MODELING AND ANALYSIS OF INFORMATION SYSTEMS, VOL. 32, NO. 2, 2025
journal homepage: www.mais-journal.ru

DISCRETE MATHEMATICS IN RELATION TO COMPUTER SCIENCE

Exact Algorithm for the Problem of the Minimum Complete Spanning Tree of a Divisible Multiple Graph

A. V. Smirnov[1]

DOI: 10.18255/1818-1015-2025-2-132-149

[1]P.G. Demidov Yaroslavl State University, Yaroslavl, Russia

MSC2020: 05C85, 05C65, 05C05
Research article
Full text in Russian

Received April 23, 2025
Revised May 27, 2025
Accepted May 28, 2025

We study undirected multiple graphs of any natural multiplicity $k > 1$. There are edges of three types: ordinary edges, multiple edges, and multi-edges. Each edge of the last two types is a union of k linked edges, which connect 2 or $(k + 1)$ vertices, correspondingly. The linked edges should be used simultaneously. Divisible graphs form a special class of multiple graphs. The main peculiarity of them is a possibility to divide the graph into k parts, which are adjusted on the linked edges and which have no common edges. Each part is an ordinary graph.

The multiple tree is a multiple graph with no multiple cycles. The number of edges may be different for multiple trees with the same number of vertices. Also we can consider spanning trees of a multiple graph. A spanning tree is complete if a multiple path joining any two selected vertices exists in the tree if and only if such a path exists in the initial graph. The problem of the minimum complete spanning tree of a multiple graph is NP-hard even in the case of a divisible graph. In this article, we obtain an exact algorithm for the problem of the minimum complete spanning tree of a divisible multiple graph. Also we define a subclass of divisible graphs, for which the algorithm runs in polynomial time.

Keywords: multiple graph; divisible graph; multiple tree; complete spanning tree; exact algorithm

INFORMATION ABOUT THE AUTHORS

Smirnov, Alexander V. (corresponding author)	ORCID iD: 0000-0002-0980-2507. E-mail: alexander_sm@mail.ru PhD, Associate Professor, Department of Theoretical Computer Science

Funding: Yaroslavl State University (project VIP-016).

For citation: A. V. Smirnov, “Exact algorithm for the problem of the minimum complete spanning tree of a divisible multiple graph”, *Modeling and Analysis of Information Systems*, vol. 32, no. 2, pp. 132–149, 2025. DOI: 10.18255/1818-1015-2025-2-132-149.

© Smirnov A. V., 2025
This is an open access article under the CC BY license (https://creativecommons.org/licenses/by/4.0/).

МОДЕЛИРОВАНИЕ И АНАЛИЗ ИНФОРМАЦИОННЫХ СИСТЕМ, ТОМ 32, № 2, 2025
сайт журнала: www.mais-journal.ru

DISCRETE MATHEMATICS IN RELATION TO COMPUTER SCIENCE

Точный алгоритм для задачи о минимальном полном остовном дереве в делимом кратном графе

А. В. Смирнов[1]

DOI: 10.18255/1818-1015-2025-2-132-149

[1]Ярославский государственный университет им. П.Г. Демидова, Ярославль, Россия

УДК 519.17+519.161
Научная статья
Полный текст на русском языке

Получена 23 апреля 2025 г.
После доработки 27 мая 2025 г.
Принята к публикации 28 мая 2025 г.

В статье рассматриваются неориентированные кратные графы произвольной натуральной кратности $k > 1$. Кратный граф содержит ребра трех типов: обычные, кратные и мультиребра. Ребра последних двух типов представляют собой объединение k связанных ребер, которые соединяют 2 или $(k + 1)$ вершину соответственно. Связанные ребра могут использоваться только согласованно. Делимые графы представляют собой специальный класс кратных графов. Их основная особенность состоит в возможности разделить граф на k частей, которые будут согласованы на связанных ребрах и не будут иметь общих ребер. Каждая часть является обычным графом.

Кратное дерево представляет собой кратный граф без кратных циклов. Количество ребер может быть разным для кратных деревьев с одинаковым количеством вершин. Также можно рассмотреть остовные деревья в кратном графе. Остовное дерево является полным, если кратный путь, соединяющий любые две выбранные вершины, существует в дереве тогда и только тогда, когда такой путь существует в исходном графе. Задача о минимальном полном остовном дереве в кратном графе NP-трудна даже в случае делимого графа. В данной статье мы получим точный алгоритм для задачи о минимальном полном остовном дереве в делимом кратном графе. Также мы определим подкласс делимых графов, для которых алгоритм будет выполняться за полиномиальное время.

Ключевые слова: кратный граф; делимый граф; кратное дерево; полное остовное дерево; точный алгоритм

ИНФОРМАЦИЯ ОБ АВТОРАХ

Смирнов, Александр Валерьевич (автор для корреспонденции) | ORCID iD: 0000-0002-0980-2507. E-mail: alexander_sm@mail.ru
Канд. физ.-мат. наук, доцент, кафедра теоретической информатики

Финансирование: ЯрГУ (проект VIP-016).

Для цитирования: A. V. Smirnov, “Exact algorithm for the problem of the minimum complete spanning tree of a divisible multiple graph”, *Modeling and Analysis of Information Systems*, vol. 32, no. 2, pp. 132–149, 2025. DOI: 10.18255/1818-1015-2025-2-132-149.

© Смирнов А. В., 2025
Эта статья открытого доступа под лицензией CC BY license (https://creativecommons.org/licenses/by/4.0/).

Введение

В данной статье мы рассмотрим *остовные деревья* в делимом кратном графе. Ранее мы определили кратный граф и кратное дерево произвольной натуральной кратности $k > 1$ (см. [1—3]). Кратные графы содержат три типа ребер (обычные, кратные и мультиребра) и являются обобщением обычных графов — по сути, обычный граф имеет кратность $k = 1$. Делимые графы представляют собой специальный класс кратных графов. Их основная особенность состоит в возможности разделить граф на k частей, которые будут согласованы на связанных ребрах и не будут иметь общих ребер. Каждая часть является обычным графом.

Также мы рассмотрели обобщения для кратных графов ряда классических задач теории графов. В частности, в статье [4] мы доказали что задача о минимальном остовном дереве является NP-трудной как для произвольного, так и для делимого кратного графа, если кратность $k \geqslant 3$.

Среди других известных обобщений графов наиболее близкими нам концепциями являются мультиграфы, гиперграфы (см., например, [5, 6]), а также метаграфы (см. [7, 8]). Действительно, как и в мультиграфах, в кратных графах допускается наличие нескольких ребер между парой вершин (набор таких ребер мы будем в дальнейшем называть *кратным ребром*), однако в случае кратного графа количество таких ребер должно быть строго равным k. В кратных графах присутствуют *мультиребра*, соединяющие между собой $(k + 1)$ вершину. Но в отличие от гиперребер гиперграфа, мультиребро представляется в виде k связанных ребер, имеющих один общий конец, причем все эти k ребер должны использоваться согласованно, то есть все характеристики ребра должны иметь одинаковые значения для всех связанных ребер мультиребра. Кроме того, мультиребро может включаться в какую-либо новую структуру только как единое целое. По сути, понятие мультиребра близко понятию ребра между вершиной и метавершиной в метаграфе. При этом в метаграфе, напомним, метапуть между двумя метавершинами фактически моделирует причинно-следственные связи в некоторой предметной области. Однако в кратном графе используется принципиально иной подход к определению пути: *кратный путь* должен состоять ровно из k обычных путей, проходящих по обычным ребрам, а также по связанным ребрам кратных и мультиребер; при этом пути должны быть согласованы (одинаковы) на кратных и мультиребрах. Поэтому кратный граф нельзя считать частным случаем метаграфа.

Отметим, что частным случаем кратного графа является кратная сеть (см. [9, 10]), представляющая собой ориентированный кратный граф с одним источником и одним стоком. Задача о наибольшем потоке в кратной сети обобщает классическую задачу (см. [11]) и имеет ряд приложений в сфере экономики, управления, финансов.

В частности, кратные сети и потоки используются для поиска решения NP-трудной задачи целочисленного сбалансирования трех- и четырехмерной матрицы (см., например, [12, 13]), которая возникает при планировании железнодорожных грузоперевозок. Имеется матричный план по отправке вагонов, который группируется по некоторым показателям (например, направление, тип вагона, владелец вагона и т. п.). Данный план составляется на месяц и естественно является целочисленным. Однако вагоны необходимо отправлять ежедневно. При делении на количество дней в месяце план перестает быть целочисленным. Поэтому возникает проблема такого округления основных параметров, чтобы суммирующие показатели не выходили за определенные рамки. Данный план может быть представлен в виде s-мерной матрицы, где s — это число показателей, по которым ведется суммирование. Задача целочисленного сбалансирования может быть сведена к задаче о наибольшем кратном потоке (кратность сети равна 2 для трехмерной матрицы и 5 — для четырехмерной).

В разделе 1 мы дадим необходимые определения относительно кратных графов и деревьев.

В разделе 2 мы поставим задачи о минимальном остовном дереве и о минимальном полном остовном дереве. Затем мы сформулируем и обсудим точный алгоритм для второй задачи для случая делимого кратного графа.

1. Кратные графы и кратные деревья: определения

Напомним некоторые определения относительно кратных графов и деревьев, которые были ранее сформулированы в статьях [1—3].

Определение 1. *Кратный граф* G произвольной натуральной кратности $k > 1$ — это граф, вершины которого могут соединяться ребрами одного из 3 видов:

1. *Обычное ребро* e^o; множество обычных ребер обозначим через E^o.
2. *Кратное ребро* e^k между двумя вершинами, которое состоит из k одинаковых связанных ребер; связанные ребра кратного ребра могут использоваться только согласованно; множество кратных ребер обозначим через E^k.
3. *Связанное ребро* e между двумя вершинами, имеющее один общий конец с другим $(k-1)$ ребром (у любых двух из k связанных ребер только один конец является общим); множество связанных общей вершиной ребер будем называть *мультиребром* e^m; связанные ребра мультиребра могут использоваться только согласованно; множество мультиребер обозначим через E^m.

Если вершина инцидентна какому-либо кратному ребру, то она может быть инцидентна другим кратным ребрам, а также она может быть общим концом какого-либо мультиребра.

Если вершина является общим концом какого-либо мультиребра, то она не может быть общим концом никакого другого мультиребра.

Если вершина является отдельным концом мультиребра или инцидентна обычному ребру, то она не может быть общим концом мультиребра и не может быть инцидентна кратному ребру.

Множества вершин и ребер графа G обозначим через V и E соответственно ($E = E^o \cup E^k \cup E^m$). Обычное или кратное ребро, соединяющее две вершины x и y, обозначается стандартным образом: $\{x, y\}$. Мультиребро, соединяющее общую вершину x с k отдельными вершинами $y_1, \ldots, y_k$, обозначается так: $e^m_x = \{x, \{y_1, \ldots, y_k\}\}$. Далее мы будем рассматривать только неориентированные кратные графы.

Рис. 1 иллюстрирует определение 1. На рис. 1 (a) кратное ребро представлено в виде объединения k одинаковых ребер между двумя вершинами, что показано штрихами. Равенство (или согласованность) связанных ребер предполагает, что все характеристики этих ребер (например, длина) одинаковы, и эти ребра могут использоваться только одновременно. Так, если осуществляется проход в определенном направлении по одному из связанных ребер, то одновременно с этим оставшееся $(k-1)$ ребро проходится в том же самом направлении. Кратное ребро может включаться в какие-либо новые структуры только целиком. В дальнейшем мы будем обозначать кратные ребра жирными линиями, как на рис. 1 (b).

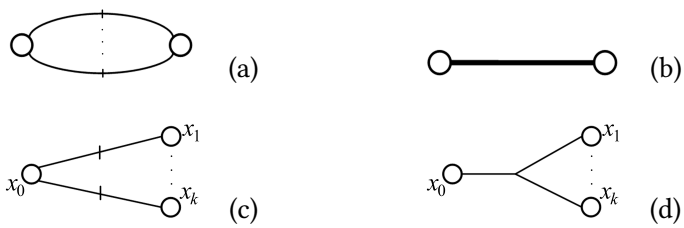

Fig. 1. Multiple and multi-edge

Рис. 1. Кратное и мультиребро

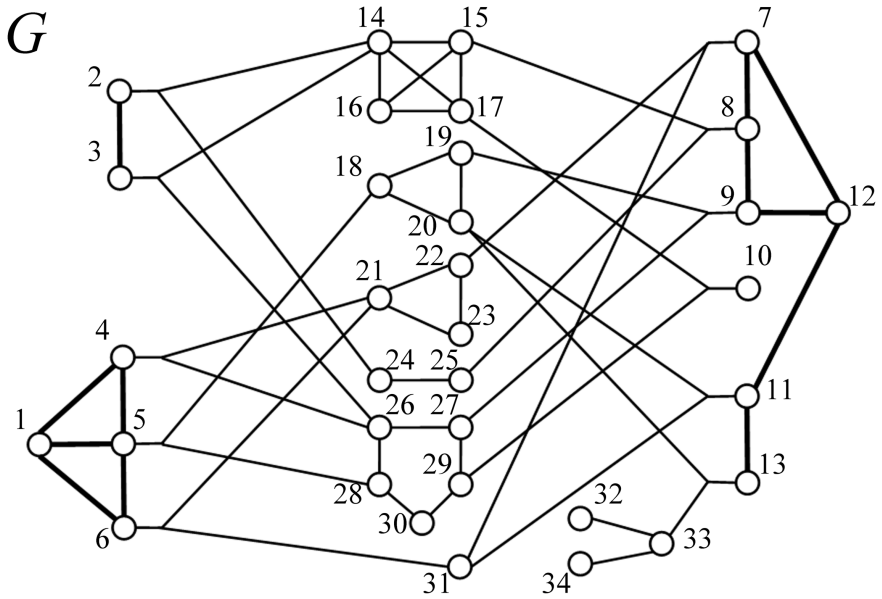

Fig. 2. Divisible graph of multiplicity 2

Рис. 2. Делимый граф кратности 2

На рис. 1 (c) мультиребро $\{x_0, \{x_1, \ldots, x_k\}\}$ представлено в виде объединения k одинаковых ребер, связывающих общую вершину x_0 с k попарно различными вершинами $x_1, \ldots, x_k$. Как и на рис. 1 (a), равенство ребер показано штрихами. Согласованность связанных ребер имеет тот же смысл, что и для кратных ребер. Если мы проходим связанное ребро $\{x_0, x_i\}$ от или к общему концу x_0, мы должны одновременно проходить все остальные связанные ребра в том же направлении. В дальнейшем мультиребра мы будем изображать при помощи расщепляющихся на k частей линий, как на рис. 1 (d).

Определение 2. *Обычной вершиной* назовем вершину, которая инцидентна обычному ребру или является отдельным концом мультиребра.

Кратной вершиной назовем вершину, которая инцидентна кратному ребру или является общим концом мультиребра.

Из определения 1 следует, что множества обычных и кратных вершин не пересекаются. При этом кратная вершина может быть соединена с обычными только посредством мультиребра.

Определение 3. *Делимым кратным графом* назовем такой кратный граф $G(V, E)$, в котором все связанные ребра кратных и мультиребер можно пронумеровать от 1 до k таким образом, что граф представляется в виде объединения $G = \cup_{i=1}^{k} G_i$. При этом каждый подграф G_i ($i \in \overline{1,k}$) содержит связанные ребра с номером i всех кратных и мультиребер, а также компоненты связности, состоящие из обычных ребер и инцидентные i-ым связанным ребрам всех мультиребер.

Подграфы G_i будем называть *частями* делимого графа.

При удалении всех мультиребер делимый граф распадется на $n \geqslant 1$ компонент связности (связность здесь понимается в том же смысле, что и для обычных графов), каждая из которых содержит только кратные ребра либо только обычные ребра. Очевидно, что каждая часть G_i является обычным графом. При этом возможность выделения частей G_i является особенностью делимых графов. В общем случае получить части G_i не удастся.

Пример 1. Рассмотрим делимый кратный граф G кратности 2, показанный на рис. 2. Для лучшей читаемости рисунка будем обозначать вершины их номерами без буквы «x».

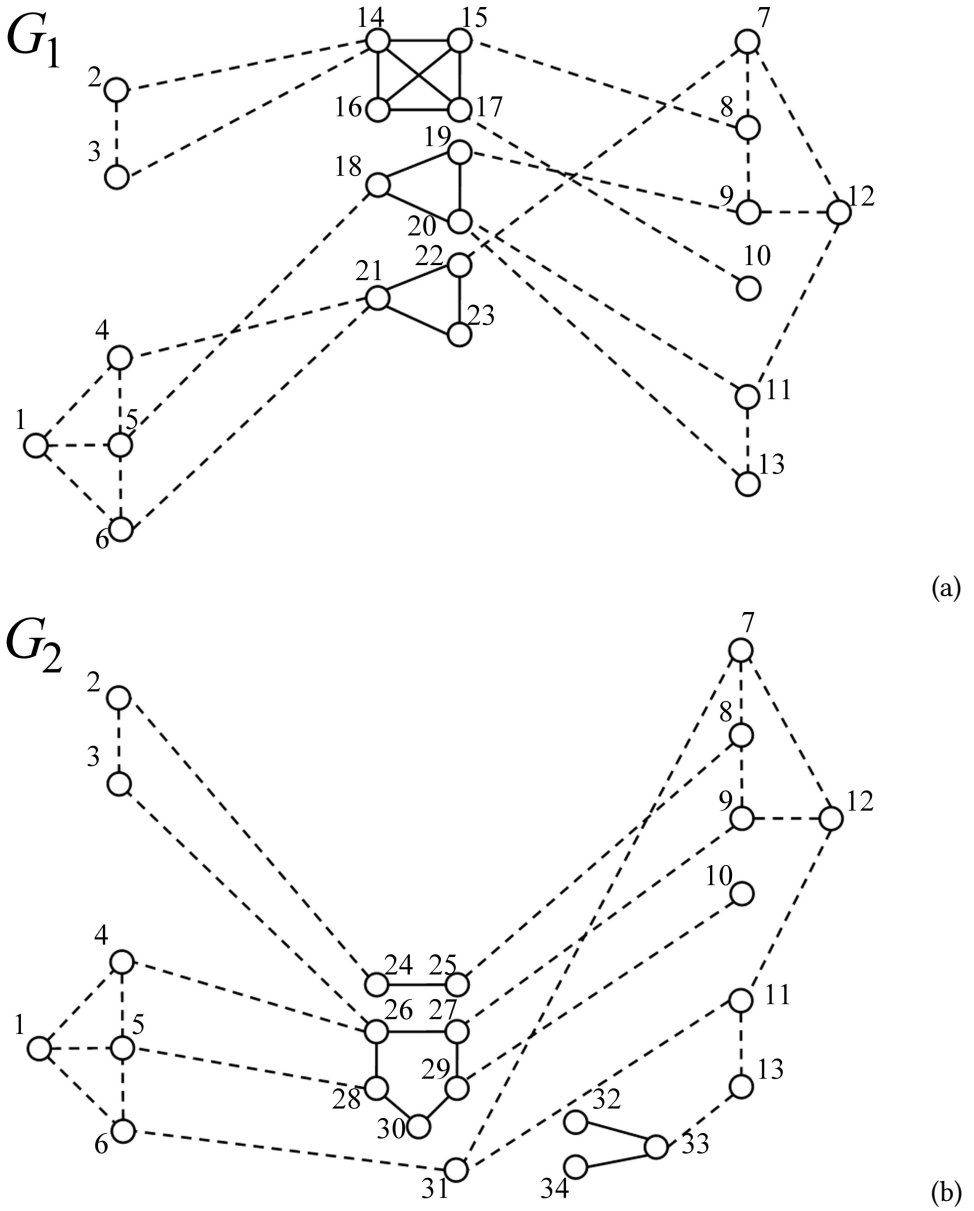

Fig. 3. Partition of a divisible graph

Рис. 3. Части делимого графа

На рис. 3 показаны части G_1 и G_2 делимого кратного графа G. Все связанные ребра обозначены пунктирными линиями.

Часть G_1 этого графа (рис. 3 (a)) состоит из всех обычных вершин с номерами от 14 до 23, всех соединяющих их обычных ребер, всех связанных ребер инцидентных им мультиребер, а также из всех кратных вершин и первых связанных ребер каждого мультиребра.

Часть G_2 (рис. 3 (b)) состоит из всех обычных вершин с номерами от 24 до 34, всех соединяющих их обычных ребер, всех связанных ребер инцидентных им мультиребер, а также из всех кратных вершин и вторых связанных ребер каждого мультиребра.

Дадим теперь определение кратного пути. Основное отличие кратного пути от пути в обычном графе состоит в том, что связанные ребра каждого кратного и мультиребра должны проходиться в этом пути согласованно.

Определение 4. $S(x,y) = \cup_{i=1}^{k} S^i(x,y)$ является *кратным путем из вершины x в вершину y* в графе $G(V,E)$, если выполнены следующие условия:

1. $S^i(x,y) = \left(\{x, v_1^i\}, \{v_1^i, v_2^i\}, \ldots, \{v_{l_i-1}^i, v_{l_i}^i\}, \{v_{l_i}^i, y\}\right)$, где $l_i \geqslant 0$, — последовательность ребер, представляющая собой обычный (некратный) путь из x в y, где каждое ребро $\{a,b\}$ является либо обычным ребром в графе $G(V,E)$, либо i-ым связанным ребром кратного или мультиребра. Значения l_i и l_j $(i \neq j)$ не согласовываются и могут быть как равными, так и различными. Если в путь $S(x,y)$ не входит ни одного кратного или мультиребра, то $S^2(x,y) = S^3(x,y) = \ldots = S^k(x,y) = \varnothing$.
2. Любая обычная вершина может встретиться в $S^i(x,y)$ несколько раз, то есть $S^i(x,y)$ может содержать циклы.
3. Никакая кратная вершина не может встретиться в $S^i(x,y)$ дважды.
4. Любое обычное ребро может встречаться в $S^i(x,y)$ несколько раз, причем направления, в которых оно проходится в разных вхождениях, могут не совпадать.
5. Обычное ребро, входящее в $S^i(x,y)$, может также входить в любой $S^j(x,y)$, $j \neq i$.
6. Все пути $S^i(x,y)$ согласованы (одинаковы) на общей части. Это условие означает, что если связанное ребро какого-то кратного или мультиребра входит в некоторый путь $S^i(x,y)$, то остальные связанные ребра должны входить во все $S^j(x,y)$, $j \neq i$ (по одному связанному ребру в каждый $S^j(x,y)$). При этом порядок вхождения всех кратных и мультиребер во все $S^i(x,y)$ одинаков.
   Фактически это значит, что если e_1 и e_2 — это два ребра пути $S(x,y)$, каждое из которых либо кратное, либо мультиребро, и в проекции $S^i(x,y)$ связанное ребро из e_1 проходится раньше связанного ребра из e_2, то во всех остальных проекциях $S^j(x,y)$ связанные ребра из e_2 могут проходиться только после связанных ребер из e_1.
7. Если $S(x,y)$ содержит мультиребро $\{x_0, \{x_1, \ldots, x_k\}\}$, проходимое в направлении от общего конца, то он не может содержать никакого другого мультиребра $\{y_0, \{x_1, \ldots, x_k\}\}$, проходимого в том же направлении. Аналогичное условие должно выполняться и в случае движения к общему концу.

Определение 5. Кратный путь $S(x,y)$ является *кратным циклом*, если $x = y$ и $S(x,y) \neq \varnothing$.

Пример 2. Рассмотрим граф G, показанный на рис. 2, и получим следующий кратный путь $S(x_8, x_{10})$:

$$S^1(x_8,x_{10}) = \big(\underline{\{x_8,x_{15}\}},\ \{x_{15},x_{16}\},\ \underline{\underline{\{x_{16},x_{14}\}}},\ \underline{\{x_{14},x_2\}},\ \underline{\{x_2,x_3\}},\underline{\{x_3,x_{14}\}},\ \underline{\underline{\{x_{14},x_{16}\}}},\ \{x_{16},x_{17}\},\ \underline{\{x_{17},x_{10}\}}\big),$$

$$S^2(x_8,x_{10}) = \big(\underline{\{x_8,x_{25}\}},\ \{x_{25},x_{24}\},\ \underline{\{x_{24},x_2\}},\ \underline{\{x_2,x_3\}},\ \underline{\{x_3,x_{26}\}},\{x_{26},x_{27}\},\ \{x_{27},x_{29}\},\ \underline{\{x_{29},x_{10}\}}\big).$$

На рис. 4 (a) показан этот кратный путь $S(x_8,x_{10})$. На рис. 4 (b) и (c) показаны части кратного пути $S^1(x_8,x_{10})$ и $S^2(x_8,x_{10})$ соответственно.

Все ребра общей части S^1 и S^2 согласованы (эти ребра отмечены подчеркиванием). Пример демонстрирует особенности кратного пути. Действительно, у нас есть обычное ребро $\{x_{14},x_{16}\}$ (отмечено двойным подчеркиванием), которое входит в путь S^1 дважды и проходится в противоположных направлениях. Более того, в S^1 есть обычный цикл, однако кратного цикла в пути S нет.

Определение 6. *Множеством достижимости по кратным ребрам* для некоторой кратной вершины x назовем множество R_x^k всех вершин y таких, что существует путь из x в y, проходящий только по кратным ребрам.

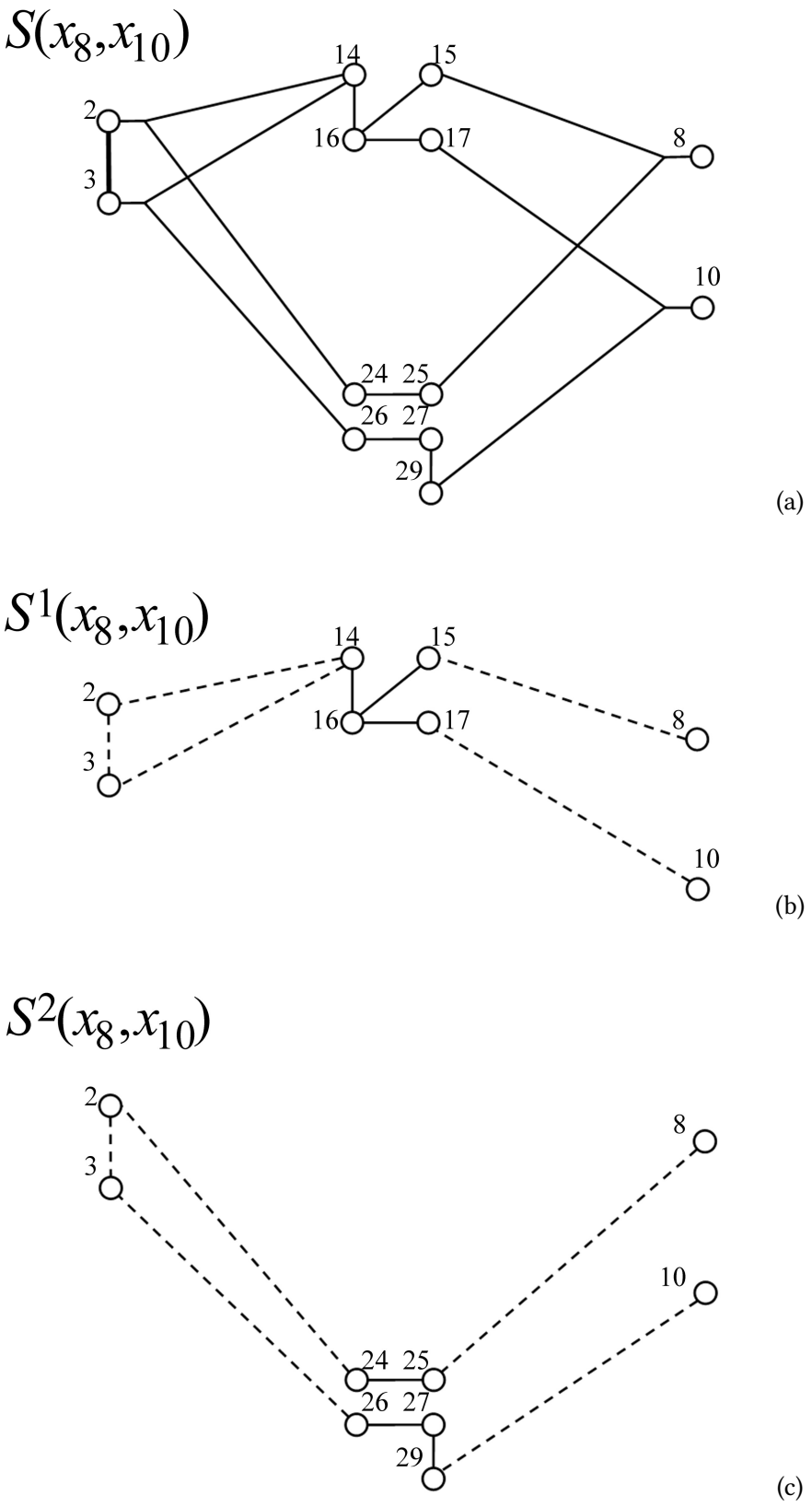

Fig. 4. Multiple path and its parts

Рис. 4. Кратный путь и его части

Определение 7. *Множеством достижимости по обычным ребрам* для некоторой обычной вершины x назовем множество R_x^o всех вершин y таких, что существует путь из x в y, проходящий только по обычным ребрам.

Определение 8. Множества достижимости R_x^k и R_y^k являются *смежными*, если для произвольных вершин $a \in R_x^k$, $b \in R_y^k$ существует соединяющий их кратный путь $S(a, b)$.

Определение 9. Кратный граф $G(V, E)$ является *связным*, если одновременно выполнены два условия:

1. Кратный путь $S(x, y)$ существует для любых двух кратных вершин $x \in V$, $y \in V$.
2. Невозможно выделить такой подграф $G' \subset G$, который будет содержать только обычные ребра, и при этом подграфы G' и $G \setminus G'$ не будут соединены ни одним ребром (обычным ребром или связанным ребром мультиребра).

В отличие от обычных графов, связность кратного графа не предполагает наличие кратных путей из каждой вершины в каждую. Фактически в связном кратном графе между каждой парой вершин должен существовать обычный (некратный) путь, использующий связанные ребра кратных и мультиребер несогласованно, а кратные пути обязательно должны существовать только для пар кратных вершин.

Определение 10. Делимый кратный граф $G(V, E)$ является *связным*, если одновременно выполнены два условия:

1. Кратный путь $S(x, y)$ существует для любых двух кратных вершин $x \in V$, $y \in V$.
2. Каждая из частей G_i является связным (некратным) графом.

Определение 11. *Кратное дерево* — это связный кратный граф без циклов.

В статьях [2, 3] мы показали, что два кратных дерева с одинаковым количеством кратных и обычных вершин могут содержать разное количество ребер.

Определение 12. *Остовным деревом* в кратном графе $G(V, E)$ называется кратное дерево $T(V, E')$, для которого $E' \subseteq E$.

Заметим, что в остовном дереве заведомо будут существовать кратные пути $S(x, y)$ для всех кратных вершин x, y. Однако, если хотя бы одна из вершин x, y является обычной, существование пути $S(x, y)$ не гарантировано даже в том случае, когда такой путь существует в исходном графе $G(V, E)$.

Определение 13. Остовное дерево $T(V, E')$ в кратном графе $G(V, E)$ является *полным*, если для любой пары вершин $x \in V$, $y \in V$ кратный путь $S_T(x, y)$ в дереве $T(V, E')$ существует тогда и только тогда, когда существует кратный путь $S_G(x, y)$ в исходном графе $G(V, E)$.

2. Задача о минимальном полном остовном дереве в делимом кратном графе и алгоритм ее решения

2.1. Постановка задачи

Напомним определение длины ребра из [2, 3].

Определение 14. Целочисленная функция $l(e)$, определенная для всех ребер $e \in E$, является *длиной (весом) ребра* в кратном графе $G(V, E)$, если выполнено следующее:

1. $l(e) > 0$ для любого ребра e.

2. Если e является кратным или мультиребром, то $l(e_1) = l(e_2) = \ldots = l(e_k)$ и $l(e) = k \cdot l(e_1)$, где $e_1, \ldots, e_k$ — это связанные ребра данного ребра e.

Тогда *вес кратного графа* $G(V, E)$ определяется по формуле

$$w(G) = \sum_{e \in E} l(e).$$

Поставим две задачи о минимальном остовном дереве.

Задача 1 (минимальное остовное дерево)**.** *В кратном графе* $G(V, E)$ *требуется найти такое остовное дерево* $T^{\min}(V, E')$*, что для любого другого остовного дерева* $T(V, E'')$ *выполнено*

$$w(T^{\min}(V, E')) \leqslant w(T(V, E'')).$$

Задача 2 (минимальное полное остовное дерево)**.** *В кратном графе* $G(V, E)$ *требуется найти такое полное остовное дерево* $T^{\min}_{complete}(V, E')$*, что для любого другого полного остовного дерева* $T_{complete}(V, E'')$ *выполнено*

$$w(T^{\min}_{complete}(V, E')) \leqslant w(T_{complete}(V, E'')).$$

Задача 2 является более интересной, поскольку кратные пути существуют в полном остовном дереве для всех пар вершин, для которых соответствующие пути существуют в исходном графе. Кроме того, очевидно, что

$$w(T^{\min}) \leqslant w(T^{\min}_{complete}).$$

Существование полного остовного дерево в связном кратном графе доказывается в статьях [2] (для делимого графа) и [3] (для произвольного кратного графа). В статье [4] мы доказали, что задачи 1 и 2 являются NP-трудными даже в случае делимого графа, если кратность $k \geqslant 3$. Предположительно, задачи являются NP-трудными и для кратности $k = 2$.

2.2. Точный алгоритм для задачи 2

В статьях [2, 3] мы получили эвристические алгоритмы для задачи 2 для случаев делимого и произвольного кратного графа. Построим теперь точный алгоритм для задачи 2 для случая делимого графа. Будем формулировать алгоритм для графов кратности 2 (далее будет показано, что аналогичный алгоритм может быть получен и для любой другой кратности $k > 2$).

Пусть имеется взвешенный делимый граф $G(V, E)$ кратности 2.

Алгоритм 1 (редукция графа)**.**

1. Находим все множества достижимости по обычным ребрам R^o_x (используется полиномиальный алгоритм 2 из статьи [1]).
2. С помощью алгоритма Краскала (см. [14]) находим минимальное остовное дерево T^o_x в каждом подграфе G^o_x, образованном множеством вершин R^o_x и всеми обычными ребрами, которые соединяют вершины из R^o_x в графе $G(V, E)$. Множеством ребер дерева T^o_x будет $E^o_x \subseteq E^o$. Получение таких деревьев является необходимым и достаточным условием полноты кратного остовного дерева в делимом графе (теорема 3 из статьи [2]).
3. Обозначим каждое найденное дерево через T^p_r, где r — это номер дерева, а p — это номер части G_p исходного графа G.
4. Получим новый делимый граф $G_{red}(V_{red}, E_{red})$ следующим образом:
   (a) Поместим все кратные вершины из множества V в множество V_{red}. Поместим обычные вершины T^p_r в множество V_{red}. Каждая из них соответствует обычному дереву с предыдущего шага.

(b) Установим $E^k_{red} = E^k$ и $E^o_{red} = \varnothing$.

(c) Получим мультиребра $\{x_i, \{T^1_r, T^2_s\}\}$ на основе всех мультиребер $\{x_i, \{x_t, x_u\}\} \in E^m$, заменяя обычные вершины x_t и x_u на обычные вершины T^1_r и T^2_s, которые соответствуют обычным деревьям, содержащим x_t и x_u. Добавим все такие мультиребра $\{x_i, \{T^1_r, T^2_s\}\}$ в множество E^m_{red}.

5. Граф G_{red} является делимым и не содержит обычных ребер. Редуцируем его до обычного графа $G_{ord}(V_{ord}, E_{ord})$. Некоторые вершины нового графа будут соответствовать сразу паре вершин графа G_{red}. Такие вершины будем называть *квазивершинами* здесь и далее.

   (a) Создадим *квазивершины* $q_{rs} = \{T^1_r, T^2_s\}$ для каждого мультиребра $\{x_i, \{T^1_r, T^2_s\}\} \in E^m_{red}$ и включим эти квазивершины в множество V_{ord}.

   (b) Преобразуем каждую кратную вершину $x_i \in V_{red}$ в обычную вершину $x_i \in V_{ord}$.

   (c) Преобразуем каждое кратное ребро $\{x_i, x_j\} \in E^k_{red}$ в обычное ребро $\{x_i, x_j\} \in E_{ord}$ той же длины.

   (d) Преобразуем каждое мультиребро $\{x_i, \{T^1_r, T^2_s\}\} \in E^m_{red}$ в обычное ребро $\{x_i, q_{rs}\} \in E_{ord}$ той же длины.

Заметим, что каждое минимальное полное остовное дерево графа G_{red} соответствует минимальному полному остовному дереву графа G. Последнее может быть получено заменой всех вершин T^p_r в минимальном полном остовном дереве графа G_{red} на обычные минимальные остовные деревья, найденные на шаге 2 алгоритма 1. Более того, каждое остовное дерево графа G_{red} является полным, поскольку в графе G_{red} нет обычных ребер.

Граф G_{ord} строится таким образом, что каждое остовное дерево в нем соответствует полному остовному дереву в графе G_{red}. Однако минимальное остовное дерево в графе G_{ord} может не соответствовать минимальному полному остовному дереву в графе G_{red}. Причина в том, что каждая квазивершина представляет собой пару обычных деревьев. Эти пары могут пересекаться, поэтому какие-то квазивершины будут «избыточными» для минимального остовного дерева в графе G_{red}. Тем не менее, эти «избыточные» вершины должны присутствовать в остовном дереве в графе G_{ord}.

Вышесказанное приводит к идее исключения каких-то квазивершин при поиске минимального остовного дерева в графе G_{ord}. Мы можем перебрать все возможные варианты такого исключения, получить минимальные остовные деревья в обычном графе для тех вариантов, в которых графы G_{ord} и G_{red} остаются связными, и выбрать кратчайшее из деревьев, которое и будет соответствовать минимальному полному остовному дереву в графе G_{red}.

Но этот подход требует перебора 2^q вариантов, где q — это количество квазивершин. Заметим, что мы можем уменьшить глубину перебора, если предварительно определим квазивершины, которые являются или не являются обязательными для графа производного от графа G_{ord}.

Разобьем множество квазивершин на четыре непересекающихся подмножества:

- Q_A — это множество квазивершин, которые обеспечивают смежность каких-то множеств достижимости по кратным ребрам R^k_x и R^k_y в графе G_{red}. Это значит, что в графе G_{red} невозможно получить остовное дерево, не включив в него некоторое мультиребро, соединяющее вершину из множества R^k_x с вершинами из q_{rs}, и мультиребро, соединяющее вершину из множества R^k_y с вершинами из q_{rs}, где $q_{rs} \in Q_A$;
- Q_B — это множество квазивершин, которые *могут* обеспечивать смежность каких-то множеств достижимости по кратным ребрам R^k_x и R^k_y в графе G_{red}. Однако можно обеспечить указанную смежность и альтернативным способом, если определенная вершина из множества Q_B будет исключена из графа G_{ord};
- Q_C — это множество квазивершин, которые являются листьями в графе G_{ord}; при этом хотя бы одна из компонент T^1_r, T^2_s квазивершины $q_{rs} \in Q_C$ не является компонентой никакой квазивершины из $Q_A \cup Q_B$;

- Q_D — это множество квазивершин, которые являются листьями в графе G_{ord}; при этом обе компоненты T_r^1, T_s^2 квазивершины $q_{rs} \in Q_D$ являются компонентами каких-то квазивершин из $Q_A \cup Q_B$.

Очевидно, $Q_A \cup Q_B$ содержит все квазивершины из V_{ord}, не являющиеся листьями.

Мы можем найти все множества Q_A, Q_B, Q_C и Q_D с помощью следующего полиномиального алгоритма. Также мы можем профильтровать Q_D, исключив квазивершины, которые гарантированно «избыточны». В то же время, множество Q_A состоит из квазивершин, которые не могут быть исключены, поскольку минимальное остовное дерево в графе G_{red} должно содержать все пары их компонент (как отдельные концы мультиребер).

Алгоритм 2 (классификация и фильтрация квазивершин графа G_{ord})**.**

1. Последовательно просмотрим все квазивершины q_{rs}, для которых $\deg q_{rs} > 1$. Если $\deg q_{rs} = n$, у нас есть n ребер $\{x_{rs}^i, q_{rs}\} \in E_{ord}$ ($i \in \overline{1,n}$). Временно исключим все эти ребра из E_{ord} и проверим существование n путей $S(x_{rs}^1, x_{rs}^2)$, …, $S(x_{rs}^{n-1}, x_{rs}^n)$ и $S(x_{rs}^n, x_{rs}^1)$ (фактически мы проверяем связность графа G_{ord} после исключения вершины q_{rs}). Если хотя бы один из путей не существует, включаем q_{rs} в Q_A. Иначе включаем q_{rs} в Q_B.
2. Последовательно просмотрим все квазивершины q_{rs}, для которых $\deg q_{rs} = 1$. Возьмем их компоненты T_r^1 и T_s^2 и проверим, являются ли они компонентами каких-то квазивершин из $Q_A \cup Q_B$. Если для обеих компонент ответ «да», включим q_{rs} в Q_D. Иначе включим q_{rs} в Q_C.
3. Профильтруем Q_D. Последовательно просмотрим все квазивершины $q_{rs} \in Q_D$. Возьмем их компоненты T_r^1 и T_s^2 и проверим, являются ли они обе компонентами каких-то квазивершин из Q_A. Если для обеих компонент ответ «да», исключим q_{rs} из Q_D и из V_{ord}. Также исключим единственное инцидентное этой квазивершине ребро $\{x_i, q_{rs}\}$ из E_{ord}.

Далее будем рассматривать граф G_{ord} в виде, полученном на шаге 3 алгоритма 2.

Определение 15. Обычный граф G_{der} с квазивершинами называется *производным от графа* G_{ord}, если он получен из графа G_{ord} в результате исключения каких-то квазивершин из V_{ord} и исключения всех инцидентных этим квазивершинам ребер из E_{ord}. Граф G_{der} должен также соответствовать следующим условиям:

1. G_{der} связен.
2. Каждое остовное дерево в графе G_{der} соответствует полному остовному дереву в графе G_{red}.

Заметим, что только квазивершины из Q_B, Q_C или Q_D могут исключаться из V_{ord} при получении производного графа, поскольку все квазивершины из Q_A обязательны для связности графа. Мы можем получить все возможные производные графы G_{der} и найти минимальные остовные деревья в них. Очевидно, кратчайшее из этих деревьев порождает минимальное полное остовное дерево в делимом графе G_{red}.

Обозначим через DER множество всех графов производных от G_{ord}.

Алгоритм 3 (генерация всех графов производных от G_{ord})**.**

1. Определим *базовую часть* G_{base} с множествами вершин и ребер V_{base} и E_{base}. Она содержит вершины и ребра, которые являются обязательными для каждого производного графа.
   Поэтому базовая часть должна содержать все обычные вершины из V_{ord}, все квазивершины из Q_A и все ребра из E_{ord}, связывающие указанные вершины.
2. Рассмотрим множество $Q_B = \{q_{r_1s_1}, \ldots, q_{r_ps_p}\}$. Найдем все его подмножества:

$$Q_B^1 = \varnothing, \quad Q_B^2 = \{q_{r_1s_1}\}, \quad \ldots, \quad Q_B^{p+1} = \{q_{r_ps_p}\},$$

$$Q_B^{p+2} = \{q_{r_1s_1}, q_{r_2s_2}\}, \quad \ldots, \quad Q_B^{2^p} = Q_B.$$

Обозначим через E_B^j множество ребер инцидентных вершинам из Q_B^j в графе G_{ord} ($j \in \overline{1, 2^p}$). Положим $DER = \varnothing$, $i = 1$.

3. Сгенерируем граф G_{der}^i производный от G_{ord}. Его множества вершин и ребер обозначим через V_{der}^i и E_{der}^i. Сначала положим $V_{der}^i = V_{base} \cup Q_B^i$ и $E_{der}^i = E_{base} \cup E_B^i$. Если граф не связен, производный граф G_{der}^i не существует, переходим на шаг 7. Иначе переходим на шаг 4.
4. Сформируем множество Q_D^i. Для этого произведем дополнительную фильтрацию множества Q_D. Последовательно просмотрим все квазивершины $q_{rs} \in Q_D$. Возьмем их компоненты T_r^1 и T_s^2 и проверим, являются ли они компонентами каких-то квазивершин из $Q_A \cup Q_B^i$. Если хотя бы для одной компоненты ответ «нет», включим q_{rs} в Q_D^i.
5. Рассмотрим множество $Q_C \cup Q_D^i$. Каждая квазивершина q_{rs} из этого множества содержит хотя бы одну компоненту T_r^1 или T_s^2, которая не является компонентой никакой квазивершины из множества $Q_A \cup Q_B^i$. Все такие компоненты образуют два множества:

$$T^1 = \{T_{r_1}^1, \ldots, T_{r_l}^1\}, \qquad T^2 = \{T_{s_1}^2, \ldots, T_{s_n}^2\}.$$

Для каждой квазивершины $q_{rs} \in Q_C \cup Q_D^i$ определим ее вес $w(q_{rs}) = l(\{x_t, q_{rs}\})$, где $\{x_t, q_{rs}\}$ — это единственное ребро, инцидентное q_{rs}.
Затем выделим подмножество Q_{CD}^i минимального суммарного веса $\sum\limits_{q_{rs} \in Q_{CD}^i} w(q_{rs})$ среди всех подмножеств множества $Q_C \cup Q_D^i$ такое, что оно покрывает T^1 и T^2, то есть каждый элемент T^1 и T^2 является компонентой хотя бы одной квазивершины из подмножества. Очевидно, что

$$\max\{l, n\} \leqslant |Q_{CD}^i| \leqslant l + n.$$

Определим $E_{CD}^i \subseteq E_{ord}$ как множество ребер инцидентных квазивершинам из множества Q_{CD}^i.
6. Граф G_{der}^i определяется множествами вершин и ребер:

$$V_{der}^i = V_{base} \cup Q_B^i \cup Q_{CD}^i, \qquad E_{der}^i = E_{base} \cup E_B^i \cup E_{CD}^i.$$

Добавим этот граф G_{der}^i в множество DER.
7. Положим $i = i + 1$. Если $i > 2^p$, выход. Иначе переходим на шаг 3.

Заметим, что алгоритм 3 выполняется за экспоненциальное число шагов в общем случае. Действительно, мы ищем $2^{|Q_B|}$ подмножеств на шаге 2 и выполняем шаги 3–7 для каждого подмножества. Более того, на шаге 5 требуется выбрать одно из $\left(C_{|Q_C \cup Q_D^i|}^{\max\{l,n\}} + \ldots + C_{|Q_C \cup Q_D^i|}^{l+n}\right)$ подмножеств в худшем случае, что также требует экспоненциального числа операций.

Но задача 2 NP-трудная, поэтому любой точный алгоритм для нее потребует экспоненциального числа шагов, если $P \neq NP$. Тем не менее, алгоритм 3 будет выполняться за приемлемое время, если $|Q_B|$, $|Q_C|$ и $|Q_D|$ невелики в то время, как $|Q_A|$ может быть большим.

Будем использовать алгоритмы 1–3 на шагах точного алгоритма для задачи о минимальном полном остовном дереве в делимом кратном графе. Пусть $G(V, E)$ — взвешенный связный делимый кратный граф. Будем искать его минимальное полное остовное дерево $T_{complete}^{\min}(V, E^T)$, где $E^T \subseteq E$.

Алгоритм 4 (точный алгоритм для задачи 2 для делимого графа)**.**

1. С помощью алгоритма 1 находим все обычные минимальные остовные деревья T_r^p на множествах достижимости по обычным ребрам R_x^o, редуцированный кратный граф G_{red} и обычный граф G_{ord} с квазивершинами.
2. Классифицируем и фильтруем квазивершины графа G_{ord} с помощью алгоритма 2.
3. Рассмотрим граф G_{ord}, полученный после фильтрации на шаге 2. С помощью алгоритма 3 находим множество DER всех графов G_{der}^i производных от G_{ord}.

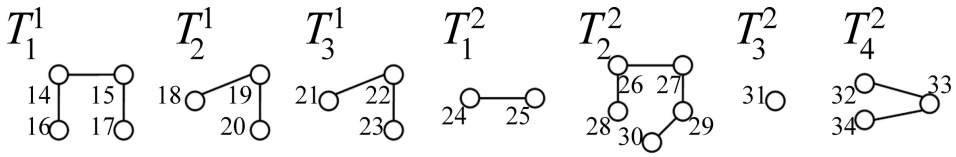

Fig. 5. The minimum spanning trees for the ordinary edges reachability sets

Рис. 5. Минимальные остовные деревья на множествах достижимости по обычным ребрам

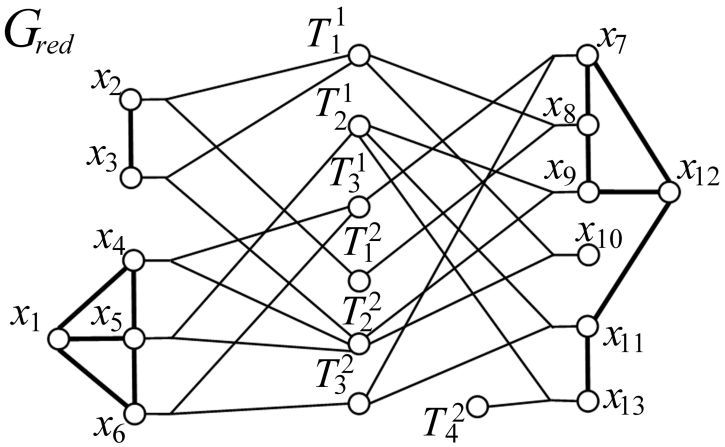

Fig. 6. Reduced divisible graph G_{red}

Рис. 6. Редуцированный кратный граф G_{red}

4. В каждом графе $G^i_{der} \in DER$ найдем минимальное остовное дерево T^i_{der} с помощью алгоритма Краскала. Выберем кратчайшее из них и обозначим его через $T^{\min}_{der}(V^{\min}_{der}, E^{\min}_{der})$.
5. Преобразуем дерево $T^{\min}_{der}$ в минимальное полное остовное дерево $T^{\min}_{red}(V_{red}, E^T_{red})$ графа G_{red}, где $E^T_{red} \subseteq E_{red}$. Получим E^T_{red} следующим образом.
   (a) Если для каких-то обычных вершин $x_t \in V^{\min}_{der}$, $x_u \in V^{\min}_{der}$ ребро $\{x_t, x_u\} \in E^{\min}_{der}$, включаем кратное ребро $\{x_t, x_u\} \in E_{red}$ в множество E^T_{red}.
   (b) Если для какой-то обычной вершины $x_t \in V^{\min}_{der}$ и для какой-то квазивершины $q_{rs} \in V^{\min}_{der}$ ребро $\{x_t, q_{rs}\} \in E^{\min}_{der}$, включаем мультиребро $\{x_t, \{T^1_r, T^2_s\}\} \in E_{red}$ в множество E^T_{red}.
6. Преобразуем дерево $T^{\min}_{red}$ в минимальное полное остовное дерево $T^{\min}_{complete}$ графа G. Для этого заменим все вершины $T^p_r \in V_{red}$ на соответствующие обычные деревья T^p_r, полученные на шаге 1.

2.3. Пример работы алгоритма

Пример 3. Рассмотрим делимый граф G кратности 2, показанный на рис. 2. Установим длины всех кратных ребер равными 2, длины всех обычных ребер равными 1, $l(\{x_4, \{x_{21}, x_{26}\}\}) = 2$, $l(\{x_6, \{x_{21}, x_{31}\}\}) = 2$, $l(\{x_7, \{x_{22}, x_{31}\}\}) = 6$, $l(\{x_{11}, \{x_{20}, x_{31}\}\}) = 2$, а длины всех остальных мультиребер равными 4.

Применим алгоритм 4 к графу G. Сначала найдем минимальные остовные деревья на множествах достижимости по обычным ребрам (рис. 5). Затем заменим их на соответствующие вершины T^p_r и получим редуцированный кратный граф G_{red} (рис. 6).

Далее сформируем квазивершины, чтобы получить граф G_{ord} (рис. 7 (а), квазивершины закрашены черным). Классификация квазивершин дает следующие множества:

$$Q_A = \{q_{11}, q_{12}\}, \quad Q_B = \{q_{22}, q_{33}\}, \quad Q_C = \{q_{24}\}, \quad Q_D = \{q_{23}, q_{32}\}.$$

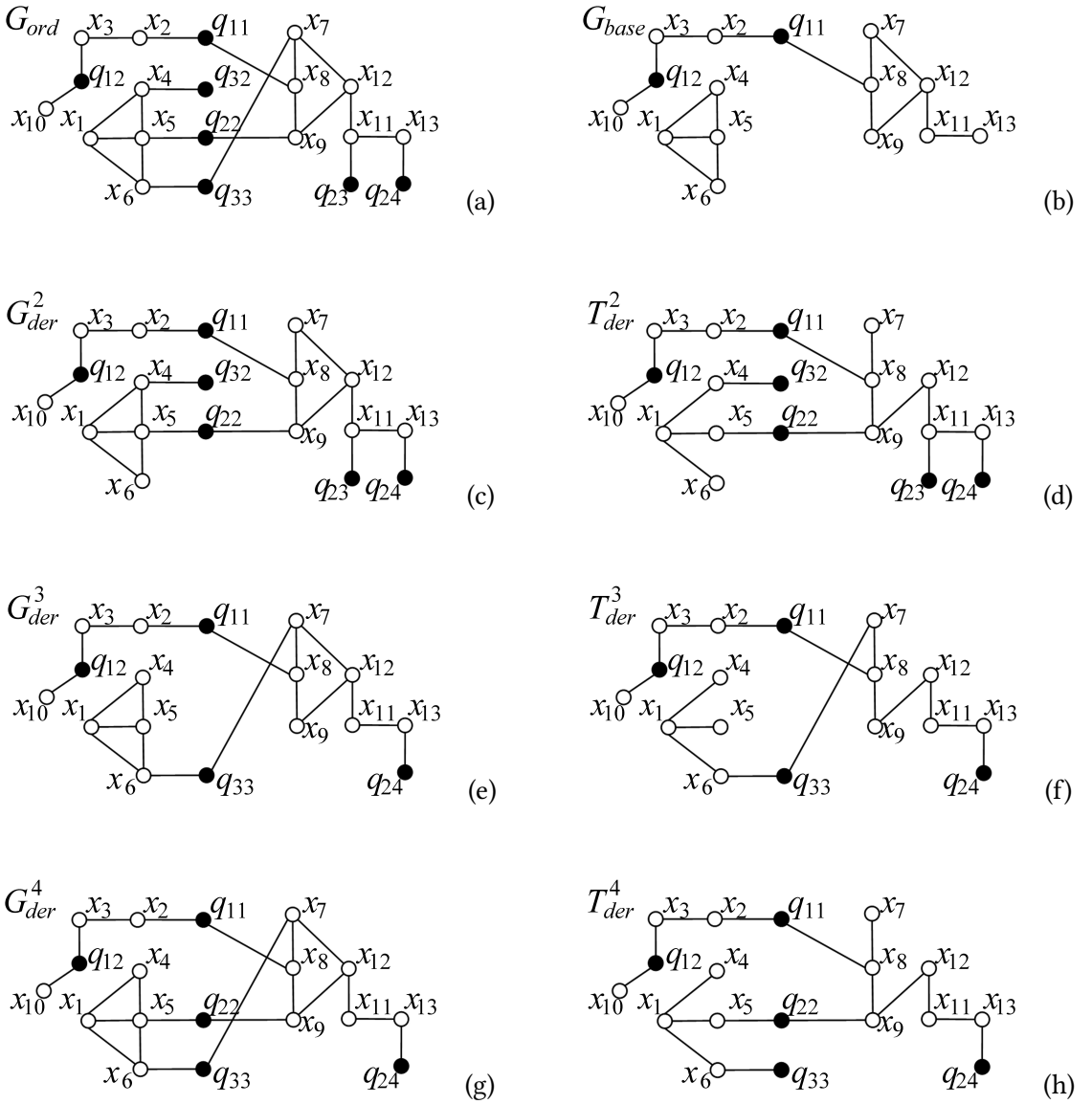

Fig. 7. Ordinary graph G_{ord} with quasi-vertices, its derivative graphs and theirs minimum spanning trees

Рис. 7. Обычный граф G_{ord} с квазивершинами, его производные графы и их минимальные остовные деревья

Базовая часть G_{base} графа G_{ord} состоит из всех обычных вершин, всех квазивершин из множества Q_A и из всех ребер, соединяющих эти вершины, см. рис. 7 (b). Сформируем подмножества Q^i_B ($i \in \overline{1,4}$):

$$Q^1_B = \varnothing, \quad Q^2_B = \{q_{22}\}, \quad Q^3_B = \{q_{33}\}, \quad Q^4_B = \{q_{22}, q_{33}\},$$

и попробуем получить производные графы G^i_{der} ($i \in \overline{1,4}$). Граф G^1_{der} не существует, потому что добавление Q^1_B в G_{base} приводит к несвязному графу. Остальные производные графы определяются следующими подмножествами:

$$Q^2_D = \{q_{23}, q_{32}\}, \quad Q^2_{CD} = \{q_{23}, q_{24}, q_{32}\};$$

$$Q^3_D = \{q_{23}\}, \quad Q^3_{CD} = \{q_{24}\};$$

$$Q^4_D = \varnothing, \quad Q^4_{CD} = \{q_{24}\}.$$

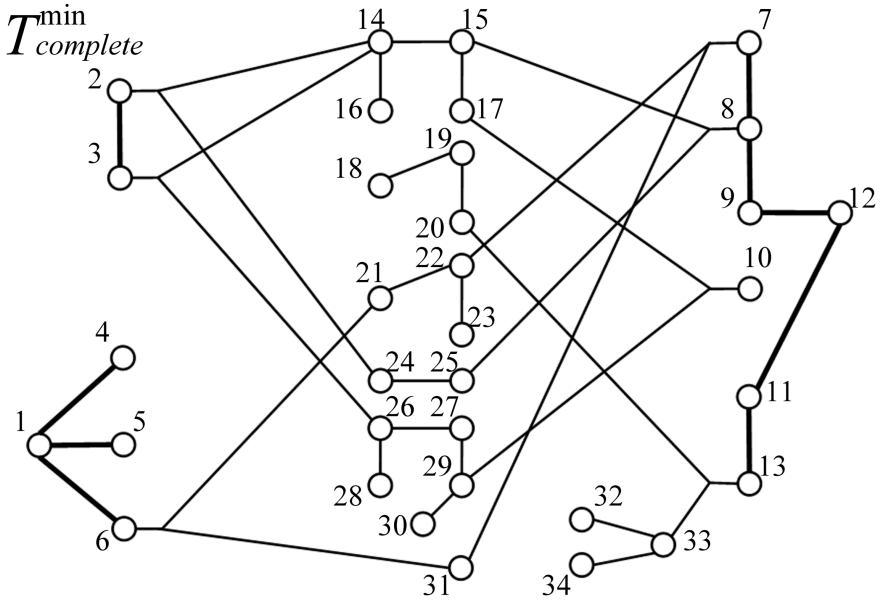

Fig. 8. The minimum complete spanning tree of the divisible graph G

Рис. 8. Минимальное полное остовное дерево делимого графа G

Производные графы показаны на рис. 7 (c), (e), (g). Чтобы получить минимальное остовное дерево в каждом графе, мы должны исключить ребра $\{x_4, x_5\}$, $\{x_5, x_6\}$ и $\{x_7, x_{12}\}$. Также мы должны исключить ребро $\{x_7, q_{33}\}$ из G^4_{der}. В итоге мы получим минимальные остовные деревья T^2_{der}, T^3_{der} и T^4_{der} этих графов; они показаны на рис. 7 (d), (f), (h). Их веса:

$$w(T^2_{der}) = 50, \quad w(T^3_{der}) = 46, \quad w(T^4_{der}) = 48.$$

Кратчайшее из них — T^3_{der}, оно порождает минимальное полное остовное дерево $T^{min}_{complete}$ исходного графа G веса $w(T^{min}_{complete}) = 60$ (рис. 8).

2.4. Анализ алгоритмов

Алгоритмы 1–4 сформулированы для делимых графов кратности 2. Однако они могут быть легко переформулированы для любой другой кратности $k \geqslant 3$. Действительно, отличия будут только в количестве частей G_p делимого графа ($p \in \overline{1,k}$), в диапазоне значений индекса p для обычных деревьев T^p_r, а также в количестве компонент квазивершины. В общем случае у каждой квазивершины будет ровно k компонент, поэтому нужно рассматривать квазивершины вида $q_{r_1 \ldots r_k} = \{T^1_{r_1}, \ldots, T^k_{r_k}\}$. Все остальное остается прежним.

Пусть теперь G — делимый граф произвольной кратности $k \geqslant 2$. Применим алгоритм 1 к графу G и получим обычный граф G_{ord} с квазивершинами.

Определение 16. Делимый кратный граф G назовем *графом простой структуры*, если в соответствующем обычном графе G_{ord} с квазивершинами $|Q_B| = |Q_C| = 0$.

Теорема 1. *Задача о минимальном полном остовном дереве в делимом кратном графе простой структуры может быть решена за полиномиальное время.*

Доказательство. Во-первых, из условия $|Q_B| = 0$ следует, что все компоненты $T^1_{r_1}, \ldots, T^k_{r_k}$ каждой квазивершины $q_{r_1 \ldots r_k} \in Q_D$ являются компонентами каких-то квазивершин из Q_A. Поэтому все квазивершины будут исключены из $|Q_D|$ на шаге 3 алгоритма 2. Значит, мы можем пропустить

шаги 4–5 алгоритма 3, поскольку $Q_D = \varnothing$ после фильтрации и $Q_C = \varnothing$ в силу простоты структуры графа.

Во-вторых, из условия $|Q_B| = 0$ следует, что только одно подмножество $Q_B^1 = \varnothing$ будет найдено на шаге 2 алгоритма 3. Поэтому шаги 3–7 этого алгоритма будут выполнены ровно один раз и ровно один граф G_{der}^1 будет производным от графа G_{ord}. Очевидно, что $V_{der}^1 = V_{ord}$ и $E_{der}^1 = E_{ord}$.

Следовательно, алгоритм 3 выполняется за полиномиальное время для любого делимого графа простой структуры. Шаг 4 алгоритма 4 выполняется ровно один раз, поскольку у нас есть только один граф производный от графа G_{ord}.

В итоге, все шаги алгоритма 4 выполняются однократно и каждый шаг полиномиален для данного класса графов. Таким образом, задача 2 полиномиальна в случае делимого кратного графа простой структуры.

□

Если мы будем применять алгоритм 4 на практике, мы можем столкнуться с проблемой нехватки памяти из-за того, что в алгоритме мы получаем экспоненциальное количество производных графов. Чтобы преодолеть указанную проблему, мы можем объединить шаги 3 и 4 алгоритма 4 следующим образом.

1. Зададим переменную $w_{cur} = \infty$ для хранения текущего веса минимального остовного дерева в производном графе. Обозначим текущее минимальное остовное дерево веса w_{cur} через T_{cur}. В начальный момент $T_{cur} = \varnothing$. Таким образом, мы будем хранить дерево T_{cur} и переменную w_{cur} вместо множества *DER*.
2. После шага 6 алгоритма 3 (поиск очередного графа G_{der}^i) переходим на шаг 4 алгоритма 4 для поиска минимального остовного дерева T_{der}^i в графе G_{der}^i. Если $w(T_{der}^i) < w_{cur}$, полагаем $T_{cur} = T_{der}^i$ и $w_{cur} = w(T_{der}^i)$. Переходим на шаг 7 алгоритма 3.
3. После последней итерации указанной процедуры полагаем $T_{der}^{\min} = T_{cur}$.

Также мы можем оптимизировать алгоритм для случая графа с большим количеством вершин и ребер. Мы можем сравнивать текущий вес $w(T_{der}^i)$ с весом w_{cur} на шаге 4 алгоритма 4 после добавления каждого нового ребра алгоритмом Краскала. Если $w(T_{der}^i) \geqslant w_{cur}$, можно прервать построение дерева и перейти к следующей итерации.

References

[1] A. V. Smirnov, “The shortest path problem for a multiple graph”, *Automatic Control and Computer Sciences*, vol. 52, no. 7, pp. 625–633, 2018. DOI: 10.3103/S0146411618070234.

[2] A. V. Smirnov, “The spanning tree of a divisible multiple graph”, *Automatic Control and Computer Sciences*, vol. 52, no. 7, pp. 871–879, 2018. DOI: 10.3103/S0146411618070325.

[3] A. V. Smirnov, “Spanning tree of a multiple graph”, *Journal of Combinatorial Optimization*, vol. 43, no. 4, pp. 850–869, 2022. DOI: 10.1007/s10878-021-00810-5.

[4] A. V. Smirnov, “NP-completeness of the minimum spanning tree problem of a multiple graph of multiplicity $k \geqslant 3$”, *Automatic Control and Computer Sciences*, vol. 56, no. 7, pp. 788–799, 2022. DOI: 10.3103/S0146411622070173.

[5] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, 3rd. The MIT Press, McGraw-Hill Book Company, 2009.

[6] C. Berge, *Graphs and Hypergraphs*. North-Holland Publishing Company, 1973.

[7] A. Basu and R. W. Blanning, “Metagraphs in workflow support systems”, *Decision Support Systems*, vol. 25, no. 3, pp. 199–208, 1999. DOI: 10.1016/S0167-9236(99)00006-8.

[8] A. Basu and R. W. Blanning, *Metagraphs and Their Applications* (Integrated Series in Information Systems). Springer US, 2007, vol. 15.

[9] V. S. Rublev and A. V. Smirnov, "Flows in multiple networks", *Yaroslavsky Pedagogichesky Vestnik*, vol. 3, no. 2, pp. 60–68, 2011, in Russian.

[10] A. V. Smirnov, "The problem of finding the maximum multiple flow in the divisible network and its special cases", *Automatic Control and Computer Sciences*, vol. 50, no. 7, pp. 527–535, 2016. DOI: 10.3103/S0146411616070191.

[11] L. R. Ford and D. R. Fulkerson, *Flows in Networks*. Princeton University Press, 1962.

[12] V. S. Roublev and A. V. Smirnov, "The problem of integer-valued balancing of a three-dimensional matrix and algorithms of its solution", *Modeling and Analysis of Information Systems*, vol. 17, no. 2, pp. 72–98, 2010, in Russian.

[13] A. V. Smirnov, "Network model for the problem of integer balancing of a four-dimensional matrix", *Automatic Control and Computer Sciences*, vol. 51, no. 7, pp. 558–566, 2017. DOI: 10.3103/S0146411617070185.

[14] J. B. Kruskal, "On the shortest spanning subtree of a graph and the traveling salesman problem", *Proceedings of the American Mathematical Society*, vol. 7, no. 1, pp. 48–50, 1956. DOI: 10.1090/S0002-9939-1956-0078686-7.