували базовий рекурсивний алгоритм P_0^{r} відшукання максимального простого ланцюга [7]. Тепер перевірки лише довжини ланцюга буде недостатньо, потрібно додатково перевіряти виконання серединної умови. Процедура *TestPathWithMiddleCond* (внутрішня реалізація) доповнена перевірками визначених типів серединних умов відповідно до заданого вектора $Z = \{r_i(M)\}$ і прийняттям рішення про запам'ятовування ланцюга. При цьому відсутність серединної умови є просто одним з варіантів виконання модифікованого алгоритму еквівалентно до базового алгоритму.

Ізольовані серединні умови. Розглядаємо предикат $r_i = r_i(V_i)$, тобто предикат є функцією лише власної вершини, а всі вершини графа взаємно незалежні в максимальному простому ланцюзі. Значення предиката визначають окремо для кожної вершини, враховуючи потребу отримати точнішу структуру шуканого

максимального ланцюга. Отже, розглядаємо серединні умови як вектор $Z = \{r_i(V_i)\}$. Такий вектор назвемо ізольованими серединними умовами [8].

Ізольована серединна умова типу 1. Умовою типу 1 (або [+1; 0]) називають вектор $Z = \{r_i(V_i)\}, i = 1,\ 2,\ ...,\ |M|$, де $r_i(V_i) = 1$ для вершин *i*, які повинні входити до ланцюга *L*, і $r_i(V_i) = 0$ для всіх інших вершин.

Умову типу 1 застосовуємо тоді, коли максимальний ланцюг має обов'язково проходити через деякі фіксовані вершини, хоча порядок проходження через такі вершини не визначений. У ланцюгу мають бути граничні вершини і всі вершини з умовою $r_i(V_i) = 1$, наявність решти вершин визначають критерієм максимізації.

Ізольована серединна умова типу 2. Умовою типу 2 (або [–1; 0]) називають вектор $Z = \{r_i(V_i)\}, i = 1,\ 2,\ ...,\ |M|$, де $r_i(V_i) = -1$ для вершин *i*, які не повинні бути у ланцюзі *L*, і $r_i(V_i) = 0$ для всіх інших вершин.

Умову типу 2 застосовуємо тоді, коли з максимального ланцюга треба вилучити деякі фіксовані вершини. В ланцюзі мають бути граничні вершини, а також вершини з умовою $r_i(V_i) = 0$ за критерієм максимізації.

Ізольована серединна умова типу 3. Умовою типу 3 (або [±1;0]) називають вектор $Z = \{r_i(V_i)\},\ i = 1,\ 2,\ ...,\ |M|$, де $r_i(V_i) = 1$ для вершин *i*, які повинні входити до ланцюга *L*, $r_i(V_i) = -1$ для вершин *i*, які не повинні бути у ланцюзі *L*, і $r_i(V_i) = 0$ для всіх інших вершин.

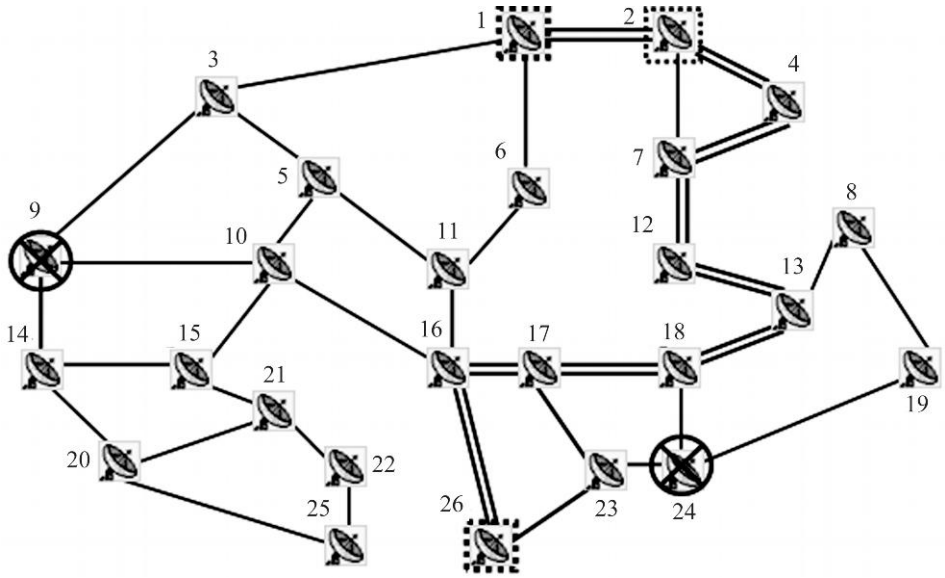

Рис. 2. Простий максимальний ланцюг 1–26 за умови $r_2 = +1$, $r_9 = -1$, $r_{24} = -1$.

Умову типу 3 застосовуємо тоді, коли максимальний ланцюг одночасно має обов'язково проходити через деякі фіксовані вершини і в цьому разі з ланцюга треба вилучити інші фіксовані вершини. В ланцюзі мають бути граничні вершини, всі вершини з умовою $r_i(V_i) = 1$, а також з умовою $r_i(V_i) = 0$ за критерієм максимізації. Підмножина вершин з умовою $r_i(V_i) = 0$ для умови типу 3 має менше елементів, ніж для умови типу 1.

На рис. 2 подано приклад моделі графа комутованої мережі передавання сигналів. Ребра графа (одинарна лінія) – це наявність прямого зв'язку. Задача по-

лягає в побудові максимального шляху проходження сигналу між двома заданими пунктами 1 і 26 за умови однократного включення пунктів – простий шлях. Серединна умова типу 3 визначена так: $r_2 = +1$, $r_9 = -1$, $r_{24} = -1$. Подвійною лінією показано розв'язок задачі: 1–2–4–7–12–13–18–17–16–26.

Зв'язані серединні умови. Розглядаємо предикат $r_i = r_i(M)$, де $M = \{V\} \setminus \{V_a, V_b\}$ – множина вершин, які не є граничними для шуканого ланцюга. Загальний випадок предиката визначає його як функцію всіх вершин графа, крім граничних, для шуканого простого ланцюга. Вектор $Z = \{r_i(M)\}$ назвемо зв'язаними серединними умовами [9].

Отже, розглядаємо задачу на графі за двома умовами: 1) відшукати максимальний ланцюг між двома заданими вершинами; 2) врахувати зв'язані серединні умови вигляду $r_i = r_i(M)$, $M = \{V\} \setminus \{V_a, V_b\}$ для вершин всередині шуканого ланцюга.

Зв'язана серединна умова типу 4. Умовою типу 4 (або $[pos(+1);0]$) називають вектор $Z = \{r_i(M)\}$, $i = 1, 2, ..., |M|$, де $r_i(M) = 1$ для вершин i, які повинні бути включені до ланцюга L і для яких виконується умова $U(pos_i) = true$, і $r_i(M) = 0$ для всіх інших вершин. За умовою типу 4 вершини, які повинні бути включені до ланцюга L, взаємно залежні щодо порядку розташування в ланцюзі.

Нехай деяка вершина i повинна бути включена до ланцюга L. Позначимо pos_i позицію вершини i у знайденому ланцюзі як відстань від вершини V_a. Тоді умову $U(pos_i)$ визначимо так:

$$U(pos_i) = (pos_l < pos_i < pos_r);\ \text{для всіх}\ l = 1, 2, ..., i-1;\ r = i+1, i+2, ..., k.$$

l – вершини, розташовані на шляху ланцюга перед вершиною i; r – вершини, розташовані на шляху ланцюга після вершини i; k – кількість вершин, для яких $r_i(M) = 1$. У розгорнутому вигляді умову U можна записати так:

$$U = (pos_1 < pos_2 < pos_3 < ... < pos_k).$$

Умову типу 4 застосовуємо тоді, коли максимальний ланцюг має обов'язково проходити через деякі фіксовані вершини і порядок проходження через них визначений додатковою умовою $U(pos_i)$, принаймні для окремих вершин.

Зв'язана серединна умова типу 5. Умовою типу 5 (або $[pos(+1);-1;0]$) називають вектор $Z = \{r_i(M)\}, i = 1, 2, ..., |M|$, де $r_i(M) = 1$ для вершин i, які повинні бути включені до ланцюга L і для яких виконується умова $U(pos_i) = true$, $r_i(M) = -1$ для вершин, які не повинні бути у ланцюзі L, і $r_i(M) = 0$ для всіх інших вершин. Умову $U(pos_i)$ визначають так само, як і серединну умову типу 4. За умовою типу 5 вершини, які повинні бути включені до ланцюга L, взаємно залежні щодо порядку розташування в ланцюзі.

Умову типу 5 застосовуємо тоді, коли потрібно розв'язати задачу з такими додатковими обмеженнями: максимальний ланцюг має обов'язково проходити через деякі фіксовані вершини; порядок проходження через такі вершини визначений умовою $U(pos_i)$, принаймні для однієї пари вершин; з ланцюга потрібно вилучити інші фіксовані вершини.

Конструктивна повнота серединних умов. У теоретичному дослідженні викладеної задачі і методів розв'язування має важливе значення те, які розв'язки в принципі можна отримати за розробленим алгоритмом [7] і побудованими

визначеннями серединних умов [8, 9]. Відповідь ґрунтуємо на введеному понятті конструктивної повноти серединних умов, яке досліджено в працях [8, 9].

Означення 1. Тип серединної умови $T = T(V_a, V_b)$ для граничних вершин V_a, V_b називається *конструктивно повним для вершин*, якщо граф $G = (V, E)$ зв'язний і максимальний простий ланцюг від V_a до V_b може бути побудований за таким типом умови для будь-якої підмножини попарно суміжних вершин графа $G = (V, E)$, які можна сполучити в ланцюг за структурою графа, і *конструктивно неповним для вершин* у протилежному випадку. Тобто ланцюг можна отримати для будь-якої підмножини вершин $\{V_{i1}, V_{i2}, \ldots, V_{ik}\}$, де суміжними є всі пари

$$(V_a, V_{i1}),\ (V_{i1}, V_{i2}),\ (V_{i2}, V_{i3}),\ (V_{i3}, V_{i4}),\ \ldots,\ (V_{i(k-1)}, V_{ik}),\ (V_{ik}, V_b)$$

з точністю до порядку сполучення.

Інтерпретація повноти серединної умови для вершин означає обернене формулювання задачі побудови максимального ланцюга: чи можна записати сукупність серединних умов $Z_1 \cup Z_2 \cup \ldots \cup Z_N$ так, щоб розв'язками задачі були всі можливі підмножини вершин $\{V\} \setminus \{V_a, V_b\}$, які належать хоча б одному простому ланцюгу від V_a до V_b, який можна побудувати за структурою графа. Порядок обходу вершин ланцюга значення не має.

Означення 2. Тип серединної умови $T = T(V_a, V_b)$ для граничних вершин V_a, V_b називається *конструктивно повним для шляхів*, якщо граф $G = (V, E)$ зв'язний і максимальний простий ланцюг може бути побудований за таким типом умови для будь-якого допустимого ланцюга графа $G = (V, E)$ від вершини V_a до V_b, і *конструктивно неповним для шляхів* у протилежному випадку.

Інтерпретація повноти серединної умови для шляхів означає обернене формулювання задачі побудови максимального ланцюга: чи можна записати сукупність серединних умов $Z_1 \cup Z_2 \cup \ldots \cup Z_N$ так, щоб розв'язками задачі були всі можливі прості ланцюги від V_a до V_b, які можна побудувати за структурою графа. Порядок обходу вершин ланцюга має значення.

Для всіх п'яти типів серединних умов досліджена їх конструктивна повнота [8, 9] (див. таблицю).

Властивості серединних умов

Тип умови	Повнота для вершин	Повнота для шляхів
тип 1	ні	ні
тип 2	так	ні
тип 3	так	ні
тип 4	ні	ні
тип 5	так	так

З наведеної таблиці випливає, що за використання зв'язаної серединної умови типу 5 можна подати як розв'язок будь-який існуючий ланцюг графа, тобто потенційно – розв'язок будь-якої задачі побудови ланцюга.

Окремі випадки простого ланцюга. Важливим на практиці випадком простого ланцюга є пошук кільцевого шляху, коли крайові умови початку і закінчення руху збігаються $V_a = V_b$. Для визначеного на початку статті випадку неорієнтованого графа з незваженими ребрами або з ребрами однакової ваги рух між кожною визначеною парою сусідніх вершин двосторонній в обох напрямах:

$(V_k, V_m) = (V_m, V_k)$, $d(V_k, V_m) = d(V_m, V_k)$. Тобто шукаємо кільце вершин, рух за яким можливий в обох напрямах [10].

Можливість побудови циклічного ланцюга врахована в алгоритмі [7] і виглядає схематично так:

```
if (((MatrLnk[Ncity, i] == 1) && !CurrentSetVertex.ContainsKey(i))
 // можна перейти до вершини i
 || ((i == FinishVertex) && (MatrLnk[Ncity, FinishVertex] == 1)))
 // дійшли до Vb у випадку Va=Vb (циклічний ланцюг)
{ ... GetNextCity(i,false); // продовжуємо рекурсивно ... }.
```

Рекурсивна функція GetNextCity() перевіряє можливість переходу до наступної вершини *i*, якщо такої ще немає на побудованій частині шляху, або перевіряє окремо перехід (повернення) до початкової вершини V_a. Вона ж виконує перевірку закінчення побудови ланцюга:

```
if (!dirforward && Ncity == FinishVertex) // дійшли до Vb
 { // виконуємо процедуру перевірки придатності маршруту
 TestPathWithMiddleCond(); // відповідно до серединної умови
. . . }.
```

Отже, кільцевий шлях є частиною алгоритму P_0^{r} і не вимагає окремих обчислень. Зокрема, для кільцевого шляху можна застосовувати всі п’ять типів серединних умов. На рис. 3 подано приклад розрахунку кільцевого транспортного руху обласного масштабу.

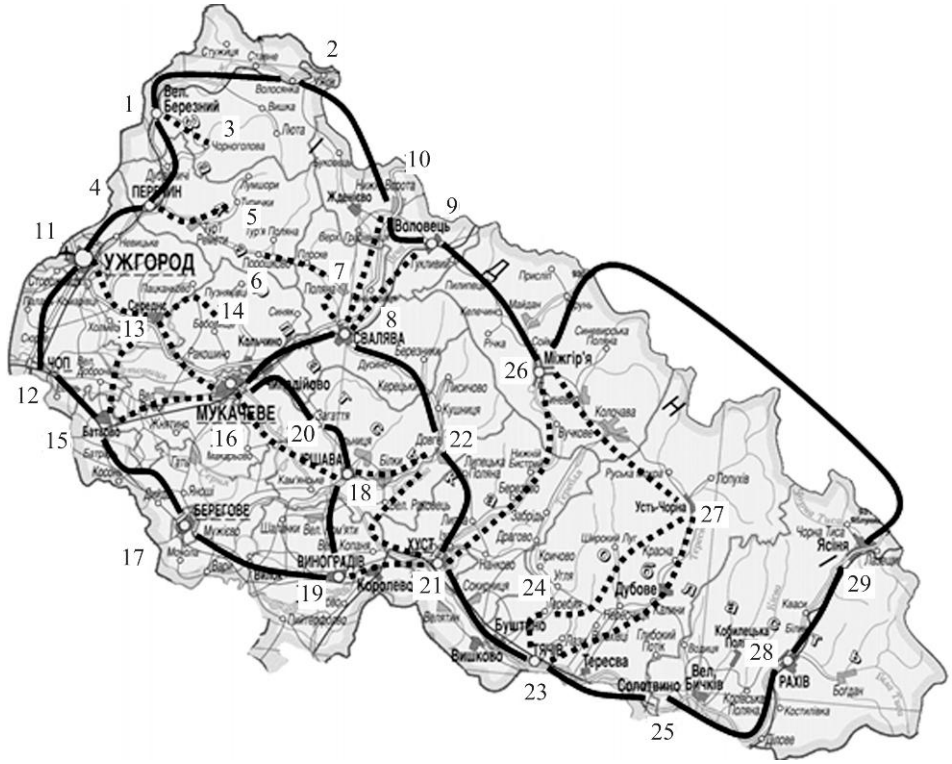

Рис. 3. Приклад розрахунку кільцевого руху для Закарпатської області.

Застосування алгоритму P_0^{r} для $V_a = 16$, $V_b = 16$ і графа, показаного на рис. 3, дало такі результати: граф має 29 вершин; знайдено шлях довжини 22 пункти: Мукачеве–Свалява–Довге–Хуст–Тячів–Солотвино–Рахів–Ясіня–Міжгір’я–

Воловець–Нижні Ворота–Волосянка–Великий Березний–Перечин–Ужгород–Чоп–Батьово–Берегове–Виноградів–Іршава–Загаття–Мукачеве 16–8–22–21–23–25–28–29–26–9–10–2–1–4–11–12–15–17–19–18–20–16; шляхів такої самої довжини є чотири; разом перевірено 2878 шляхів; побудованих шляхів від V_a до V_b 815.

Для заданого графа отримали 4 шляхи однакової довжини 22 пункти: $L(path_1) = L(path_2) = L(path_3) = L(path_4)$. За результат обрано показаний вище шлях, який знайшли найпершим. Кількість сумарно перевірених шляхів 2878 є оцінкою обчислювальної складності алгоритму. Шляхом вважаємо будь-який ланцюг $V_a \Rightarrow V_1 \Rightarrow V_2 \Rightarrow \ldots \Rightarrow V_k$, коли $V_k = V_b$, або не існує ланцюга $V_a \Rightarrow V_1 \Rightarrow V_2 \Rightarrow \ldots \Rightarrow V_k \Rightarrow V_{k+1}$. Іншими словами, вершина V_k є граничною межею побудови чергового ланцюга, від якої немає продовження руху.

Кількість побудованих шляхів від V_a до V_b (815) показує загальну кількість можливих кільцевих шляхів для заданих крайових умов.

Знайдений шлях максимальної довжини має дві важливі властивості: 1) рух по ньому можливий як у вказаному напрямку, так і в зворотному; 2) за початковий і, відповідно, кінцевий пункт руху можна обрати будь-який пункт, який включений до обчисленого максимального шляху; оскільки шлях циклічний, то це дасть такий самий головний результат – довжину шляху і порядок відвідання суміжних вершин.

ВИСНОВКИ

Максимальний простий шлях неповного графа є новою постановкою задачі і дає можливість розширити коло пошукових задач на графах.

1. *Fomin F. V., Heggernes P., Telle J. A.* Graph searching, elimination trees, and a generalization of bandwidth // Lecture Notes in Computer Science. – 2003. – **2751**. – P. 73–85.
2. *Hvalica D.* Searching for a minimal solution subgraph in explicit and/or graphs // Discrete Applied Mathematics. – 2001. – **110**, № 2–3. – P. 213–225.
3. *Finding* Hamilton cycles in robustly expanding digraphs / D. Christofides, P. Keevash, D. Kühn, D. Osthus // J. Graph Algorithms and Applications. – 2012. – **16**, № 2. – P. 335–358.
4. *Haverkort H., Toma L.* I/O-Efficient algorithms on near-planar graphs // J. Graph Algorithms and Applications. – 2011. – **15**, № 4. – P. 503–532.
5. *Kluge S., Brokate M., Reif K.* New complexity results for time-constrained dynamical optimal path problems // J. Graph Algorithms and Applications. – 2010. – **14**, № 2. – P. 123–147.
6. *Frontier* Search / R. E. Korf, W. Zhang, I. Thayer, H. Hohwald // J. of the ACM. – 2005. – **52**, № 5. – P. 715–748.
7. *Черняхівський В. В.* Рекурсивний алгоритм побудови максимального простого ланцюга неповного графа // Вісник Львів. ун-ту. Сер. прикл. матем. та інформ. – 2007. – Вип. 13. – С. 45–50.
8. *Черняхівський В. В.* Ізольовані серединні умови та їхні властивості для задачі максимізації побудови простого ланцюга графа // Вісник Львів. ун-ту. Серія прикл. матем. та інформ. – 2013. – Вип. 20. – С. 109–117.
9. *Черняхівський В. В.* Задачі побудови простого ланцюга графа для зв'язаних серединних умов // Вісник Львів. ун-ту. Серія прикл. матем. та інформ. – 2014. – Вип. 21. – С. 139–147.
10. *Черняхівський В. В.* Циклічний максимальний простий ланцюг неповного графа // Вісник Львів. ун-ту. Серія прикл. матем. та інформ. – 2014. – Вип. 22. – С. 129–135.

Одержано 17.09.2015