Графы в компьютерных науках

Симоненко Е.А.
easimonenko@mail.ru

14 июня 2023 г.

Симоненко Е.А. Графы в компьютерных науках. – Санкт-Петербург: 2018, Краснодар: 2020. – 51 с.

В книге рассматриваются различные аспекты обработки графов на компьютерах и применения графов в компьютерных науках.

В репозитории `https://github.com/easimonenko/graphs-in-computer-science` находятся полные исходные тексты книги в LaTeX, а также сборка в формате PDF для печати на бумаге формата A4. Вы также можете адаптировать эту книгу для другого подходящего вам формата.

[1] https://creativecommons.org/licenses/by-nd/4.0/legalcode

Оглавление

Предисловие

Теория графов – один из самых молодых разделов дискретной математики. Возникновение теории графов обязано работе Эйлера[2], датированной 1736 годом, опубликованной в 1741 году и посвящённой решению задачи о Кёнигсбергских мостах (смотри [Фляйшнер]). Первое комплексное изложение теории графов было сделано венгерским математиком Денешом Кёнигом в книге «Theorie der endlichen und unendlichen Graphen» (Теория конечных и бесконечных графов) [Кёниг], изданной в 1936 году. Первой книгой по теории графов на русском языке стал перевод с французского книги Берж «Теория графов и её применения» [Берж], выпущенный в 1962 году. Второй, также переводной книгой, стала книга норвежского математика Ойстина Оре «Теория графов» (1968) [Оре]. С тех пор было издано огромное число книг как переводных, так и написанных советскими и российскими математиками, здесь отмечу, например, [Окулов] и [Ландо].

В этой книге, «Графы в компьютерных науках», рассматриваются различные аспекты обработки графов на компьютерах и применения их в компьютерных науках. Главное отличие этой книги о графах в том, что она не является пространным изложением теории графов или её отдельных аспектов. В современной теории графов, да и вообще в теории графов, много интересных задач и результатов, некоторые из которых можно узнать из современных учебников по теории графов или дискретной математике. Однако на момент написания этой книги автор не обнаружил легко доступного обобщения и систематического изложения вопросов работы с графами на компьютере и их применения в компьютерных науках, что и сподвигло к написанию этой книги параллельно занятиям этой тематикой. В главе «Основные понятия теории графов» даются все необходимые базовые определения и результаты, благодаря которым книгу можно читать без предварительного изучения теории графов.

В репозитории `https://github.com/easimonenko/graphs-in-computer-science` находятся полные исходные тексты книги в LaTeX, а также сборка в формате PDF для печати на бумаге формата A4. Вы также можете адаптировать эту книгу для себя для другого подходящего вам формата. Эта книга распространяется под лицензией CC BY-ND 4.0. Если вы хотите внести изменения и распространять изменённый вариант этой книги, вам нужно обратиться к автору за получением отдельного разрешения. Вы можете произвести исправления и дополнения к этой книге и передать их автору этой книги для рассмотрения им возможного внесения этих изменений в исходный вариант при сохранении лицензии на эту книгу.

[2]Эти строки я пишу всего в паре кварталов по набережной Лейтенанта Шмидта от дома на Неве в Санкт-Петербурге, где жил Леонард Эйлер, великий российский и германский математик.

Глава 1

Основные понятия теории графов

1.1. Основные понятия

Графом G называют пару конечных множеств $G = (V, E)$ [1], где элементы множества V называются *вершинами*, а элементы множества E представляют собой пары элементов из множества V и называются *рёбрами*. Если рёбра в графе представляют собой неупорядоченные пары $\{u, v\}$, то соответствующий граф называют *неориентированным*. Если рёбра – упорядоченные пары (u, v), то граф называют *ориентированным* (орграфом). В случае орграфов рёбра также называют *дугами*. Говорят также, что вершины u и v *соединены* ребром.

Неформально граф является совокупностью некоторых сущностей, имеющих между собой связи. Например, это могут быть населённые пункты и другие объекты, соединённые между собой дорогами. Или социальная сеть, где люди дружат между собой или подписаны на других.

Часто графы представляются графическими изображениями, где кружки или точки изображают вершины, а линии, соединяющие их, рёбра, как, например, на рисунке 1.1.

Петлёй называют ребро соединяющее одну и ту же вершину. Граф, у которого есть пара вершин, соединённых двумя или более рёбрами, называют *мультиграфом*. Граф на рисунке 1.1 является мультиграфом. Граф, не имеющий петель и не являющийся мультиграфом, называют *простым*.

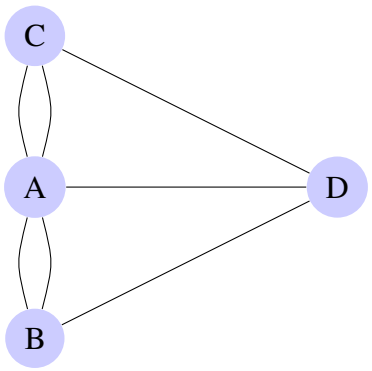

Рис. 1.1: Граф задачи о Кёнигсбергских мостах

$$G = (\{A, B, C, D\}, \{\{A, B\}^2, \{A, C\}^2, \{A, D\}, \{B, D\}, \{C, D\}\}) \tag{1.1}$$

[1] V от английского слова Vertices (вершины), а E от Edges (рёбра).

Заметим, что мы не накладываем ограничение на мощность множеств V и E. Если V и E являются конечными множествами, то говорят, что граф *конечный*, и это обычная ситуация, так что, если не уточняется, то подразумевается, что граф конечный. В случае конечного V, но бесконечного E, мы имеем граф с бесконечным числом кратных рёбер или петель. В случае с бесконечным V, но конечным E, имеем граф с бесконечным числом изолированных вершин. Наконец, если и V, и E бесконечны, то такой граф называют *бесконечным*.

Для дальнейшего удобства будем полагать, что $|V| = n$ и $|E| = m$ для случая, когда множество V или E конечно.

Граф называется *помеченным*, если его вершинам приписаны различные метки. Обычно в качестве меток используются натуральные числа в диапазоне от 1 до n, где $n = |V|$, часто также применяются строчные латинские буквы. При необходимости рёбра также могут быть помечены.

Вершины, соединённые ребром, называются *смежными*. Рёбра, имеющие общую вершину, также называются *смежными*. Ребро и любая из его двух вершин называются *инцидентными*. Говорят также, что ребро *инцидентно* своим вершинам.

Каждый граф можно представить на плоскости в виде множества точек, соответствующих вершинами, которые соединены линиями, соответствующими рёбрам. В трёхмерном пространстве любой граф можно представить таким образом, что линии, соответствующие рёбрам, не будут пересекаться.

Граф, у которого множество $E = \emptyset$, то есть, в котором отсутствуют рёбра, называется *нуль-графом*.

Граф называется *полным*, если любые две его вершины смежны. Будем обозначать полный граф с n вершинами символом K_n.

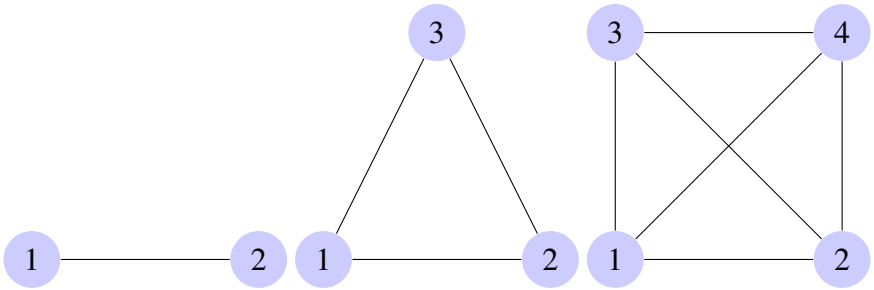

Рис. 1.2: Примеры полных графов: K_2, K_3, K_4

Число рёбер в полном графе равно $C_n^2 = \frac{n \cdot (n-1)}{2}$. Количество помеченных графов с фиксированным множеством вершин V, $|V| = n$, равно количеству подмножеств множества рёбер полного графа, то есть $2^{C_n^2}$.

Граф H называется *подграфом* графа G, если все его вершины и рёбра принадлежат графу G. *Остовный* подграф – это подграф графа G, содержащий все его вершины. *Клика* графа – это его максимальный полный подграф.

Два графа G и H *изоморфны* ($G \simeq H$), если между множествами их вершин можно установить взаимно однозначное соответствие, при котором сохраняется отношение смежности. Это означает, что в различных представлениях эти графы могут выглядеть разными, но структура у них будет одинаковой.

Дополнением $\tilde{G}$ графа $G = (V, E)$ называется граф со множеством вершин V, две вершины которого смежны тогда и только тогда, когда они не смежны в G.

Степенью вершины графа G называется число инцидентных ей рёбер. Максимальная степень

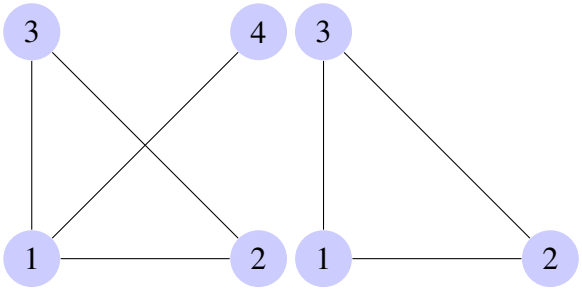

Рис. 1.3: Пример клики графа

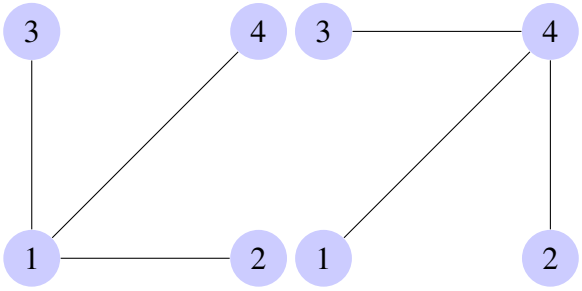

Рис. 1.4: Пример изоморфных графов

вершин графа G обозначается $\Delta(G)$, а минимальная – $\delta(G)$. Вершина степени 0 называется *изолированной*, степени 1 – *концевой* или *висячей*. Ребро, инцидентное концевой вершине, также называют *концевым* или *висячим*. На рисунке 1.5 вершина 1 графа $\tilde{G}$ изолированная, а вершина 4 графа G – висячая.

Граф G, у которого $\Delta(G) = \delta(G)$ называют *регулярным* или *однородным*. Очевидно, что любой полный граф является регулярным, также и нуль-граф.

Теорема 1 (Лемма о рукопожатиях)**.** *Сумма степеней всех вершин графа равна удвоенному числу рёбер.*

Доказательство. Каждое ребро увеличивает степень одновременно двух вершин. Таким образом, если начать с нулевого количества рёбер, при котором степени всех вершин будут равны 0, то каждое из «добавляемых» рёбер будет увеличивать сумму степеней вершин на 2. □

Граф называется *транзитивным*, если всегда из того, что вершины u и v, v и w соединены ребром, при этом u, v и w различны, следует, что также и вершины u и w соединены ребром. Транзитивный граф, получающийся из исходного путём добавления недостающих рёбер, называется *транзитивным замыканием*. См. рисунок 1.7.

1.1.1. Упражнения

1. Докажите формулу $C_n^2 = \frac{n \cdot (n-1)}{2}$ для количества рёбер графа K_n.
2. Докажите формулу $2^{C_n^2}$ для количества помеченных графов с n вершинами.
3. Охарактеризуйте графы, заданные графически (рис. 1.8):
4. Найдите клики в графах на рис. 1.8.
5. Найдите максимальные клики в графе G_1 (рис. 1.8).

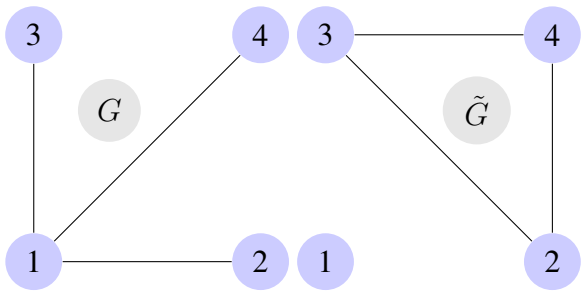

Рис. 1.5: Пример графа и его дополнения

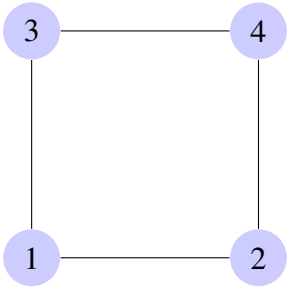

Рис. 1.6: Пример регулярного графа

6. Изобразите граф K_6.

7. Найдите дополнения до полного графа для графов на рис. 1.8.

1.2. Граф как бинарное отношение

Бинарным отношением R между множествами X и Y называется подмножество произведения $X \times Y$, $R \subseteq X \times Y$, и обозначается XRY. Если $X = Y$, то бинарное отношение называется *отношением на множестве* X, $R \subseteq X^2$. Если x находится в отношении с y, то пишут xRy. Примерами отношений являются: отношение равенства, неравенство, эквивалентность, отношение порядка.

Очевидно, что с точки зрения теории множеств граф является бинарным отношением.

Множество X бинарного отношения $R \subseteq X \times Y$ называется *областью определения отношения* R и обозначается $\mathrm{Dom}R$. Множество Y, соответственно, называется *областью значения отношения* R и обозначается $\mathrm{Im}R$.

Множество обратных пар отношения R, $\{(y, x) \mid (x, y) \in R\}$, называется *обратным* или *инверсией* и обозначается R^{-1}.

Композиция (или *суперпозиция*) бинарных отношений R и S это множество $\{(x, z) \mid \exists y(xRy \wedge ySz)\}$. Композиция обозначается $R \circ S$.

Бинарное отношение R называется:

– *рефлексивным*, если $\forall x \in X\colon (x, x) \in R$;

– *антирефлексивным* или иррефлексивным, если $\forall x \in X\colon (x, x) \notin R$;

– *корефлексивным*, если $\forall x, y \in X\colon xRy \Rightarrow x = y$;

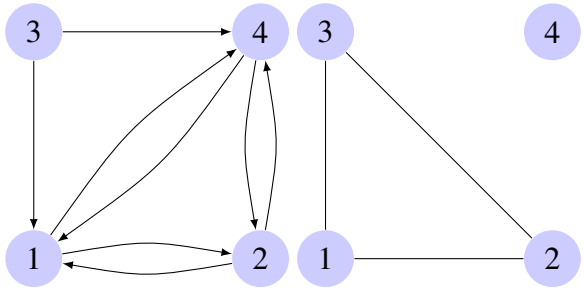

Рис. 1.7: Примеры транзитивных графов

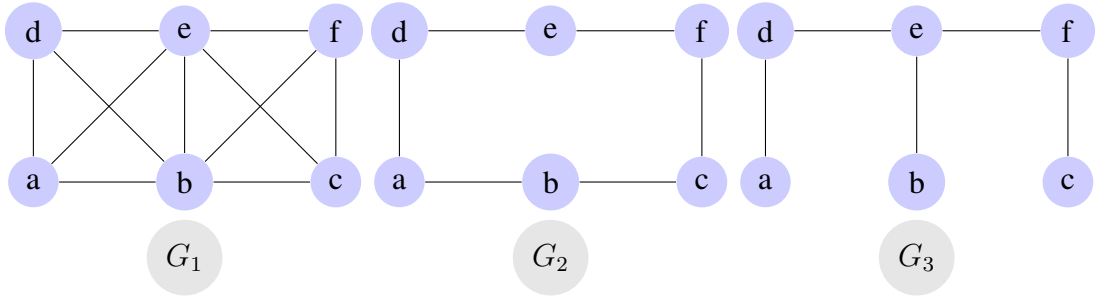

Рис. 1.8: Графы G_1, G_2, G_3

– *симметричным*, если $\forall x, y \in X, x \neq y$: $(x, y) \in R \Rightarrow (y, x) \in R$;

– *антисимметричным*, если $\forall x, y \in X$: $(x, y) \in R, (y, x) \in R \Rightarrow x = y$;

– *асимметричным*, если $\forall x, y \in X : (x, y) \in R \Rightarrow \neg(y, x) \in R$;

– *транзитивным*, если $\forall x, y, z \in X$: $(x, y) \in R, (y, z) \in R \Rightarrow (x, z) \in R$;

– *евклидовым*, если $\forall x, y, z \in X : (x, y) \in R \wedge (x, z) \in R \Rightarrow (y, z) \in R$;

– *полным (или универсальным)*, если $\forall x, y \in X, x \neq y$: $(x, y) \in R \vee (y, x) \in R$;

– *нулевым*, если $\forall x, y \in X$: $(x, y) \neq R$.

– *коннексным*, если $\forall x, y \in X : (x, y) \in R \vee (y, x) \in R \vee x = y$;

– *имеющим трихотомию*, если $\forall x, y \in X : (x, y) \in R \oplus (y, x) \in R \oplus x = y$.

Бинарное отношение может быть задано с помощью *матрицы бинарного отношения*, в которой 1 обозначает наличие пары (x, y) в бинарном отношении, и 0 – её отсутствие. В теории графов такую матрицу называют *матрицей смежности*. В случае мультиграфов вместо нулей и единиц указывают количество рёбер между соответствующими вершинами.

Рефлексивность с точки зрения графов означает, что у каждой вершины графа есть петля. Антирефлексивность, напротив, запрещает петли; простой граф этому удовлетворяет. Симметричности удовлетворяют неориентированные графы, и ориентированные графы, если каждой дуге соответствует противоположно направленная ей. Антисимметричность, напротив, это запрещает. Таким образом, ни один неориентированный простой граф антисимметричностью не обладает, однако, граф, у которого есть только петли, уже удовлетворяет. Ориентированные графы, у которых для любой дуги нет противоположно направленной, также удовлетворяют. Транзитивному отношению соответствуют так называемые транзитивные графы. Их мы рассмотрим ниже. Полному графу соответствует универсальное отношение. Нуль-графу соответствует нулевое отношение.

Можно заметить, что матрицы смежности графов будут обладать следующими свойствами:

- при наличии петель на главной диагонали будут попадаться числа, отличные от нуля;
- у простого графа на главной диагонали всегда находятся нули;
- матрица неориентированного простого графа симметрична относительно главной диагонали; симметричность для ориентированного графа означает присутствие дуг обоих направлений;
- у полного графа все элементы матрицы смежности, кроме, возможно, диагональных, ненулевые;
- у нуль-графа матрица нулевая.

1.3. Представление графов

Здесь мы рассмотрим следующие классические способы представления графов:

- матрица смежности;
- матрица инцидентности;
- список связности;
- список рёбер.

Указанные способы рассмотрим на примере помеченного простого графа, представленного на рисунке 1.9.

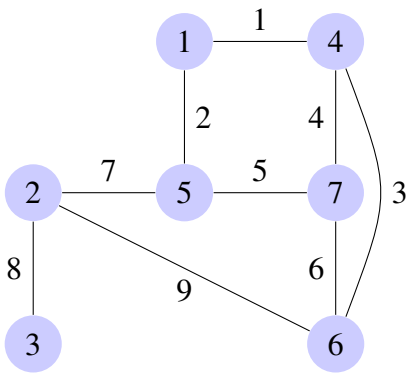

Рис. 1.9: Пример графа для рассмотрения способов представления

1.3.1. Матрица смежности

Мы уже затрагивали матрицу смежности графа, здесь же ещё раз проговорим основные моменты и составим матрицу смежности для нашего примера графа.

Матрица смежности содержит информацию о смежности всех пар вершин. Индексы в этой матрице являются номерами вершин, а элементы матрицы содержат 1, если вершины с соответствующими индексам номерами смежные, и 0, если – нет. В случае с неориентированными

графами матрица смежности является симметричной относительно главной диагонали. В случае с орграфами это уже может не соблюдаться. Кроме того, в случае с графами без петель на главной диагонали будут всегда располагаться нули, а в случае кратных рёбер вместо единицы будет записываться кратность рёбер. Это представление графов обусловлено тем, что граф является бинарным отношением, а матрица смежности – не что иное как матрица бинарного отношения. Ниже пример матрицы смежности для графа из рисунка 1.9.

	1	2	3	4	5	6	7
1	0	0	0	1	1	0	0
2	0	0	1	0	1	1	0
3	0	1	0	0	0	0	0
4	1	0	0	0	0	1	1
5	1	1	0	0	0	0	1
6	0	1	0	1	0	0	1
7	0	0	0	1	1	1	0

Таблица 1.1: Пример матрицы смежности

1.3.2. Матрица инцидентности

Матрица инцидентности содержит информацию об инцидентности вершин рёбрам. Первый из индексов в этой матрице является пометкой вершины, а второй – пометкой ребра (вместо пометки можно использовать пару соответствующих вершин). Если вершина i инцидентна ребру j, то соответствующий элемент матрицы с индексом (i, j) содержит 1, в противном случае – 0. Ниже пример матрицы инцидентности для графа из рисунка 1.9.

	1	2	3	4	5	6	7	8	9
1	1	1	0	0	0	0	0	0	0
2	0	0	0	0	0	0	1	1	1
3	0	0	0	0	0	0	0	1	0
4	1	0	1	1	0	0	0	0	0
5	0	1	0	0	1	0	1	0	0
6	0	0	1	0	0	1	0	0	1
7	0	0	0	1	1	1	0	0	0

Таблица 1.2: Пример матрицы инцидентности

1.3.3. Список связности

Список связности представляет собой список списков, содержащий n элементов (по количеству вершин графа), каждый из которых является списком вершин смежных с данной. Ниже пример списка связности для графа из рисунка 1.9.

1.3.4. Список рёбер

Список рёбер представляет собой простое перечисление рёбер с указанием инцидентных им вершин. Ниже пример списка рёбер для графа из рисунка 1.9.

1	4	5	
2	3	5	6
3	2		
4	1	6	7
5	1	2	7
6	2	4	7
7	4	5	6

Таблица 1.3: Пример списка связности

1	1	4
2	1	5
3	4	6
4	4	7
5	5	7
6	6	7
7	2	5
8	2	3
9	2	6

Таблица 1.4: Пример списка рёбер

1.3.5. Упражнения

1. Представьте изображённые на рис. 1.8 графы разными способами (множествами, матрицами, списками).

1.4. Обход графа в глубину

Под обходом графа понимается просмотр (посещение) вершин графа в определённом порядке с целью получения дополнительной информации о нём. Примеры применения обхода графа будут рассмотрены позже. Здесь же рассмотрим обход графа в глубину и обход графа в ширину в общем виде.

При обходе графа в глубину начинают с некоторой вершины и просматривают по очереди все вершины, смежные с ней. Для каждой из этих вершин этот процесс просмотра повторяется. Другими словами, при обходе графа в глубину происходит максимально возможное продвижение по графу от начальной вершины с последующим возвратом в неё. Для избегания зацикливания уже просмотренные вершины помечаются и в дальнейшем в обходе в глубину они уже не участвуют. В некоторых задачах не достаточно помечать посещённые вершины. В таких случаях чаще всего применяется раскраска вершин графа в один из трёх цветов: белый (white), серый (gray), чёрный (black). До обхода графа в глубину все его вершины «красятся» в белый цвет. При посещении вершины она приобретает серый цвет. А при возвратном прохождении через вершину она красится в чёрный. Посещать разрешается только белые вершины, серый цвет служит признаком прохождения вершины при обходе в глубину с последующим возвратом через неё.

Рассмотрим обход в глубину на примере графа 1.9 раннее рассмотренного в теме «Представление графов». Начнём с вершины номер 1. Тогда последовательность просмотра вершин будет следующей (полужирным выделены просматриваемые вершины, а курсивом – через кото-

рые возвращаемся назад; верхняя строка нумерует вершины в порядке посещения, включая те, что посещены при возврате):

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	**4**	**6**	**2**	**3**	*3*	*2*	**5**	**7**	*7*	*5*	*2*	*6*	*4*	*1*

Таблица 1.5: Пример порядка посещения вершин при обходе в глубину

Процесс «раскраски» вершин на каждом шаге представлен в таблице 1.6 (рамками выделим вершину, в которой мы находимся на данном шаге) и на рисунке 1.10 (стрелки обозначают переход от одной вершины к другой, а номер над стрелкой порядок перехода). На шаге 7 произошёл возврат в вершину 2, из которой продолжился обход в глубину через вершину номер 5. На шаге 12 также произошёл возврат в вершину номер 2, но так как других смежных с ней и ещё не посещённых вершин больше нет, то обход в глубину из вершины номер 2 закончился, и произошёл возврат в вершину номер 6. После завершения обхода графа в глубину все его вершины приобрели чёрный цвет.

	1	2	3	4	5	6	7
0	w	w	w	w	w	w	w
1	g	w	w	w	w	w	w
2	g	w	w	g	w	w	w
3	g	w	w	g	w	g	w
4	g	g	w	g	w	g	w
5	g	g	g	g	w	g	w
6	g	g	b	g	w	g	w
7	g	g	b	g	w	g	w
8	g	g	b	g	g	g	w
9	g	g	b	g	g	g	g
10	g	g	b	g	g	g	b
11	g	g	b	g	b	g	b
12	g	b	b	g	b	g	b
13	g	b	b	g	b	b	b
14	g	b	b	b	b	b	b
15	b	b	b	b	b	b	b

Таблица 1.6: Пример процесса раскраски вершин при обходе в глубину

Обратите внимание, что три ребра оказались не пройденными, это нормально, так как обход в глубину сосредоточен на посещении всех вершин, а не рёбер.

Программно обход в глубину легче всего реализовать рекурсивным методом продвижения вперёд с возвратами (по-английски backtracking). Вместо рекурсии можно воспользоваться структурой данных стек. Преимуществом этого метода является несколько лучшая производительность и независимость от настройки глубины системного стека, который неявным образом используется при рекурсивном методе.

Обход в глубину также носит название *Depth-First Search (DFS)*.

На рис. 1.11 дано описание алгоритма обхода в глубину на псевдокоде.

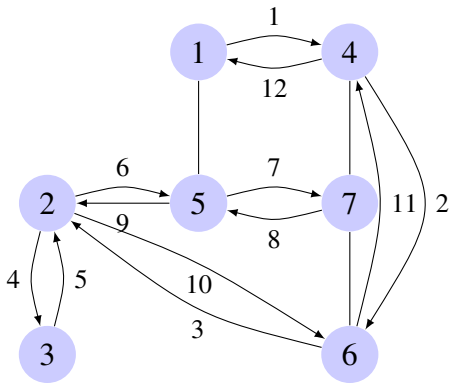

Рис. 1.10: Пример процесса раскраски вершин при обходе в глубину

DFS:

Вход:

```
  граф Graph, массив цветов вершин Colors, номер стартовой вершины U.
```

Начало.

```
  Красим вершину U в серый цвет: Colors[U] := Grey.
  Для всех вершин V графа Graph, смежных с U:
    Если цвет V белый (Colors[V] == White):
      DFS(Graph, Colors, V).
  Красим вершину U в чёрный цвет: Colors[U] := Black.
```

Конец.

Рис. 1.11: Псевдокод алгоритма обхода в глубину

1.4.1. Упражнения

1. Осуществите обход графов на рис. 1.8 в глубину и предъявите соответствующие деревья обхода:

   (a) G_1 начиная с вершин a и b.

   (b) G_2 начиная с вершины a.

   (c) G_3 начиная с вершин a, b, e, d.

1.5. Обход графа в ширину

При обходе графа в ширину начинают с некоторой вершины и просматривают одновременно все вершины, смежные с ней. Так как обеспечить одновременность просмотра невозможно, то смежные с данной вершины помещают в очередь на обработку. Затем процесс повторяется для поставленных в очередь вершин. Также как и при обходе в глубину необходимо раскрашивать вершины графа в процессе обработки в разные цвета, например, с тою целью, чтобы не повторять обработку уже обработанных вершин, если они встретятся среди смежных ещё раз (а они как минимум встретятся повторно один раз).

Обход в ширину также носит название *Breadth-First Search (BFS)*.

Рассмотрим обход в ширину на примере графа 1.9. Начнём с вершины номер 1. Тогда последовательность просмотра вершин будет следующей (верхняя строка нумерует вершины в порядке посещения; в скобках указывается вершина, из которой видна очередная; прочерк означает, что из данной вершины не видно ещё не обработанных вершин):

1	2 (1)	3 (1)	4 (4)	5 (4)	6 (5)	7 (6)	8 (7)	9 (2)
1	4	5	6	7	2	–	–	3

Таблица 1.7: Пример порядка посещения вершин при обходе в ширину

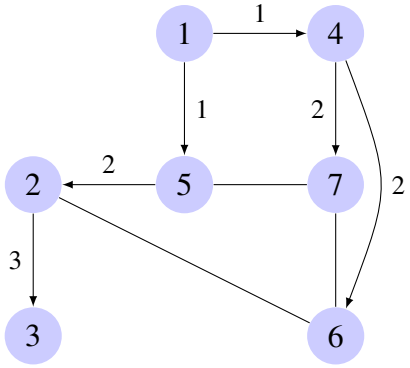

Рис. 1.12: Пример процесса раскраски вершин при обходе в ширину

Описание алгоритма обхода графа в ширину на псевдокоде смотри ниже 1.13.

1.5.1. Упражнения

1. Осуществите обход графов на рис. 1.8 в ширину и предъявите соответствующие деревья обхода:

```
BFS:

Вход:
  граф Graph, массив цветов вершин Colors, номер стартовой вершины U.

Начало.

  Создаём пустую очередь Queue.
  Помещаем в Queue вершину U.
  Красим вершину U в серый цвет: Colors[U] := Gray.
  Пока Queue не пуста:
    Извлекаем из очереди очередную вершину U.
    Для всех вершин V графа Graph, смежных с U:
      Если цвет V белый (Colors[V] == White):
        Помещаем в Queue вершину V.
        Красим вершину V в серый цвет: Colors[V] := Gray.
      Красим вершину U в чёрный цвет: Colors[V] := Black.

Конец.
```

Рис. 1.13: Псевдокод алгоритма обхода в ширину

(a) G_1 начиная с вершин a и b.

(b) G_2 начиная с вершины a.

(c) G_3 начиная с вершин a, b, e, d.

1.6. Маршруты, циклы и расстояния

Последовательность смежных рёбер графа называют *маршрутом*. Если все рёбра маршрута различны, то его называют *цепью*. Цепь называется *простой*, если вершины рёбер кроме смежных и, возможно, начальных и конечных не повторяются. Если начальная вершина первого ребра цепи совпадает с конечной вершиной последнего ребра, то цепь называется *циклической* или *циклом*.

Для представления маршрутов, цепей и циклов можно использовать как последовательность рёбер, так и последовательность вершин, так как пары соседних вершин в такой последовательности полностью определяют соответствующие рёбра.

Для выявления наличия в графе цикла можно воспользоваться обходом графа в глубину с раскраской вершин в три цвета. Признаком наличия цикла будет обнаружение чёрной вершины среди ещё непосещённых. Почему это так: обозначим обнаруженную чёрную вершину v, а вершину, смежную с ней, и из которой мы её обнаружили чёрной, u, тогда это означает, что будет построен маршрут из u в v, который замкнёт ребро $\{u, v\}$ (см. рис. 1.14).

Описание алгоритма обнаружения наличия цикла в графе на псевдокода на рис. 1.15.

Длина маршрута это количество рёбер этого маршрута. *Расстоянием между двумя вершинами* будем называть наименьшую длину маршрута между ними. Очевидно, что маршрут наименьшей длины будет являться простой цепью.

[TODO: алгоритм нахождения кратчайшего пути обходом в ширину.]

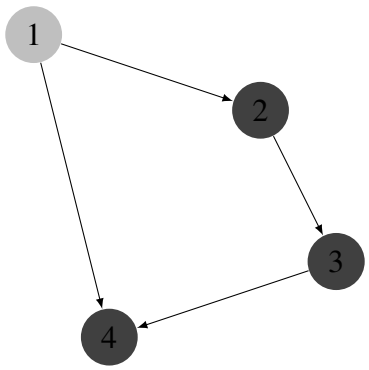

Рис. 1.14: Граф с циклом

CYCLE_DETECTION:

Вход:
граф Graph, массив цветов вершин Colors, номер стартовой вершины U.

Выход:
True, если цикл обнаружен; False, если цикла нет.

Начало.

```
Красим вершину U в серый цвет: Colors[U] := Grey.
Для всех вершин V графа Graph, смежных с U:
  Если цвет V чёрный (Colors[V] == Black):
    Возвращаем True.
  Если цвет V белый (Colors[V] == White):
    Возвращаем результат CYCLE_DETECTION(Graph, Colors, V).
Красим вершину U в чёрный цвет: Colors[U] := Black.
Возвращаем False.
```

Конец.

Рис. 1.15: Псевдокод алгоритма обнаружения наличия цикла в графе

Диаметром графа называют наибольшее среди расстояний между всеми парами вершин этого графа. Диаметр графа 1.9 равен 3.

Радиусом графа называется наименьшее среди всех наибольших расстояний от каждой вершины графа до всех остальных. Радиус графа 1.9 равен 2.

Понятия диаметра и радиуса графа можно сформулировать через понятие эксцентриситета вершины. *Эксцентриситетом* $\epsilon(v)$ вершины v называют следующую величину:

$$\epsilon(v) = \max_{u \in V} d(v, u),$$

где $d(v, u)$ — расстояние между вершинами v и u.

Тогда радиус $r(G)$ и диаметр $D(G)$ графа G можно определить так:

$$r(G) = \min_{v \in V} \epsilon(v), D(G) = \max_{v \in V} \epsilon(v)$$

Также на основе понятия эксцентриситета мы можем ввести понятия центральной и периферийной вершины. Назовём вершину *центральной*, если её эксцентриситет равен радиусу графа. Соответственно, назовём вершину *периферийной*, если её эксцентриситет равен диаметру графа. В графе 1.9 вершина 5 является центральной, а вершина 3 — периферийной.

1.6.1. Упражнения

1. Докажите наличие или отсутствие циклов в представленных выше графах. (Примените метод обхода графа в глубину.)

2. Найдите кратчайшие расстояния от каждой вершины графа 1.9 до всех остальных. (Примените метод обхода графа в ширину.)

1.7. Связность

Граф называется *связным*, если любая его пара вершин соединена маршрутом. Максимальный связный подграф называется *компонентой связности*.

Для выяснения связности графа, а также для подсчёта числа компонент связности можно использовать обход графа в глубину или в ширину. Если в результате обхода графа останется хотя бы одна непосещённая вершина (вершина белого цвета), то граф несвязен. Для подсчёта числа компонент связности необходимо повторять обход графа до тех пор, пока не останется непосещённых вершин, количество таких обходов и будет искомым числом (естественно, что очередной обход нужно начинать с очередной ещё непосещённой вершины).

Степень и характер связности графов может существенно различаться. Различают вершинную и рёберную связность.

Вершинная связность $\chi(G)$ это наименьшее количество вершин связного графа, удаление которых нарушает эту связность. Для полных графов $\chi(K_n) = n - 1$, для несвязных графов $\chi(G) = 0$.

Рёберная связность $\lambda(G)$ это наименьшее число рёбер связного графа, удаление которых нарушает эту связность. Для полных графов $\lambda(K_n) = n - 1$, для несвязных графов $\lambda(G) = 0$.

Точкой сочленения называют вершину графа, если её удаление приводит к увеличению числа компонент связности. Для графа G, имеющего точку сочленения $\chi(G) = 1$. Граф, не имеющий точек сочленения, называют *блоком*. Граф G называют k*-связным*, если $\chi(G) \geq k$.

Граф, не совпадающий с K_1, односвязен тогда и только тогда, когда он связен, двусвязен, если он не содержит точек сочленения.

Граф 1.9 имеет одну точку сочленения, это вершина 2. Если её удалить, то появится дополнительная компонента связности, состоящая из единственной вершины 3.

Мостом графа называют ребро, удаление которого приводит к увеличению числа компонент связности. Для графа G, имеющего мост, $\lambda(G) = 1$.

Мостом в графе 1.9 является ребро $\{2, 3\}$.

1.8. Деревья

Деревом в теории графов называют связный ациклический граф. *Ациклический* означает без циклов.

В любом дереве количество рёбер на единицу меньше количества вершин: $m = n - 1$. Доказывается это по индукции. Простейшее дерево состоит из одной вершины и нуля рёбер. Добавление к нему второй вершины приведёт к появлению единственного ребра. Добавление третьей вершины позволит добавит ровно одно ребро. Не добавлять рёбра нельзя так как дерево по определению связно, а добавление больше одного ребра приведёт к возникновению в графе циклов, что также противоречит определению дерева.

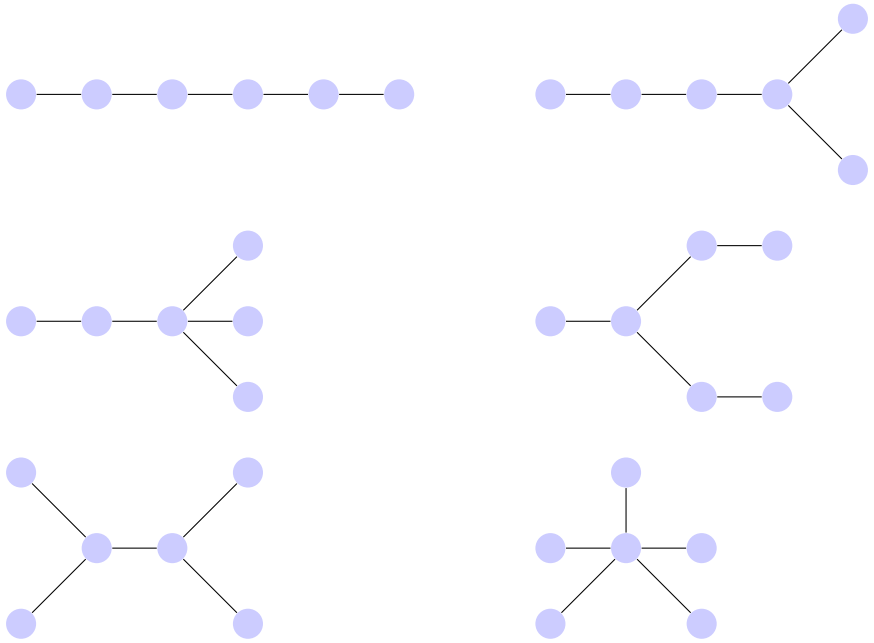

Рис. 1.16: Пример всех деревьев с шестью вершинами

Стягивающим деревом, или *остовным деревом*, или *каркасом графа* $G = (V, E)$ называют любое дерево (V, T), $T \subseteq E$.

1.9. Другие понятия и свойства графов

Порождённый (индуцированный) подграф это подграф $G' = (V', E')$ некоторого графа $G = (V, E)$, образованный некоторым подмножеством вершин графа G, $V' \subset V$, и множеством всех рёбер этого графа, инцидентных этим вершинам. Также говорят, что подграф G' индуцирован множеством вершин V'. Например, для графа на 1.9 подграф, индуцированный вершинами $V' = \{1, 4, 5, 6, 7\}$ выглядит как на рис. 1.17, вершины и рёбра индуцированного подграфа выделены особо.

Окрестностью вершины v в графе G называется подграф графа G, порождённый множеством вершин, смежных с v. Если мы хотим включить во множество вершин, порождающих окрестность, также и саму вершину v, то тогда назовём окрестность вершины *закрытой*. Ради различения, будем также называть окрестность вершины *открытой*, если вершину v мы всё же не включаем в это множество, что соответствует, собственно, определению окрестности вершины. Например, для графа на 1.9 окрестностью вершины 7 будет подграф изображённый на рисунке ниже (1.18). Открытую окрестность вершины v обозначают $N_G(v)$, закрытую – $N_G[v]$.

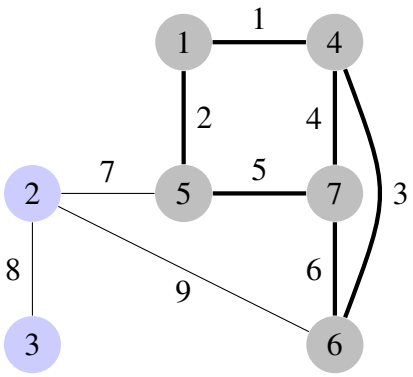

Рис. 1.17: Индуцированный подграф

Можно также рассматривать *окрестность множества вершин*, которая представляет собой объединение окрестностей, соответствующих каждой из вершин этого множества.

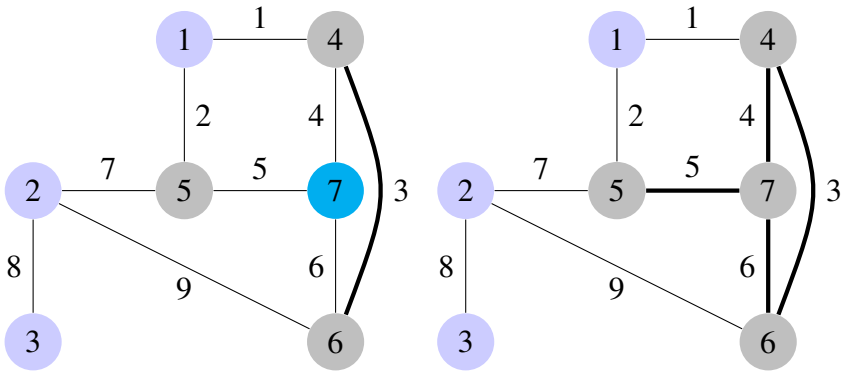

Рис. 1.18: Окрестности вершины (открытая и закрытая, соответственно)

[TODO: обхват графа]

[TODO: хроматическое число графа]

1.9.1. Упражнения

1. Найдите в графе G_1 подграфы, порождённые следующими множествами вершин:
   (a) $\{a, b, c, e\}$
   (b) $\{a, b, c, d, f\}$
   (c) $\{a, b, d, e\}$
2. Найдите в графе G_1 окрестности (открытые и закрытые) следующих вершин:
   (a) $\{a\}$
   (b) $\{e\}$

1.10. Эйлеровы графы

Эйлеровым циклом называется цикл, проходящий по каждому ребру графа ровно один раз. Соответственно, граф, содержащий такой цикл, называется *эйлеровым*.

Сформулируем критерий эйлеровости графа:

Теорема 2 (Эйлер, 1736г.)**.** *Связный неориентированный граф содержит эйлеров цикл тогда и только тогда, когда число вершин нечётной степени равно нулю.*

Если эйлеров цикл в графе существует, то это означает, что следуя вдоль этого цикла, можно нарисовать граф на бумаге, не отрывая от неё карандаша.

Пусть дан неориентированный граф G, удовлетворяющий условию теоремы Эйлера. Рассмотрим алгоритм нахождения эйлерова цикла в этом графе. Этот алгоритм основан на обходе графа в глубину. При переходе в очередную вершину будем удалять соответствующее пройденное ребро. При обнаружении вершины, из которой не выходят рёбра (мы их удалили раннее при обходе в глубину), будем записывать её номер в стек. Обнаружение вершины с нулевым числом рёбер говорит о том, что найден цикл. Его можно удалить, при этом чётность степеней вершин не изменится. Процесс продолжается до тех пор, пока есть не пройденные рёбра. После окончания обхода в глубину всего графа в стеке будут записаны номера вершин графа в порядке, соответствующем эйлерову циклу.

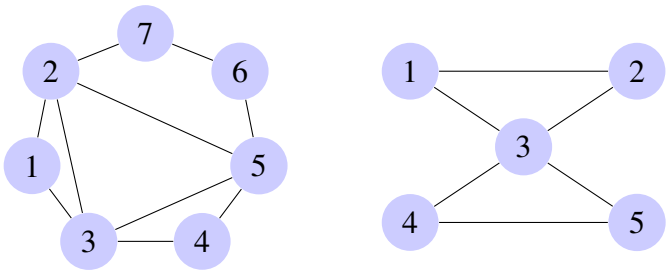

Рис. 1.19: Примеры эйлеровых графов

Euler:

Вход:
```
  граф Graph, представленный матрицей смежности;
  стек вершин эйлерова цикла Stack;
  номер стартовой вершины U.
```

Начало.

```
  Для всех вершин V графа Graph, смежных с U:
    Graph[U,V] := 0, Graph[V,U] := 0.
    Euler(Graph, Stack, V).
  Помещаем в Stack вершину U.
```

Конец.

В первом графе на рисунке 1.19, если начнём с вершины 1, то получим следующий эйлеров цикл: (1, 3, 5, 6, 7, 2, 5, 4, 3, 2, 1).

1.11. Гамильтоновы графы

Простой цикл называется *гамильтоновым*, если он содержит все вершины графа. Соответственно, граф, содержащий такой цикл, называется *гамильтоновым*. *Гамильтоновой* также называют простую цепь, если она проходит через все вершины графа. Этот класс графов называют

так в честь ирландского математика Уильяма Гамильтона, который в 1859 году предложил игру «Кругосветное путешествие», в которой требуется найти на графе додекаэдра (рисунок 1.20) простой цикл, проходящий через все вершины графа.

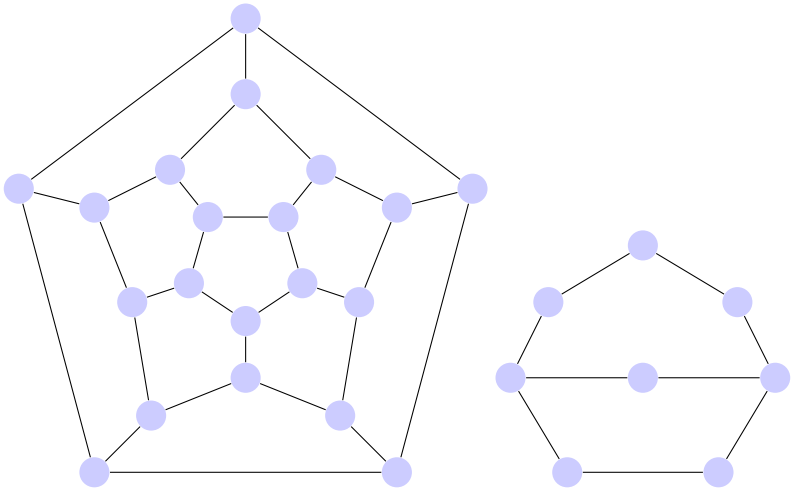

Рис. 1.20: Граф додекаэдра и пример тэта-графа

В отличие от графов Эйлера для гамильтоновых графов нет простого и ясного их описания.

Тэта-графом называется граф, содержащий только вершины степени 2 и две несмежные вершины степени 3.

Теорема 3. *Каждый гамильтонов граф двусвязен. Каждый негамильтонов двусвязный граф содержит тэта-подграф.*

Теорема 4 (У. Татт, 1946г.)**.** *Всякий 4-связный планарный граф является гамильтоновым.*

С задачей поиска гамильтоновых циклов тесно связана *задача о коммивояжёре* (о бродячем торговце). Формулируется она так: есть n городов, расстояния между которыми известны. Торговец должен посетить все n городов по одному разу, вернувшись в тот, с которого начал свой путь. Требуется найти путь для коммивояжёра с минимальным суммарным расстоянием.

1.12. Планарные графы

Говорят, что граф *укладывается* на какой-то поверхности, если его можно нарисовать на этой поверхности без пересечения рёбер. Если граф можно уложить на плоскости, то такой граф называют *планарным*. Уложенный на плоскости граф называют *плоским*. Некоторые просто устроенные графы являются планарными, например, графы K_1, K_2, K_3, K_4 1.21, любая простая цепь. Однако уже K_5 нельзя уложить.

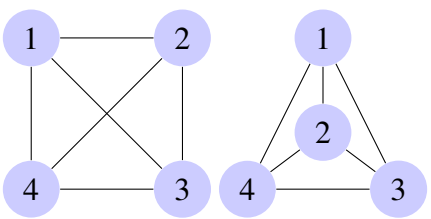

Рис. 1.21: Граф K_4 на плоскости

Область, ограниченная рёбрами плоского графа и не содержащая внутри себя других вершин и рёбер этого графа, называется *гранью*. Внешняя часть плоскости относительно графа также считается гранью. Число граней плоского графа G обозначим как $r(G)$.

Теорема 5 (Формула Эйлера)**.** *В связном плоском графе* $G = (V, E)$ *справедливо равенство* $n - m + r = 2$*, где* $n = |V|$*,* $m = |E|$ *и* $r = r(G)$.

Если G – связный плоский граф, $n > 3$, то $m \leq 3 \cdot n - 6$. Действительно, каждая грань ограничена по крайней мере тремя рёбрами, а каждое ребро ограничивает не более двух граней, следовательно, $3 \cdot r \leq 2 \cdot m$. Отсюда и из формулы Эйлера: $2 = n - m + r \leq n - m + 2 \cdot m/3$. Далее: $3 \cdot n - 3 \cdot m + 2 \cdot m \geq 6 \Rightarrow m \leq 3 \cdot n - 6$.

Покажем, что графы K_5 и $K_{3,3}$ не являются планарными. Для графа K_5 имеем $n = 5$ и $m = 5 \cdot 4/2 = 10$. Подставим n и m в неравенство: $10 \leq 3 \cdot 5 - 6 \Rightarrow 10 \leq 9$. Получили противоречие, граф K_5 не может быть планарным.

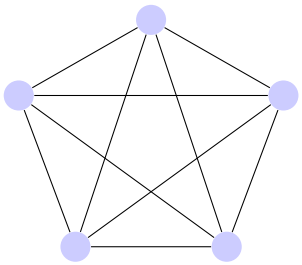

Рис. 1.22: Граф K_5

Для графа $K_{3,3}$ имеем $n = 6$ и $m = 9$. В этом графе нет «треугольников», поэтому при укладке на плоскость каждая грань ограничена не менее чем четырьмя рёбрами и, следовательно, $4 \cdot r \leq 2 \cdot m \Rightarrow 2 \cdot r \leq m$. По формуле же Эйлера: $6 - 9 + r = 2 \Rightarrow r = 5$. Подставляем r в неравенство: $2 \cdot 5 \leq 9 \Rightarrow 10 \leq 9$. Получили противоречие.

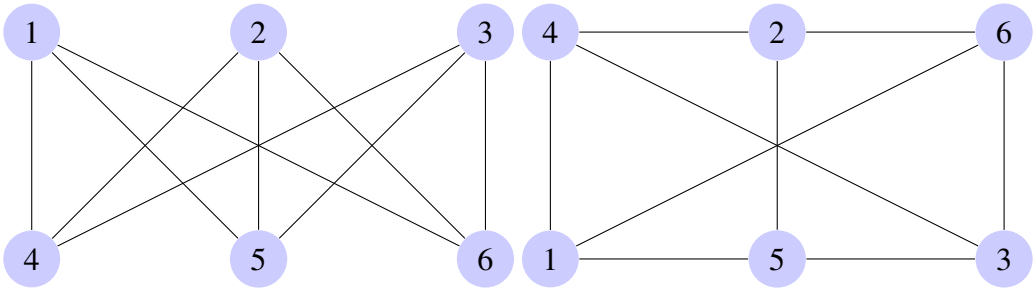

Рис. 1.23: Граф $K_{3,3}$

Плоский граф, у которого все грани, включая внешнюю, являются треугольниками, называют *плоской триангуляцией*. *Плоским максимальным графом* называется граф, который перестаёт быть плоским при добавлении любого ребра.

Теорема 6. *Граф является плоским максимальным графом тогда и только тогда, когда он является плоской триангуляцией.*

Для всякого максимального планарного графа выполняется равенство $m = 3 \cdot n - 6$.

1.13. Покрытие и независимость

Говорят, что вершина графа *покрывает* инцидентные ей рёбра, а ребро графа *покрывает* инцидентные ему вершины. При этом возникают две задачи: 1) поиск минимального числа вершин, покрывающих все рёбра графа G (*число вершинного покрытия* $\alpha_0(G)$); 2) поиск минимального числа рёбер, покрывающих все вершины графа G (*число рёберного покрытия* $\alpha_1(G)$).

Для полного графа K_n:

$$\alpha_0(K_n) = n - 1, \alpha_1(K_n) = \left\lceil \frac{n}{2} \right\rceil$$

В графе $K_{3,3}$ 1.23 можно выделить два покрывающих множества вершин: $\{1, 2, 3\}$ и $\{4, 5, 6\}$. Покрывающих множеств рёбер можно выделить несколько, например:
$\{\{1, 4\}, \{2, 5\}, \{3, 6\}\}$, $\{\{1, 5\}, \{2, 6\}, \{3, 4\}\}$.

$$\alpha_0(K_{3,3}) = 3, \alpha_1(K_{3,3}) = 3$$

Множество вершин графа G называют *независимым*, если никакие две вершины в этом множестве не смежны. Наибольшее число вершин в независимом множестве называют *вершинным числом независимости* $\beta_0(G)$, а соответствующее множество *наибольшим*. Множество несмежных рёбер графа G называют *независимым*. Наибольшее число рёбер в независимом множестве называют *рёберным числом независимости* $\beta_1(G)$.

Для полного графа K_n:

$$\beta_0(K_n) = 1, \beta_1(K_n) = \left\lfloor \frac{n}{2} \right\rfloor$$

В графе $K_{3,3}$ 1.23 есть два независимых множества вершин: $\{1, 2, 3\}$ и $\{4, 5, 6\}$. Независимых множеств рёбер можно выделить несколько, например:
$\{\{1, 4\}, \{2, 5\}, \{3, 6\}\}$, $\{\{1, 5\}, \{2, 6\}, \{3, 4\}\}$.

$$\beta_0(K_{3,3}) = 3, \beta_1(K_{3,3}) = 3$$

Теорема 7. *Для любого графа без изолированных вершин выполняются равенства:*

$$\alpha_0 + \beta_0 = n = \alpha_1 + \beta_1$$

1.14. Ориентированные графы

В некоторых задачах, решаемых теорией графов, связи между объектами несимметричны, например, в сети дорог могут быть дороги с односторонним движением. Для таких случаев понятие графа требует видоизменения. Более того, некоторые понятия при этом теряют смысл.

Будем называть *ориентированным графом* или просто *орграфом* G пару $G = (V, E)$, где E — конечное множество, элементы которого называются *вершинами*, и V — множество упорядоченных пар (u, v), где $u, v \in V$, называемых *дугами*, вершина u при этом называется *началом* дуги, а v – *концом*. Графически дуги изображаются линиями со стрелками.

Исходящей степенью вершины u называют количество пар вида (u, v), $v \in V$, $(u, v) \in E$. *Входящей степенью* вершины u называют количество пар вида (v, u), $v \in V$, $(v, u) \in E$. Другими словами, исходящая степень вершины это сколько стрелок выходит из неё, а входящая – сколько входит.

Вершину называют *истоком*, если её входящая степень ноль. Вершину называют *стоком*, если её исходящая степень ноль.

Так как в орграфе между парой вершин может существовать маршрут только в одну сторону, так называемый *ориентированный маршрут*, то здесь различают виды связности. Понятие связности, аналогичное для неориентированных графов, носит название *сильной связности*. Если в исходном ориентированном графе заменить дуги на неориентированные рёбра, и такой граф окажется связным, то исходный граф будет называться *слабо-связным*.

Бесконтурным или *ациклическим* орграфом будем называть орграф, в котором отсутствуют ориентированные циклы. Бесконтурный орграф является обобщением понятия дерева.

Орграф называют *транзитивным*, если для каждой пары рёбер (u, v) и (v, w) имеется также и ребро (u, w). Граф называют *транзитивным замыканием* графа $G = (V, E)$, если помимо каждой пары рёбер $(u, v), (v, w) \in V$ в нём имеется также и ребро (u, w).

Теорема 8. *Ориентированный граф имеет эйлеров цикл тогда и только тогда, когда он связный и степень входа каждой вершины равна степени её выхода.*

Топологической сортировкой бесконтурного орграфа G называется перенумерация вершин графа G в соответствии с частичным порядком, заданным дугами орграфа G на множестве его вершин.

1.14.1. Упражнения

1. Охарактеризуйте орграфы, заданные графически (рис. 1.24):

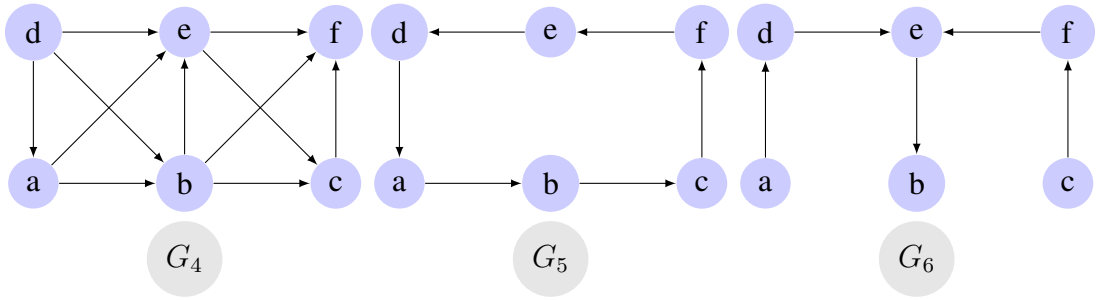

Рис. 1.24: Графы G_4, G_5, G_6

2. Представьте изображённые на рис. 1.24 графы разными способами (множествами, матрицами, списками).

3. Постройте транзитивное замыкание графов с рис. 1.24.

1.15. Литература

Об основных понятиях теории графов можно почитать в [Окулов]. Много интересных задач приводится в [Ландо].

Глава 2

Представление графов в компьютере

...

2.1. Список рёбер

Обычно список рёбер используется для хранения графа в файле. В этом случае часто рёбра не нумеруются явно. При считывании же списка рёбер из файла для представления графа в памяти компьютера используют один из трёх вышеописанных способов.

...

Глава 3

Графы зависимостей и их приложения

3.1. Основные понятия

...

3.2. Приложения

...

Глава 4

Визуализация графов

...

Глава 5

Анализ графов

...

Глава 6

Графовые базы данных

В этой главе рассматривается общая концепция графовых баз данных, структуры хранения графов с таких базах данных, а также некоторые алгоритмы выполнения запросов к графам. Для более подробной информации о существующих реализациях графовых БД и их языков запросов следует обратиться к специализированным руководствам.

...

Глава 7

Социальные сети

...

Приложение А

Англо-русский словарик

breadth-first search (BFS) обход в ширину
complete graph полный граф
connectivity связность
curvature кривизна
dependency graph граф зависимостей
depth-first search (DFS) обход в глубину
digraph орграф
directed graph ориентированный граф
graph граф
graph processing обработка графов
graph rewriting переписывание графа
dual graph двойственный граф
edge ребро
neighborhood окрестность
planar планарный
reversibility обратимость
vertice вершина

Приложение В

Русско-английский словарик

вершина vertice
граф graph
граф зависимостей dependency graph
двойственный граф dual graph
дополнение графа complimentary graph
кривизна curvature
обработка графов graph processing
окрестность neighborhood
орграф digraph
ориентированный граф directed graph
переписывание графа graph rewriting
планарный planar
преобразование графа graph transformation
ребро edge
связность connectivity

Приложение C

История изменений

- ***2023-06-13*** *(Краснодар)*
  - Возобновлена работа на книгой.
- ***2020-01-08*** *(Краснодар)*
  - Дополнена и исправлена глава ”Основные понятия теории графов”.
- ***2018-02-27*** *(Санкт-Петербург)*
  - Подготовлен общий макет книги.
  - Опубликована первая глава ”Основные понятия теории графов”.

Литература

[Кёниг] König D. Theorie der endlichen und unendlichen Graphen. – Leipzig, 1936. – 268 s.

[Берж] Берж К. Теория графов и её применения. – Пер. с фр. – М.: Издательство иностранной литературы, 1962. – 320 с.

[Ландо] Ландо С.К. Введение в дискретную математику. – М.: МЦНМО, 2012. – 265 с.

[Окулов] Окулов С.М. Дискретная математика. Теория и практика решения задач по информатике: учебное пособие. – М.: БИНОМ. Лаборатория знаний, 2008. – 422 с.

[Оре] Оре О. Теория графов. – Пер. с англ. – М.: НАУКА, 1968. – 352 с.

[Фляйшнер] Фляйшнер Г. Эйлеровы графы и смежные вопросы. – М.: Мир, 2002. – 335 с.

Список иллюстраций

Список таблиц