

МАРШРУТЫ С ЛОКАЛЬНЫМИ ОГРАНИЧЕНИЯМИ

Т.А. Панюкова

THE PATHS WITH LOCAL RESTRICTIONS

Т.А. Panukova

В данной работе рассмотрена задача покрытия графа минимальным числом цепей, удовлетворяющих заданным локальным ограничениям. Показана возможность распознавания системы переходов, допускающей линейную сложность задачи построения допустимого пути. Построены алгоритмы отыскания в графе допустимого эйлерова цикла за линейное время, либо, если такого цикла нет, – покрытия графа допустимыми цепями.

Ключевые слова: граф, маршрут, цепь, запрещенный переход, покрытие

The research is devoted to the problem of graph covering by minimal number of trails corresponding some local restrictions. The opportunity to recognize the transitions system is observed. The problem of allowed path construction has linear complexity. Algorithm of allowed Eulerian cycle construction is also considered. If graph does not have such a cycle then algorithm constructs covering of graph by allowed trails. This algorithm also runs by linear time.

Keywords: graph, path, trail, forbidden transition, covering

Введение

Многие задачи нахождения маршрутов, удовлетворяющих определенным ограничениям, появились из конкретных практических ситуаций. В задачах раскроя листового материала плоский граф является моделью раскройного плана, а маршрут, покрывающий все ребра, определяет траекторию режущего инструмента. Ограничением является отсутствие пересечения внутренних граней любой начальной части маршрута с ребрами его оставшейся части [1]. При построении систем управления манипуляторами с помощью неориентированного графа отображают всевозможные элементы траектории манипулятора. При этом возникают проблемы построения маршрутов, удовлетворяющих различным ограничениям, например: прямолинейных маршрутов [2]; маршрутов, в которых следующее ребро определяется заданным циклическим порядком на множестве ребер, инцидентных текущей вершине [3 – 5]; маршрутов, в которых часть ребер следует пройти в заданном порядке [5].

Ограничения на порядок обхода вершин и ребер графа можно классифицировать на локальные, когда следующее ребро в маршруте определяется условиями, заданными в текущей вершине или на текущем ребре [2, 3 – 5, 7, 8] и на глобальные (эйлеровы, гамильтоновы циклы, бинаправленные двойные обходы и т.д.). Большинство работ посвящено алгоритмам с локальными ограничениями на порядок обхода ребер. В данной работе рассмотрена задача покрытия графа минимальным числом цепей, удовлетворяющих заданным локальным ограничениям.

1. Алгоритм построения допустимой цепи

Обобщение большинства частных случаев задачи построения простой цепи с локальными ограничениями и анализ вычислительной сложности данной проблемы даны С.Зейдером [7]. Приведем основные определения и результаты данной работы.

Ограничимся рассмотрением *конечных простых* графов. Множество вершин и множество ребер графа G будем обозначать соответственно через $V(G)$ и $E(G)$. Для вершины $v \in V(G)$ определим $E_G(v)$, множество всех ребер графа G , инцидентных вершине v . Степень вершины v будем обозначать как $d(v)$; для $d > 0$ положим $V_d(G) := \{v \in V(G) \mid d(v) = d\}$. Будем писать $H \leq G$, если H – вершинно-индуцированный подграф графа G , т.е. подграф, полученный из графа G отбрасыванием некоторого множества вершин и всех ребер, инцидентных вершинам этого множества, и только их.

Ограничения на маршруты в графе G можно сформулировать в терминах графа разрешенных переходов.

Определение 1. Пусть G – граф. **Графом переходов** $T_G(v)$ вершины $v \in V(G)$ будем называть граф, вершинами которого являются ребра, инцидентные вершине v , т.е. $V(T_G(v)) = E_G(v)$, а множество ребер – допустимые переходы.

Определение 2. **Системой разрешенных переходов** (или короче, **системой переходов**) T_G будем называть множество $\{T_G(v) \mid v \in V(G)\}$, где $T_G(v)$ – граф переходов в вершине v .

Определение 3. Путь $P = v_0, e_1, v_1, \dots, e_k, v_k$ в графе G является **T_G -совместимым**, если $e_i, e_{i+1} \in E(T_G(v_i))$ для каждого i ($1 \leq i \leq k-1$).

Теорема 1. [С. Зейдер]. Если все графы переходов принадлежат либо классу M полных многодольных графов, либо классу P паросочетаний, то задача построения T_G -совместимой цепи является разрешимой за время $O(|E(G)|)$. В противном случае данная задача является NP -полной.

Если система переходов вершины $v \in V(G)$ является паросочетанием, то задача сводится к задаче для графа

$$G' : V(G') = V(G) \setminus \{v\}, \quad E(G') = (E(G) \setminus E_G(v)) \cup \{\{v_i v_j\} : \{v_i v, v v_j\} \in E(T_G(v))\}.$$

Если для любой вершины $v \in V(G)$ граф $T_G(v)$ является полным многодольным графом, то цепь можно построить с помощью следующего алгоритма.

Алгоритм T_G -СОВМЕСТИМЫЙ ПУТЬ

Входные данные:

- граф $G = (V, E)$;
- вершины x, y , между которыми требуется найти цепь без запрещенных переходов;
- система переходов $T_G : (\forall v \in V(G)) \ T_G(v) \in M$.

Выходные данные:

- последовательность ребер, определяющая T_G -совместимый путь между вершинами x и y , либо сообщение об его отсутствии.

Шаг 1. Если вершина x или вершина y является изолированной, останов: пути нет.

Шаг 2. Удалить из графа G изолированные вершины.

Шаг 3. Построить вспомогательный граф G' следующим образом (рис. 1):

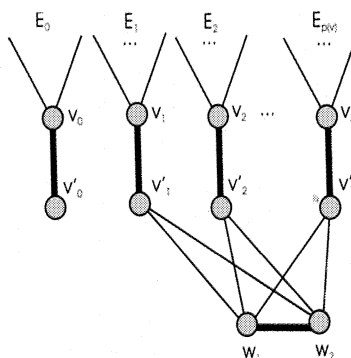


Рис. 1. Иллюстрация построения вспомогательного графа G'

- каждую вершину $v \in V(G)$ расщепить на вершины $v_1, v_2, \dots, v_{p(v)}$, где $p(v)$ – число долей графа $T_G(v)$. Вершине v_p инцидентны ребра соответствующей доли графа $T_G(v)$ и одна дополнительная вершина $v'_{p(v)}$;
- добавить две новые вершины $w_1(v)$ и $w_2(v)$, ребро $w_1(v)w_2(v)$, и ребро $v'_{p(v)}w_j(v)$ для каждой доли графа $T_G(v)$, $1 \leq j \leq 2$.

Шаг 4. Построить первоначальное паросочетание в графе G'

$$M(G') = \bigcup_{v \in V(G)} \left(\bigcup_{p=1,2,\dots,p(v)} \{v_p v'_p\} \cup \{w_1(v)w_2(v)\} \right).$$

Шаг 5. Искать чередующуюся последовательность между вершинами x и y , увеличивающую мощность паросочетания в графе G' . Если такую последовательность найти не удастся – останов (паросочетание $M(G')$ имеет максимальную мощность, а граф не имеет T_G -совместимого пути). В противном случае все ребра данного увеличивающего пути за исключением ребер, добавленных при построении графа G' , образуют T_G -совместимую цепь между вершинами x и y . Останов.

Заметим, что в работе [7] открытым остался вопрос распознавания многодольности графов $T_G(v)$, а также проблема построения допустимого маршрута или множества маршрутов, покрывающих все ребра исходного графа.

Покажем на примере графа G , представленного на рис. 2, что алгоритм **T_G -СОВМЕСТИМЫЙ ПУТЬ** не может быть использован для построения маршрутов, покрывающих все ребра графа G . Допустим для графа задана следующая система переходов T_G : $\{\{v_2 v_1\}, \{v_1 v_5\}\}, \{\{v_6 v_1\}, \{v_1 v_4\}\}, \{\{v_4 v_3\}, \{v_3 v_7\}\}, \{\{v_8 v_3\}, \{v_3 v_2\}\}, \{\{v_3 v_2\}, \{v_2 v_8\}\}, \{\{v_5 v_2\}, \{v_2 v_1\}\}, \{\{v_1 v_4\}, \{v_4 v_6\}\}, \{\{v_7 v_4\}, \{v_4 v_3\}\}, \{\{v_2 v_5\}, \{v_5 v_8\}\}, \{\{v_2 v_8\}, \{v_8 v_5\}\}, \{\{v_3 v_8\}, \{v_8 v_7\}\}, \{\{v_3 v_7\}, \{v_7 v_8\}\}, \{\{v_4 v_7\}, \{v_7 v_6\}\}, \{\{v_4 v_6\}, \{v_6 v_7\}\}, \{\{v_1 v_6\}, \{v_6 v_5\}\}, \{\{v_1 v_5\}, \{v_5 v_6\}\}.$

Граф G' , необходимый для нахождения T_G -совместимого пути между вершинами v_1 и v_7 , построение которого описано на шаге 3 алгоритма, приведен на рис. 3.

Первоначальное паросочетание $M(G')$ выделено на рисунке жирными линиями. Для данного паросочетания чередующейся увеличивающей последовательностью ребер является $\{v_{1,1} v_{5,2}\}, \{v_{5,2} v'_{5,2}\}, \{v'_{5,2} w_{5,2}\}, \{w_{5,2} w_{5,1}\}, \{w_{5,1} v'_{5,1}\}, \{v'_{5,1} v_{5,1}\}, \{v_{5,1} v_{6,2}\}, \{v_{6,2} v'_{6,2}\}, \{v'_{6,2} w_{6,2}\}, \{w_{6,2} w_{6,1}\}, \{w_{6,1} v'_{6,1}\}, \{v'_{6,1} v_{6,1}\}, \{v_{6,1} v_{7,2}\}$. Ребра этой последовательности, не

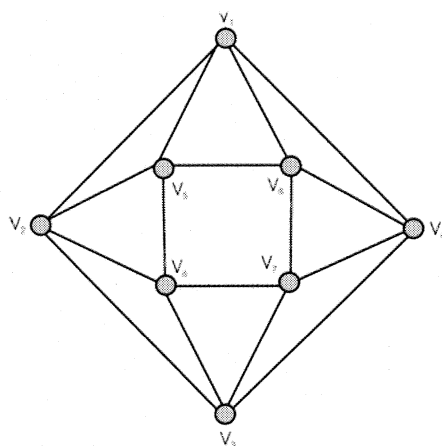


Рис. 2. Пример графа и системы переходов

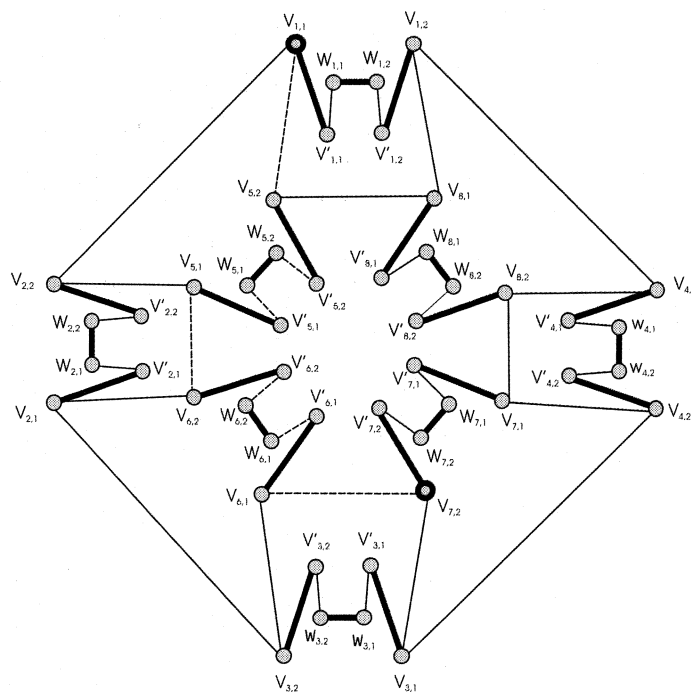


Рис. 3. Граф G' , полученный с помощью вспомогательных построений из графа G

вошедшие в первоначальное паросочетание, изображены пунктирной линией. Эти ребра образуют множество

$$\{v_{1,1}v_{5,2}\}, \{v'_{5,2}w_{5,2}\}, \{w_{5,1}v'_{5,1}\}, \{v_{5,1}v_{6,2}\}, \{w_{6,1}v'_{6,1}\}, \{v_{6,1}v_{7,2}\}.$$

Все ребра данного множества, принадлежащие графу G , т.е. $\{v_1v_5\}, \{v_5v_6\}, \{v_6v_7\}$ образуют T_G -совместимый путь из вершины v_1 в вершину v_7 .

С помощью алгоритма **T_G -СОВМЕСТИМЫЙ ПУТЬ** возможно построение только простой цепи между двумя различными вершинами (т.е. цепи, в которых любая вершина встречается ровно один раз).

Однако в общем случае непосредственное применение данного алгоритма не позволяет решить задачу нахождения T_G -совместимого маршрута, содержащего максимальное число ребер. Действительно, паросочетание максимальной мощности в графе G' не может содержать пары ребер, образующих запрещенный переход, т.к. они инцидентны одной общей вершине графа G' . В то же время, в общем случае может существовать T_G -совместимый маршрут, содержащий такую пару ребер.

Например, в графе G , приведенном на рис. 2, маршрут

$$\{v_2v_1\}, \{v_1v_4\}, \{v_4v_8\}, \{v_8v_1\}, \{v_1v_5\}, \{v_5v_2\}$$

принципиально не может быть получен с помощью построения паросочетания максимального веса в графе G' . Этот маршрут начинается с ребра v_2v_1 , а заканчивается ребром v_5v_2 . Эти ребра образуют запрещенный переход $\{v_5v_2\}, \{v_2v_1\}$, следовательно, в графе G' не существует чередующегося пути, содержащего оба эти ребра.

Таким образом, открытым остался вопрос распознавания многодольности графов $T_G(v)$, а также проблема построения допустимого маршрута или множества маршрутов, покрывающих все ребра исходного графа.

При разработке алгоритма построения допустимой цепи возник вопрос, можно ли заменить графы переходов некоторой ориентацией исходного графа. В качестве примера рассмотрим граф, представленный на рис. 4(а). Пусть для этого графа заданы следующие допустимые переходы $\{\{v_2, v_1\}, \{v_1, v_3\}\}, \{\{v_3, v_1\}, \{v_1, v_4\}\}, \{\{v_1, v_4\}, \{v_4, v_2\}\}, \{\{v_1, v_4\}, \{v_4, v_3\}\}, \{\{v_4, v_3\}, \{v_3, v_1\}\}, \{\{v_3, v_2\}, \{v_2, v_1\}\}$. Попытаемся представить их в виде дуг орграфа, как показано на рис. 4(б).

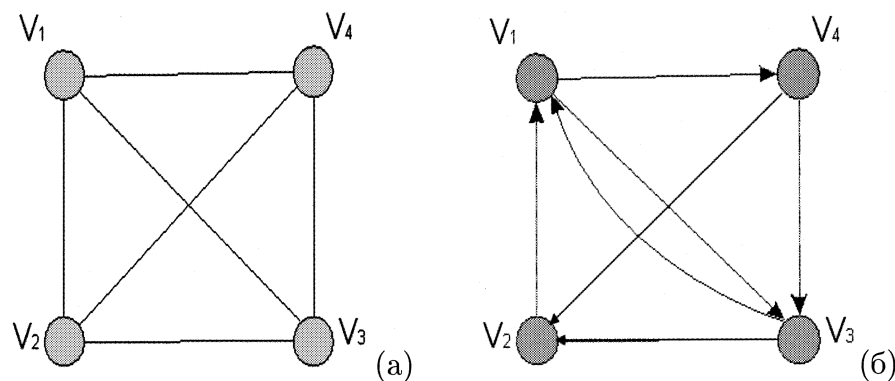


Рис. 4. Пример графа

Заметим, что в построенном орграфе присутствуют и переходы, которые не были заданы системой допустимых переходов (например, $\{\{v_2, v_1\}, \{v_1, v_4\}\}$). Для иллюстрации работы предложенного алгоритма построим вспомогательный граф в соответствии с шагом 3 (рис. 5(а)).

В частности, вершину v_1 нам потребовалось расщепить на две, чтобы показать возможные переходы в ней, как показано на рис. 5(б). Таким образом, представление в виде орграфа не гарантирует существования возможного перехода.

Этот пример показывает, что для нахождения алгоритма построения допустимой цепи невозможно представить систему переходов в виде дуг орграфа, т.е. для каждой вершины необходимо строить соответствующий граф переходов [9].

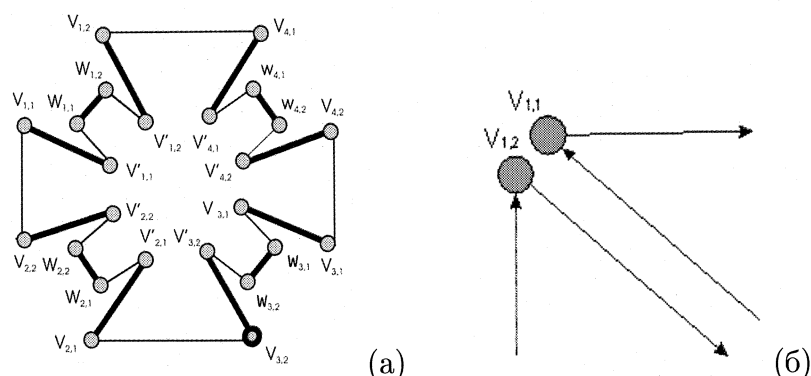


Рис. 5. Расщепленный граф для примера с рис. 4 и возможные переходы для вершины v_1

2. Алгоритм построения допустимой цепи

В предыдущем разделе были сформулированы ограничения на допустимость маршрутов в терминах системы разрешенных переходов [7] и показано, что задача построения допустимого пути в графе G разрешима за полиномиальное время, если система переходов T_G содержит только паросочетания и полные многодольные графы. Распознавание принадлежности графа разрешенных переходов классу паросочетаний тривиально. Для распознавания принадлежности графа переходов классу полных многодольных графов целесообразно использовать понятие системы разбиения [3]–[5], [8].

Понятие системы разбиения используется для определения допустимой цепи в терминах запрещенных переходов.

Определение 4. Пусть дан граф $G = (V, E)$. Пусть $P_G(v)$ – некоторое разбиение множества $E_G(v)$. Системой разбиения графа G будем называть систему множеств $P_G := \{P_G(v) \mid v \in V(G)\}$.

Определение 5. Пусть $p \in P_G(v)$, $e, f \in p$. Цепь, не содержащую переходов $e \rightarrow v \rightarrow f$ и $f \rightarrow v \rightarrow e$, будем называть P_G -совместимой, а переходы $e \rightarrow v \rightarrow f$ и $f \rightarrow v \rightarrow e$ – запрещенными.

Заметим, что граф разрешенных переходов $T_G(v)$ однозначно определяет граф запрещенных переходов $\overline{T}_G(v)$, который является дополнением графа разрешенных переходов до полного графа. Таким образом, с помощью определений 1–3 можно поставить задачу с любым графом разрешенных (запрещенных) переходов.

Напротив, граф разрешенных переходов, определяемый с помощью системы разбиения P_G , не может быть произвольным, а принадлежит классу M полных многодольных графов: элементы разбиения $P_G(v)$ определяют доли графа $T_G(v) \in M$, а множество его ребер

$$E(T_G(v)) = \{e, f \in E_G(v) : (\forall p \in P_G(v)) \{e, f\} \not\subset p\}.$$

Графом запрещенных переходов $\overline{T}_G(v)$ в данном случае будет являться набор из $|P_G(v)|$ клик, этот факт может быть использован для распознавания принадлежности $T(v) \in M$ с помощью следующего алгоритма.

Алгоритм РАЗМЕТКА

Входные данные: граф переходов $T_G(v)$.

Шаг 1. Объявить все вершины графа $T_G(v)$ непомеченными. Положить $l = 1$.

Шаг 2. Пока список непомеченных вершин не пуст, выполнять шаги 3, 4 и 5. В противном случае – останов: граф $T_G(v)$ является многодольным и вершины, принадлежащие одному элементу разбиения множества вершин, имеют одинаковые пометки.

Шаг 3. Найти некоторую непомеченную вершину v . Присвоить ей пометку l .

Шаг 4. Применить волновой алгоритм для присваивания пометки l всем вершинам, достижимым из вершины v в графе $\overline{T_G(v)}$. Очевидно, что все помеченные на данном шаге вершины будут принадлежать одной компоненте связности графа $\overline{T_G(v)}$.

Шаг 5. Если в выделенной компоненте связности любая пара вершин является смежной, то найденная компонента связности является кликой. Положить $l = l + 1$ и перейти к выполнению шага 3. В противном случае – останов: граф $T_G(v)$ не является многодольным.

Оценим сложность приведенного алгоритма. Расстановка пометок в конкретной компоненте связности T_k составляет величину $O(|E(T_k)|)$. Проверка, является ли данная компонента связности кликой, также требует не более $O(|E(T_k)|)$ операций. Таким образом, сложность алгоритма РАЗМЕТКА равна

$$O\left(\sum_{\forall k} |E(T_k)|\right) = O(|E(T)|).$$

Как было отмечено, алгоритм С. Зейдера в общем случае не позволяет строить допустимые цепи максимальной длины. Особый интерес представляют допустимые эйлеровы цепи. Необходимое и достаточное условие существования P_G -совместимых цепей дает следующая теорема [8].

Теорема 2. [А. Коциг]. *Связный эйлеров граф G имеет P_G -совместимую эйлерову цепь тогда и только тогда, когда*

$$(\forall v \in V) (\forall p \in P_G(v)) \left(|p| \leq \frac{1}{2} d_G(v) \right).$$

Очевидно, что сложность проверки условия существования P_G -совместимой эйлеровой цепи не превосходит величины $O(|E(G)|)$. Ниже приведен алгоритм построения совместимой цепи.

Алгоритм P_G -СОВМЕСТИМАЯ ЭЙЛЕРОВА ЦЕПЬ

Входные данные:

- эйлеров граф $G = (V, E)$,
- система переходов $P_G(v) \forall v \in V(G)$.

Выходные данные:

- допустимый эйлеров цикл G_{k+1} .

Шаг 1. Положить $k = 0$, $G_k = G$.

Шаг 2. Найти вершину v , у которой $d_{G_k}(v) > 2$.

Шаг 3. Найти класс $C_1 \in P_{G_k}(v) : |C_1| = \{\max |C| | C \in P_{G_k}(v)\}$.

Шаг 4. Найти любые ребра $e_1(v) \in C_1$ и $e_2(v) \in E_{G_k}(v) - C_1$. Если множество $E_{G_k}(v) - C_1 = \emptyset$, останов: P_G -совместимой эйлеровой цепи не существует. В противном случае перейти на шаг 5.

Шаг 5. Построить граф G_{k+1} , отщипив от вершины v вершину $\widehat{v}$, которой инцидентны только ребра e_1 и e_2 . Остальные ребра оставить инцидентными вершине v .

Шаг 6. Пусть класс $C_2 \in P_{G_k}(v)$ содержит ребро $e_2(v)$. Найти

$$P_{G_k}^-(v) := P_{G_k}(v) - \{C_1, C_2\},$$

$$P'_{G_{k+1}}(v) := \begin{cases} P_{G_k}^-(v) \cup \{C_1 - \{e_1(v)\}, C_2 - \{e_2(v)\}\}, & \text{если } |C_2| > 1, \\ P_{G_k}^-(v) \cup \{C_1 - \{e_1(v)\}\}, & \text{если } |C_1| > |C_2| = 1, \\ P_{G_k}^-(v), & \text{если } |C_1| = |C_2| = 1, \end{cases}$$

$$P_{G_{k+1}} = \bigcup_{x \in V(G_{1,2})} P'_{G_{k+1}}(x).$$

Шаг 7. Определить значение $\sigma(G_{k+1}) = 2(|E(G_{k+1})| - |V(G_{k+1})|)$.

Шаг 8. Если $\sigma(G_{k+1}) > 0$, положить $k = k + 1$, перейти на шаг 2, для графа G_{k+1} . В противном случае – останов: построенный граф G_{k+1} является эйлеровой цепью, не содержащей запрещенных переходов.

Теорема 3. Алгоритм P_G -СОВМЕСТИМАЯ ЭЙЛЕРОВА ЦЕПЬ корректно решает задачу построения $P(G)$ -совместимой эйлеровой цепи.

Доказательство. Если для некоторого k , такого что $C'' \in P'_{G_{k+1}}(v)$, выполнено неравенство $|C''| > |C_1 - \{e_1(v)\}|$, то $C'' \in P_{G_{k+1}}(v)$ и $|C_2| \leq |C''| = |C_1| \leq \frac{1}{2}d_{G_k}(v) - 1 = \frac{1}{2}d_{G_{k+1}}(v)$. На основании этого факта можно заключить, что $|C| \leq \frac{1}{2}d_{G_{k+1}}(v)$ для каждой вершины $v \in V(G_{k+1})$ и каждого класса $C \in P_{G_{k+1}}(v) \subset P_{G_{k+1}}$. При этом величина $\sigma(G_{k+1}) = |E(G_{k+1})| - |V(G_{k+1})| < \sigma(G_k)$. Если граф G_{k+1} является циклом, то число ребер в нем и число вершин совпадает, т.е. в данном случае $\sigma(G_{k+1}) = 0$. Если же граф G_{k+1} является цепью, то число вершин превышает число ребер на 2, следовательно, в данном случае $\sigma(G_{k+1}) = -2$. Если же на некотором этапе для $e_1(v) \in C_1$ не удалось найти $e_2(v) \in E_{G_k}(v) - C_1$, это значит, что $|C_1| > d(v)/2$, т.е. не выполнены необходимые и достаточные условия существования эйлерова цикла (теорема Коцига). Из этих фактов следует корректность выполнения алгоритма. $\square$

Оценим вычислительную сложность предложенного алгоритма. Выполнения шагов 2, 5, 6 и 7 можно организовать с использованием не более $O(1)$ операций (за счет специальных структур данных). Выполнения же шагов 3 и 4 можно организовать с использованием не более $O(d_{G_k}(v))$ операций. Цикл алгоритма будет повторен не более, чем $\sigma(G)$ раз. В итоге имеем, что алгоритм потребует число операций не более

$$O\left(\sum_{k=0,1,\dots,\sigma(G)} d_{G_k}(v_k)\right) = O(|E(G)| \cdot |V(G)|).$$

Таким образом, приведенные алгоритмы разрешимы за полиномиальное время и могут быть легко реализованы с помощью стандартных вычислительных средств.

Рассмотрим задачу покрытия графа допустимыми цепями. Будем считать, что система переходов T_G содержит только паросочетания и полные многодольные графы.

Алгоритм ПОКРЫТИЕ T_G -ДОПУСТИМЫМИ ЦЕПЯМИ

Входные данные:

- граф $G = (V, E)$,
- графы переходов $T_G(v) \forall v \in V(G)$.

Выходные данные:

- набор маршрутов $T^i, i = 1, 2, \dots, k$, покрывающих граф G , где $m = 2k$ – число вершин нечетной степени.

Шаг 1. Пусть $U = \{v \in V(G) : T_G(v) - \text{паросочетание}\}$. Сделать редукцию графа G до графа G' :

$$V(G') = V(G) \setminus U,$$

$$E(G') = \left(E(G) \setminus \bigcup_{v \in U} E_G(v) \right) \cup \left\{ \bigcup_{v \in U} \{ \{v_i v_j\} : \{v_i v, v v_j\} \in T_G(v) \} \right\},$$

графы $T_G(v)$ редуцировать до графов $T_{G'}(v)$ заменой всех вхождений вершин $u \in U$: $vu, wu \in E_{T_G}(u)$ вершиной w .

Шаг 2. Достроить граф G' до G^* введением дополнительной вершины v^* , смежной всем вершинам нечетной степени графа G' . Систему переходов $T_{G'}(v)$ модифицировать до системы переходов T_{G^*} введением для всех $v \in V'(G) : d(v) \equiv 1 \pmod{2}$ в граф переходов $T_{G^*}(v)$ вершины vv^* , смежной всем вершинам в графе $T_{G^*}(v)$.

Шаг 3. Для всех таких вершин $v \in V(G)$, что $\exists p \in P(v) : |p| > d(v)/2$, ввести $2|p| - d(v)$ дополнительных ребер $(vv^*)_i, i = 1, 2, \dots, 2|p| - d(v)$ в граф G^* . Модифицировать граф переходов $T_{G^*}(v)$ введением вершин $(vv^*)_i$, смежных всем вершинам исходного графа $T_{G^*}(v)$ и только им.

Шаг 4. Найти в G^* T_{G^*} -совместимый эйлеров цикл T^* .

Шаг 5. Построить покрытие T' графа G' цепями, удалив из T^* ребра (vv^*) .

Шаг 6. Модифицировать маршруты из T' до маршрутов из T добавлением вершин $u \in U$, удаленных на шаге 1.

Шаг 7. Останов.

Теорема 4. Алгоритм **ПОКРЫТИЕ T_G -ДОПУСТИМЫМИ ЦЕПЯМИ** корректно решает задачу минимального покрытия графа T_G -допустимыми цепями. Его сложность не превосходит величины $O(|E(G)| \cdot |V(G)|)$.

Доказательство. В результате выполнения шага 1 приходим к задаче для полного многодольного графа G' . Данное преобразование возможно выполнить, используя не более $O(|E(G)|)$ операций.

В результате выполнения шага 2 получаем задачу для эйлерова графа, в каждой вершине v которого граф переходов $T_{G^*}(v)$ является полным и многодольным. Введенная в граф $T_{G^*}(v)$ дополнительная вершина vv^* является отдельным элементом разбиения в $P_{G^*}(v)$.

На шаге 3 проверяется выполнение необходимых и достаточных условий существования допустимого эйлерова цикла (теорема Коцига). Во всех вершинах, где условия теоремы Коцига не выполнены, в граф G^* добавляются мультиребра $(vv^*)_i, i = 1, 2, \dots, 2|p(v)| - d(v)$. Также модифицируется и система разбиения $P_{G^*}(v)$ добавлением элемента разбиения, содержащего все ребра $(vv^*)_i$. Такие модификации также выполняются за время, не превосходящее $O(|V(G)| \cdot |E(G)|)$.

В результате проведенных модификаций граф G^* будет эйлеровым, а его система разбиения будет удовлетворять теореме Коцига.

Для построения допустимого эйлерова цикла, содержащего и дополнительные ребра, смежные v^* , на шаге 4 требуется не более $O(|V(G)| \cdot |E(G)|)$ операций.

На шаге 5 получим $l = d(v^*)$ простых цепей удалением ребер, инцидентных вершине v^* , которые были добавлены на шагах 2 и 3. Все полученные таким образом цепи будут $P_{G'}$ -допустимыми в графе G' . Сложность этого этапа составляет величину $O(|E(G)|)$.

На шаге 6 происходит добавление удаленных на шаге 1 вершин, что также требует не более чем $O(|E(G)|)$ операций вставки.

В результате выполненных операций получим покрытие графа $l + 1$ цепями за время $O(|V(G)| \cdot |E(G)|)$. Предположение существования покрытия с меньшим числом цепей приводит к противоречию с теоремой Коцига. $\square$

Заключение

Возможно распознавание системы переходов, которая позволяет решить задачу построения допустимого пути за линейное время. В эйлеровом графе G возможно отыскание P_G -допустимого эйлерова цикла или установление его отсутствия за время $O(|V(G)| \cdot |E(G)|)$ с помощью разработанного алгоритма **P_G -СОВМЕСТИМАЯ ЭЙЛЕРОВА ЦЕПЬ**. Покрытие графа G допустимыми цепями также возможно за время $O(|V(G)| \cdot |E(G)|)$ с помощью алгоритма **ПОКРЫТИЕ T_G -ДОПУСТИМЫМИ ЦЕПЯМИ**.

Литература

1. Panioukova, T.A. Algorithms for Construction of Ordered Enclosing Traces in Planar Eulerian Graphs / T.A. Panioukova, A.V. Panyukov // The International Workshop on Computer Science and Information Technologies' 2003, Proceedings of Workshop, Ufa, September 16 – 18, 2003/ Ufa State Technical University. – Ufa, 2003. – Vol. 1. – P. 134 – 138.
2. Pisanski, T. Straight-ahead walks in Eulerian graphs / T. Pisanski, T.W. Tucker, A. Zitnik // Discrete Mathematics, 2004. – №. 281. – P. 237 – 246.
3. Фляйшнер, Г. Эйлеровы графы и смежные вопросы / Г. Фляйшнер. – М.: Мир, 2002. – 335 с., ил.
4. Fleischner H. Eulerian Graphs and Related Topics / H. Fleischner. – Part 1, Vol.2 – Ann. Discrete Mathematics, 1991. – № 50.
5. Fleischner, H. Eulerian Graphs / H. Fleischner, L.W. Beineke, R.J. Wilson // Selected Topics in Graph Theory 2, Academic Press, London-NewYork, 1983. – P. 17 – 53.
6. Chebikin, D. On k-edge-ordered graphs / D. Chebikin // Discrete Mathematics, 2004. – № 281. – P. 115 – 128.
7. Szeider, S. Finding Paths in Graphs Avoiding Forbidden Transitions / S. Szeider // Discrete Applied Mathematics, 2003. – № 126. – P. 261 – 273.
8. Kotzig, A. Moves Without Forbidden Transitions in a Graph / A. Kotzig // Mat.-Fiz. Casopis 18, 1968. – № 1. – С. 76 – 80.
9. Панюкова, Т.А. Построение совместимых цепей в графах / Т.А. Панюкова, В.Ф. Миросов // Проблемы теоретической и прикладной математики: тр. 39-й Регион. молодеж. конф. Екатеринбург, 2008. – С. 38 – 43.

Кафедра «Экономико-математические методы и статистика»
Южно-Уральский государственный университет
kwark@mail.ru

Поступила в редакцию 25 декабря 2009 г.