Элементарная комбинаторика

1 Основные правила перечислительной комбинаторики

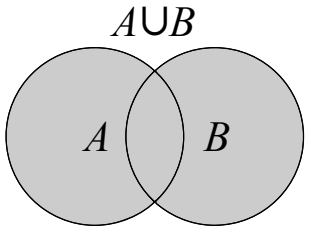
1.1. Начнем с очень краткого напоминания основных понятий теории множеств.

Определение 1.1. *Множеством* X называется совокупность различимых объектов x_i, объединенных по некоторому общему для них признаку. Иными словами, это совокупность каких-то объектов, которые мы полагаем принципиально различными и про которые мы можем четко сказать, входят они в данное множество или нет.

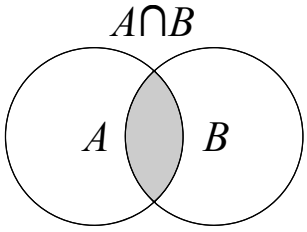
В качестве характерного примера можно рассмотреть, например, множество X всех студентов, находящихся в данной аудитории. Действительно, все студенты различимы, отличны друг от друга и объединены по признаку "собрались в данной аудитории".

Определение 1.2. Мощностью $|X|$ множества X называется количество элементов в нем. Как правило, мы будем рассматривать конечные множества, в которых $|X| = n$, $n \in \mathbb{N}$, и называть их n-множествами.

1.1.1. Основные операции над множествами — это объединение $A \cup B$ (рис.1,a), пересечение $A \cap B$ (рис.1,b), разность $A \setminus B$ (рис.2,a) и симметрическая разность (рис.2,b) двух множеств A и B. В случае, если множество A является подмножеством некоторого более широкого множества X, удобно также рассматривать операцию дополнения $A' := X \setminus A$ множества A (рис.2,c).

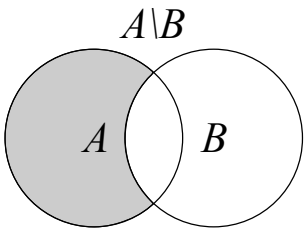


(a) Объединение

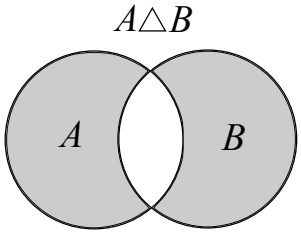


(b) Пересечение

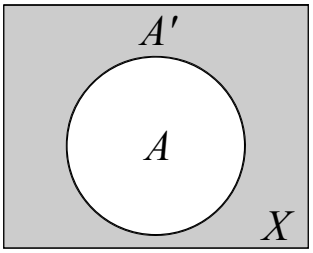
Рис. 1



(a) Разность



(b) Симметрическая разность



(c) Дополнение

Рис. 2

Свойства операций над множествами удобно изучать графически, с использованием так называемых *диаграмм Эйлера-Венна* (смотри рисунки 1 и 2). Например, с их помощью достаточно просто проиллюстрировать справедливость законов де Моргана

$$A' \cap B' = (A \cup B)', \qquad A' \cup B' = (A \cap B)' \tag{1}$$

(смотри рис.3).

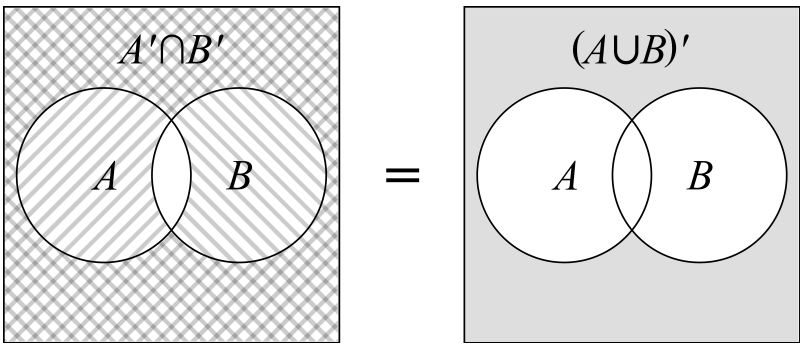


Рис. 3: Графическое доказательство закона де Моргана

1.1.2. В дальнейшем мы достаточно часто будем использовать понятие покрытия множества X семейством $\{X_1, X_2, \ldots, X_k\}$ множеств, а также всевозможные частные случаи этого понятия.

Определение 1.3. Семейство множеств $\{X_1, X_2, \ldots, X_k\}$ называется *покрытием* множества X, если их объединение дает нам все множество X:

$$X_1 \cup X_2 \cup \ldots \cup X_k = X.$$

Важным частным случаем покрытия является понятие разбиения множества.

Определение 1.4. Говорят, что семейство множеств $\{X_1, X_2, \ldots, X_k\}$ образует *разбиение* множества X, если

1. множества $X_i \neq \varnothing$, $i = 1, \ldots, n$;
2. $X_i \cap X_j = \varnothing \ \forall i \neq j$;
3. $X_1 \cup X_2 \cup \ldots \cup X_k = X$.

Элементы X_i этого семейства называются *блоками* разбиения.

В качестве характерного примера можно рассмотреть разбиение студентов данного курса на группы. Студенческие группы являются при этом блоками данного разбиения.

Если по каким-то причинам оказывается важным порядок следования блоков, то говорят об *упорядоченном разбиении* $(X_1, X_2, \ldots, X_k)$ множества X. Например, если мы выводим группы на сцену для вручения им дипломов, то важен порядок, в котором они туда выходят. Следовательно, в данном случае мы получаем упорядоченное разбиение студентов данного курса.

Наконец, еще одним частным случаем покрытия X семейством множеств $\{X_1, X_2, \ldots, X_k\}$ является понятие *разделения* множества X. Разделение есть аналог упорядоченного разбиения, в котором допускаются пустые блоки. Точное определение таково:

Определение 1.5. Разделением множества X называется упорядоченная последовательность $(X_1, X_2, \ldots, X_k)$ возможно пустых, попарно непересекающихся множеств, объединение которых дает все множество X.

1.1.3. Еще одной часто использующейся в комбинаторике операцией над множествами является операция декартова произведения множеств.

Определение 1.6. Декартовым произведением множеств A и B называется множество

$$A \times B := \{(a, b) \,|\, a \in A,\ b \in B\}$$

всех упорядоченных пар (a, b), таких, что $a \in A$, $b \in B$.

В качестве простейшего примера декартова произведения множеств обычно приводят шахматную доску. Любая клетка шахматной доски имеет координаты "буква-цифра", например, $e5$ или $h4$. Иными словами, координаты клеток шахматной доски являются элементами декартова произведения множеств $A = \{a, b, \ldots, h\}$ и $B = \{1, 2, \ldots, 8\}$.

В частном случае множества A и B могут совпадать. В этом случае декартово произведение $A \times A$ обозначается через A^2.

Определение 1.7. Декартовым произведением k множеств $X_1, X_2, \ldots, X_k$ называется множество

$$X_1 \times X_2 \times \ldots \times X_k := \{(x_1, x_2, \ldots, x_k) \,|\, x_i \in X_i,\ \forall\ i = 1, \ldots, k\}$$

всевозможных упорядоченных k-элементных последовательностей вида $(x_1, x_2, \ldots, x_k)$.

В частном случае $X_1 = X_2 = \ldots = X_k = X$ имеем декартову степень $X \times X \times \ldots \times X =: X^k$.

1.1.4. Любой элемент X^k есть упорядоченный набор из k элементов множества X, в котором некоторые элементы могут повторяться. Если же в таком k-множестве порядок элементов не важен, говорят о *мультимножестве над множеством* X. Формальное определение мультимножества таково.

Определение 1.8. Мультимножеством над n-элементным множеством X называется пара (X, φ), где $\varphi\colon X \to \mathbb{Z}_+$ есть функция, сопоставляющая любому элементу $x \in X$ количество $\varphi(x)$ его вхождений в мультимножество.

Любую функцию φ такого рода можно определить с помощью множества Ξ упорядоченных пар

$$\Xi := \{(x, \varphi(x)) \mid x \in X\}.$$

Поэтому, например, мультимножество над множеством $X = \{x, y\}$, состоящее из двух элементов x и одного элемента y, можно формально записывать в виде $(X, \{(x, 2), (y, 1)\})$. Однако чаще вместо такой формальной записи используют более наглядную форму записи вида $\{x, x, y\}$.

Самый простой и понятный пример мультимножества — это монеты в кошельке. В этом примере в качестве множества X выступает множество из девяти монет разного достоинства:

$$X = \{1 \text{ копейка}, 5 \text{ копеек}, 10 \text{ копеек}, 50 \text{ копеек}, 1 \text{ рубль}, 2 \text{ рубля}, 5 \text{ рублей}, 10 \text{ рублей}\}.$$

Любой набор из этих монет в количестве k штук образует k-мультимножество над множеством X.

1.1.5. Теория множеств как раздел математики создавалась значительно позже комбинаторики. Поэтому некоторые наиболее важные понятия теории множеств исторически получили в комбинаторике свои, специфические названия. Именно,

1. k-сочетанием без повторений называется любое k-элементное подмножество n-элементного множества;
2. k-сочетанием с повторениями называется любое k-мультимножество над n-множеством;
3. k-перестановкой без повторений называется упорядоченное k-подмножество n-элементного множества;
4. k-перестановкой с повторениями называется любой элемент декартовой степени X^k.

1.2. Теперь перейдем к двум самым простым, но в то же время достаточно важным правилам перечислительной комбинаторики — правилу суммы и правилу произведения.

1.2.1. Начнем с простейшего примера: пусть на одном блюде лежат три яблока, а на втором — две груши; сколькими способами можно выбрать один фрукт? Ответ очевиден: пятью способами.

Обобщающее этот пример простейшее *правило суммы* можно сформулировать так: если некоторый объект из множества A можно выбрать k способами, и, вне зависимости от выбора этого объекта, можно n способами выбрать некоторый элемент множества B, то выбор объекта из множества A *или* из множества B можно осуществить $k + n$ способами.

Очевидна переформулировка этого правила на языке теории множеств: пусть пересечение двух множеств A и B пусто; тогда

$$|A \cup B| = |A| + |B|.$$

В частности, если $A \subset X$ и A' — дополнение множества A, то

$$|A| + |A'| = |X|. \tag{2}$$

В более общем случае, рассматривая произвольное разбиение множества X на блоки, имеем равенство вида

$$|X| = |X_1| + |X_2| + \ldots + |X_k|,$$

которое также называется правилом суммы в комбинаторике.

1.2.2. Под правилом произведения в комбинаторике понимается равенство

$$|X_1 \times X_2 \times \ldots \times X_k| = |X_1| \cdot |X_2| \cdot \ldots \cdot |X_k|.$$

Приведем простейший пример на применение этого правила в комбинаторике. Пусть в аудитории находятся 32 студента одной группы, 24 студента другой группы и 17 студентов третьей группы. В этом случае тройку, состоящую из представителей каждой группы, можно выбрать $32 \cdot 24 \cdot 17$ способами.

1.3. Наряду с правилом суммы, в элементарной комбинаторике также достаточно часто используется и несложное его обобщение — так называемый *принцип включения-исключения*. Если правило суммы связано с разбиением множества X, то принцип включения-исключения связан с некоторым произвольным покрытием этого n-множества семейством $\{X_1, X_2, \ldots, X_k\}$. Сформулируем его для самого простого случая двух множеств.

1.3.1. Рассмотрим два конечных множества A и B, пересечение которых может быть и непусто. Тогда количество элементов в объединении этих множеств, очевидно, равно

$$|A \cup B| = |A| + |B| - |A \cap B|. \tag{3}$$

Действительно, когда мы считаем количество $|A|$ элементов в множестве A и складываем его с количеством $|B|$ элементов в множестве B, мы любой элемент, принадлежащий как множеству A, так и множеству B, считаем дважды. Чтобы этот избыток убрать, нам нужно один раз вычесть количество элементов, содержащихся в пересечении этих двух множеств.

Равенство (3) можно называть обобщенным правилом суммы — оно обобщает правило суммы на случай, когда пересечение двух множеств не пусто.

1.3.2. Предположим теперь, что A и B являются подмножествами некоторого более широкого множества X. В этом случае у множества $A \subset X$ и множества $B \subset X$ имеются дополнения к ним — множества A' и B', причем $A \cup A' = B \cup B' = X$.

Рассмотрим теперь пересечение $A' \cap B'$ дополнений множеств A и B. Согласно одной из теорем де Моргана (1), $A' \cap B' = (A \cup B)'$. Следовательно, количество элементов в этом пересечении с учетом равенства (2) и обобщенного правила суммы (3) можно сосчитать так:

$$|A' \cap B'| = |(A \cup B)'| = |X| - |A \cup B| = |X| - |A| - |B| + |A \cap B|.$$

Равенство

$$|A' \cap B'| = |X| - |A| - |B| + |A \cap B|, \tag{4}$$

и называется в комбинаторике принципом включения-исключения.

1.3.3. Приведем простейший пример его использования. Пусть в аудитории находятся 30 человек, 20 человек из которых знают английский, 12 — французский, а 6 человек знают оба языка. Сколько человек не знает ни один из этих иностранных языков? Ответ, согласно принципу включения-исключения (4), следующий:

$$N = 30 - 20 - 12 + 6 = 4.$$

1.3.4. Несложно обобщить равенства (3) и (4) на случай большего количества множеств. Например, для трех множеств A, B, C соответствующие формулы выглядят так:

а) обобщенное правило суммы:

$$|A \cup B \cup C| = |A| + |B| + |C| - |A \cap B| - |A \cap C| - |B \cap C| + |A \cap B \cap C|; \tag{5}$$

б) принцип включения-исключения:

$$|A' \cap B' \cap C'| = |X| - |A| - |B| - |C| + |A \cap B| + |A \cap C| + |B \cap C| - |A \cap B \cap C|. \tag{6}$$

Действительно, рассмотрим, к примеру, левую часть равенства (5). Она подсчитывает количество элементов, принадлежащих объединению трех множеств (смотри рис.4). Если элемент x_1, принадлежащий этому объединению, содержится в множестве A, но не содержится в множествах B и C, то он один раз подсчитывается в правой части равенства (5) (слагаемое $|A|$, отвечающее зеленой подобласти на рис.4). Если элемент x_2 принадлежит множествам A и B, но не принадлежит C (красная подобласть на рис.4), то в правой части (5) этот элемент входит в слагаемые $|A|$, $|B|$ и $-|A \cap B|$, то есть также подсчитывается ровно один раз. Наконец, если x_3 принадлежит пересечению всех трех множеств (зеленая подобласть на рис.4), то за этот элемент отвечают все слагаемые в правой части (5). Так как четыре из них входят со знаком плюс, а три — со знаком минус, то этот элемент также считается в правой части (5) лишь однажды.

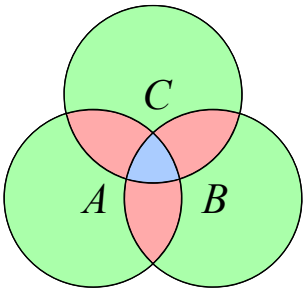


Рис. 4: Диаграмма Эйлера-Венна для трёх множеств

2 Подсчет k-сочетаний из n элементов. Биномиальные коэффициенты.

2.1. Основная задача данного параграфа состоит в подсчете количества всех k-сочетаний из n элементов. Начнем мы с подсчета количества k-сочетаний из n элементов без повторений.

2.1.1. Число k-сочетаний без повторений известно в литературе под названием биномиальных коэффициентов. Ранее в советской литературе такие числа обозначались через C_n^k. В настоящее время для этих коэффициентов используется обозначение $\binom{n}{k}$ (читается "из n по k").

2.1.2. Обычно на вопрос, чему равны биномиальные коэффициенты, вспоминают формулу

$$\binom{n}{k} = \frac{n!}{k!\,(n-k)!}.$$

Эта формула не очень удачна с двух точек зрения — с вычислительной и с идейной. С вычислительной точки зрения ее затруднительно использовать для достаточно больших значений n и k. Например, при $n = 38$ и $k = 19$ числитель и знаменатель могут просто не уместиться в диапазон изменения целых чисел для того или иного языка программирования. С идейной же точки зрения эта формула неудобна потому, что она не позволяет обобщить понятие биномиальных коэффициентов на случай целых, вещественных или комплексных значений n. В следующем параграфе мы получим более удобное выражение для этих коэффициентов, допускающее такое обобщение. Здесь же мы с помощью правила суммы выведем рекуррентное соотношение для биномиальных коэффициентов, позволяющее эти биномиальные коэффициенты эффективно вычислять для достаточно больших значений n и k.

2.1.3. Для получения рекуррентного соотношения введем множество Σ_k всех k-элементных подмножеств n-элементного множества X. Например, для $X = \{x_1, x_2, x_3\}$ множество $\Sigma_2 = = \{\{x_1, x_2\}, \{x_1, x_3\}, \{x_2, x_3\}\}$. Заметим, что биномиальный коэффициент $\binom{n}{k}$ как раз и описывает мощность такого множества. Разобьем теперь множество Σ_k на два блока — блок $\Sigma_k^{(1)}$, k-элементные подмножества которого содержат элемент x_1, и блок $\Sigma_k^{(2)}$, подмножества которого этот элемент не содержат. Понятно, что это — непустые, непересекающиеся подмножества, объединение которых дает нам все множество Σ_k. Поэтому по правилу суммы мы получаем равенство вида

$$\binom{n}{k} = |\Sigma_k| = |\Sigma_k^{(1)}| + |\Sigma_k^{(2)}|.$$

Осталось выразить через биномиальные коэффициенты количество элементов в каждом из блоков $\Sigma_k^{(1)}$, $\Sigma_k^{(2)}$. А это делается довольно легко.

2.1.4. Действительно, во всех подмножествах первого блока элемент x_1 уже выбран, и нам из оставшегося $(n-1)$-элементного множества $X \setminus x_1$ остается выбрать $(k-1)$-элементные подмножества . Сделать это можно $\binom{n-1}{k-1}$ способами. Во втором блоке содержатся k-элементные подмножества множества $X \setminus x_1$, состоящего из $(n-1)$-го элемента. Их количество, очевидно, равно $\binom{n-1}{k}$. Таким образом окончательно имеем следующее рекуррентное соотношение:

$$\binom{n}{k} = \binom{n-1}{k-1} + \binom{n-1}{k}, \qquad k \geqslant 1, \quad n \geqslant 1. \tag{7}$$

2.1.5. Соотношение (7) следует дополнить начальными и граничными условиями. Так как k-элементных подмножеств n-элементного множества в случае $k > n$ не существует, то

$$\binom{n}{k} = 0 \qquad \text{при} \quad k > n.$$

Далее, пустое подмножество можно выбрать всегда и только одним способом; поэтому

$$\binom{n}{0} = 1 \qquad \forall\ n \geqslant 0.$$

Используя эти условия, можно шаг за шагом вычислить коэффициенты $\binom{n}{k}$. Часто их записывают в виде так называемого треугольника Паскаля:

$$\begin{array}{ccccccccc}
 & & & & 1 & & & & \\
 & & & 1 & & 1 & & & \\
 & & 1 & & 2 & & 1 & & \\
 & 1 & & 3 & & 3 & & 1 & \\
1 & & 4 & & 6 & & 4 & & 1 \\
 & & & & \dots & & & &
\end{array}$$

2.1.6. Как видно, треугольник Паскаля симметричен, т.е. $\binom{n}{k} = \binom{n}{n-k}$. Комбинаторное доказательство этого факта очевидно. Действительно, выбирая любое k-элементное множество, мы тем самым однозначно выбираем и дополнение к нему, т.е. $(n-k)$-элементное множество. Следовательно, количество k-элементных и $(n-k)$-элементных подмножеств совпадает.

2.2. Наряду с треугольником Паскаля биномиальные коэффициенты допускают и еще одно очень удобное графическое представление — представление на координатной плоскости (n, k)

2.2.1. Данное представление связано со следующей довольно интересной комбинаторной задачей. Рассмотрим плоскость (n, k) и нарисуем на этой плоскости пути, исходящие из начала координат, приходящие в точку с координатами (n, k), $n \geqslant 0$, $k = 0, \dots, n$, и состоящие из диагональных и вертикальных отрезков (рис.5). В самих точках (n, k) отметим количество таких путей, приходящих в эту точку из начала координат. Заметим теперь, что попасть в точку с координатами (n, k) мы можем, пройдя только лишь через точку с координатами $(n-1, k)$ или через точку с координатами $(n-1, k-1)$. С комбинаторной точки зрения это означает, что количество путей $\binom{n}{k}$, приходящих в точку с координатами (n, k), равняется количеству $\binom{n-1}{k}$ путей, приходящих в точку с координатами $(n-1, k)$, плюс количество путей $\binom{n-1}{k-1}$, приходящих в точку с координатами $(n-1, k-1)$. Но равенство

$$\binom{n}{k} = \binom{n-1}{k} + \binom{n-1}{k-1}$$

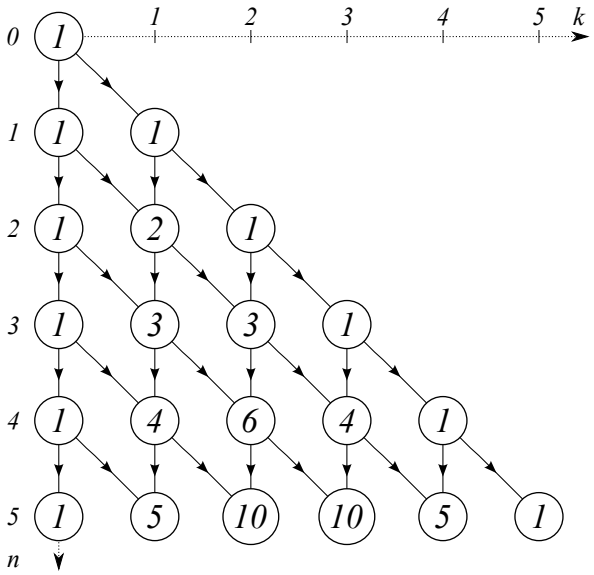


Рис. 5: Графическое представление чисел $\binom{n}{k}$ на координатной плоскости (n, k)

есть основное тождество для биномиальных коэффициентов (7). А это означает, что мы получили новую комбинаторную интерпретацию таких коэффициентов. Именно, числа $\binom{n}{k}$ описывают количество путей, состоящих из диагональных $(1, 1)$ и вертикальных $(1, 0)$ отрезков, выходящих из начала координат — точки $(0, 0)$, и оканчивающихся в точке с координатами (n, k).

2.2.2. Графическое представление чисел $\binom{n}{k}$ на плоскости (n, k) очень удобно для получения разного рода тождеств с биномиальными коэффициентами. В качестве первого примера зафиксируем какое-то конкретное значение параметра k и просуммируем биномиальные коэффициенты $\binom{m}{k}$ по m от k до некоторого фиксированного значения n. Например, выберем $k = 1$, $n = 4$. Складывая числа $1, 2, 3, 4$, мы получим число 10, то есть биномиальный коэффициент, стоящий правее и ниже рассмотренной цепочки биномиальных коэффициентов. Есть подозрение, что данный факт, а именно, равенство вида

$$\binom{k}{k} + \binom{k+1}{k} + \ldots + \binom{n}{k} = \binom{n+1}{k+1} \tag{8}$$

выполняется для любых значений параметров n и k.

2.2.3. Для формального доказательства тождества

$$\sum_{m=0}^{n} \binom{m}{k} = \underbrace{\binom{0}{k} + \binom{1}{k} + \ldots + \binom{k-1}{k}}_{=0} + \binom{k}{k} + \binom{k+1}{k} + \ldots + \binom{n}{k} = \sum_{m=k}^{n} \binom{m}{k} = \binom{n+1}{k+1},$$

называемого формулой суммирования биномиальных коэффициентов по верхнему индексу, применим рекуррентное соотношение (7) к коэффициенту $\binom{m+1}{k+1}$:

$$\binom{m+1}{k+1} = \binom{m}{k} + \binom{m}{k+1} \qquad \Longrightarrow \qquad \binom{m}{k} = \binom{m+1}{k+1} - \binom{m}{k+1}.$$

Просуммируем теперь полученное равенство по m от k до n:

$$\sum_{m=k}^{n}\binom{m}{k} = \binom{n+1}{k+1} + \sum_{m=k}^{n-1}\binom{m+1}{k+1} - \sum_{m=k}^{n}\binom{m}{k+1} =$$
$$= \binom{n+1}{k+1} + \sum_{m=k}^{n-1}\binom{m+1}{k+1} - \sum_{m=k+1}^{n}\binom{m}{k+1} =$$
$$= \binom{n+1}{k+1} + \sum_{m'=k+1}^{n}\binom{m'}{k+1} - \sum_{m=k+1}^{n}\binom{m}{k+1} = \binom{n+1}{k+1}.$$

2.2.4. Комбинаторное доказательство тождества (8) основано на следующем общем подходе: мы разбиваем множество Σ_{k+1} всех $(k+1)$-элементных подмножеств $(n+1)$-элементного множества X на блоки, подсчитываем количество элементов в каждом блоке, а затем пользуемся правилом суммы для подсчета числа $|\Sigma_{k+1}| = \binom{n+1}{k+1}$.

Для реализации этого подхода возьмем $(n+1)$-элементное множество

$$X = \{x_1, x_2, \ldots, x_{k-1}, x_k, x_{k+1}, \ldots, x_{n-1}, x_n, x_{n+1}\}$$

и будем разбивать множество всех $(k+1)$-элементных подмножеств множества X следующим образом. В первый блок поместим все $(k+1)$-элементные подмножества, содержащие элемент x_{n+1}. Такие подмножества элемент x_{n+1} гарантированно содержат. Нам из оставшихся элементов $x_1, \ldots, x_n$ множества $X \setminus x_{n+1}$ нужно выбрать недостающие k элементов. По определению биномиального коэффициента, сделать мы это можем $\binom{n}{k}$ способами.

Теперь мы рассмотрим все $(k+1)$-элементные подмножества, которые не содержат элемент x_{n+1}, но обязательно содержат элемент x_n. Для того, чтобы любое такое подмножество сформировать, нам нужно из множества элементов $\{x_1, \ldots, x_{n-1}\}$ выбрать недостающие k элементов. Это можно сделать $\binom{n-1}{k}$ количеством способов.

Затем рассмотрим все $(k+1)$-элементные подмножества, которые содержат в обязательном порядке элемент x_{n-1} и не содержат элементов x_{n+1} и x_n. Эти подмножества получаются выбором недостающих k элементов из множества элементов $\{x_1, \ldots, x_{n-2}\}$, и выбрать такие подмножества можно $\binom{n-2}{k}$ способами.

Продолжая далее, мы дойдем когда-то до ситуации, в которой нам нужно выбрать все $(k+1)$-элементные подмножества, которые содержат элемент x_{k+1} и не содержат элементы со старшими индексами. Так как элемент x_{k+1} у нас уже выбран, то нам остается из множества $\{x_1, \ldots, x_k\}$ выбрать k-элементное подмножество. Сделать мы это можем, очевидно, $\binom{k}{k} = 1$ способом.

Складывая теперь количество элементов в каждом блоке, мы и получаем тождество (8).

Пример 2.1. Пусть $n = 4$, $X = \{x_1, x_2, x_3, x_4, x_5\}$, $k = 2$, $k+1 = 3$. Приведем список всех трехэлементных подмножеств этого множества:

$$\{x_1, x_2, x_3\}, \quad \{x_1, x_2, x_4\}, \quad \{x_1, x_2, x_5\}, \quad \{x_1, x_3, x_4\}, \quad \{x_1, x_3, x_5\},$$
$$\{x_1, x_4, x_5\}, \quad \{x_2, x_3, x_4\}, \quad \{x_2, x_3, x_5\}, \quad \{x_2, x_4, x_5\}, \quad \{x_3, x_4, x_5\}.$$

В первый блок разбиения этого множества подмножеств включим подмножества, содержащие элемент x_5; таковых имеется $\binom{4}{2} = 6$ штук. Из *оставшегося* списка выберем все подмножества, содержащие x_4; их $\binom{3}{2} = 3$ штуки. Наконец, у нас остается единственное подмножество элементов, не содержащих ни x_4, ни x_5, т.е. подмножество $\{x_1, x_2, x_3, \}$.

2.2.5. В заключение данного пункта докажем комбинаторно еще одно важное тождество для биномиальных коэффициентов — так называемое тождество Вандермонта

$$\binom{n+m}{k} = \sum_{i=0}^{k} \binom{n}{i} \cdot \binom{m}{k-i}.$$

Рассмотрим для этого группу, состоящую из n мужчин и m женщин. По определению, количество способов выбрать из нее команду, состоящую из k человек, равно биномиальному коэффициенту $\binom{n+m}{k}$. С другой стороны, мы можем выбирать эту команду так, чтобы в ней было ровно i мужчин и $(k-i)$ женщин. При фиксированном i количество способов подбора такой команды, согласно правилу произведения, равно $\binom{n}{i} \cdot \binom{m}{k-i}$. Меняя теперь i от нуля до k, мы вновь получим общее количество способов образовать требуемую команду.

Замечание 2.2. Приведенные выше рассуждения очень часто используются в комбинаторике. Они даже имеют специальное название — *принцип double counting* или правило подсчета двумя способами. Основная идея такого рода рассуждений состоит в следующем: если две формулы подсчитывают количество одних и тех же элементов, то эти формулы равны.

2.3. Название "биномиальные коэффициенты" связано с тем, что они, помимо всего прочего, встречаются в формуле бинома Ньютона

$$(x+y)^n = \sum_{k=0}^{n} \binom{n}{k} x^k \, y^{n-k}. \tag{9}$$

2.3.1. Комбинаторное доказательство этой формулы довольно элементарно: нужно просто расписать $(x+y)^n$ в виде произведения n сомножителей

$$(x+y)^n = \underbrace{(x+y)}_{1} \cdot \underbrace{(x+y)}_{2} \cdot \ldots \cdot \underbrace{(x+y)}_{n}$$

и пометить каждый из таких сомножителей числом в диапазоне от единицы до n. В результате мы имеем множество X, состоящее из n *различных* экземпляров сомножителей вида $(x+y)$.

После перемножения этих n скобок получается определенный набор слагаемых вида $x^k y^{n-k}$, $k = 0, 1, \ldots, n$. Для подсчета количества этих слагаемых при фиксированном значении параметра k заметим, что любое слагаемое $x^k y^{n-k}$ можно получить так: выбрать в n-элементном множестве X k-элементное подмножество, взять в этом подмножестве в качестве сомножителей переменные x, а в оставшемся $(n-k)$-элементном подмножестве выбрать в качестве сомножителей переменные y. Как следствие, количество слагаемых $x^k y^{n-k}$ совпадает с количеством способов выбрать k-элементное подмножество n-элементного множества X и равно $\binom{n}{k}$.

2.3.2. Формула (9) оказывается чрезвычайно полезной для вывода разного рода соотношений, связанных с биномиальными коэффициентами. Например, полагая в ней $x = y = 1$, получаем тождество

$$\sum_{k=0}^{n} \binom{n}{k} = 2^n. \tag{10}$$

Заметим теперь, что, суммируя биномиальные коэффициенты $\binom{n}{k}$ по k, мы подсчитываем *все* подмножества n-элементного множества. Иными словами, мы формально доказали тот факт, что количество всех подмножеств данного n-множества равно 2^n.

2.3.3. Для комбинаторного доказательства равенства (10) воспользуемся чрезвычайно полезным и часто использующимся в комбинаторике *принципом биекции.* Формально этот принцип можно сформулировать следующим образом: пусть X, Y — пара конечных множеств, и пусть существует биекция $f\colon X \to Y$, т.е. такое отображение, что

$$\forall\, y \in Y \qquad \exists!\, x \in X : \qquad y = f(x).$$

Тогда количество элементов в множествах X и Y совпадают: $|X| = |Y| = n$.

Неформально использование принципа биекции можно проиллюстрировать на следующем примере. Предположим, что вы устраиваете вечеринку и приглашаете на нее довольно много друзей. Как гостеприимный хозяин, вы встречаете всех своих друзей на входе в дом, но запоминаете только пришедших к вам девушек. В какой-то момент вы решаете подсчитать, сколько парней пришло к вам на вечеринку. Вы знаете количество пришедших к вам девушек, и вам кажется, что количество девушек и парней одинаково. Как вам быстро проверить это предположение? Ответ достаточно очевиден: попросить каждую девушку взять ровно одного парня за руку. Если в результате этой процедуры все множество гостей разбилось на пары, то ваше предположение окажется верным. Тем самым вы сильно упростили себе жизнь — вам не пришлось проделывать довольно утомительную работу по пересчету пришедших к вам парней, вы просто воспользовались для их подсчета результатом уже проделанной работы по пересчету пришедших к вам девушек.

Вернемся теперь к равенству (10). Заметим, что любое подмножество A множества X мы можем закодировать бинарной строкой $f(A)$ длины n, то есть строкой над алфавитом $\{0, 1\}$. Единице на i-м месте строки будет при этом отвечать ситуация, при которой $x_i \in A$, а нулю — вариант, при котором $x_i \notin A$. Так, если множество X имеет вид $\{x_1, x_2, x_3, x_4\}$, а подмножество A записывается в виде $A = \{x_2, x_4\}$, то соответствующая этому подмножеству строка длины 4 равна

$$f(A) = (0, 1, 0, 1).$$

Очевидно, что построенное отображение f взаимно-однозначно. Следовательно, количество подмножеств данного n-множества X совпадает с количеством бинарных строк длины n. А таких строк у нас имеется ровно 2^n штук — действительно, на любое из n мест мы двумя способами можем поставить либо ноль, либо единицу.

2.3.4. Полагая в (9) $x = -1$, $y = 1$, мы получаем еще одно важное тождество

$$\sum_{k=0}^{n} (-1)^k \binom{n}{k} = 0, \qquad n > 0. \tag{11}$$

Для того, чтобы понять комбинаторный смысл равенства (11), перепишем его в следующем виде:

$$\binom{n}{0} - \binom{n}{1} + \binom{n}{2} - \binom{n}{3} + \ldots = 0 \qquad \Longleftrightarrow \qquad \binom{n}{0} + \binom{n}{2} + \ldots = \binom{n}{1} + \binom{n}{3} + \ldots$$

В левой части последнего равенства стоит количество всех подмножеств, в которых содержится четное количество элементов, а в правой — число подмножеств, содержащих нечетное количество элементов. Иными словами, следствием равенства (11) является тот факт, что количество четных подмножеств любого множества, отличного от $\varnothing$, равняется количеству его нечетных подмножеств.

Из (11) мы также можем заключить, что количество, например, всех нечетных подмножеств ровно в два раза меньше общего количества всех подмножеств. С учетом (10) это означает, что количество всех нечетных подмножеств равно 2^{n-1}.

2.3.5. Наконец, продифференцируем (9) по x:

$$n\,(x+y)^{n-1} = \sum_{k=1}^{n} k \binom{n}{k} x^{k-1}\, y^{n-k}. \tag{12}$$

Подставляя в это равенство $x = y = 1$, получим еще одно полезное равенство:

$$\sum_{k=1}^{n} k \binom{n}{k} = n\, 2^{n-1}. \tag{13}$$

2.4. Перейдем теперь к задачам, связанным с подсчетом k-сочетаний с повторениями.

2.4.1. Начнем с примера. Пусть множество X состоит из двух чисел 1 и 2. Выпишем все 3-сочетания с повторениями из 2-множества X:

$$\{1,1,1\}, \qquad \{1,1,2\}, \qquad \{1,2,2\}, \qquad \{2,2,2\}.$$

Как видно, таковых оказалось 4 штуки. Для подсчета количества $\left(\!\binom{n}{k}\!\right)$ таких мультимножеств в общем случае нам будет удобнее вначале конкретизировать n-множество X, а именно, взять в качестве X множество $[n] := \{1, 2, \ldots, n\}$ первых n натуральных чисел. Любое k-мультимножество такого множества можно записать, очевидно, в следующем виде:

$$1 \leqslant a_1 \leqslant a_2 \leqslant \ldots \leqslant a_k \leqslant n.$$

Например, 3-мультимножество $\{1,1,2\}$ над 2-множеством $X = \{1,2\}$ можно записать так:

$$1 \leqslant (a_1 = 1) \leqslant (a_2 = 1) \leqslant (a_3 = 2) \leqslant (n = 2).$$

Теперь превратим в этой цепочке все нестрогие неравенства в строгие. Для этого мы к a_2 прибавим единицу, к a_3 — двойку, к a_4 — тройку, и так далее. К последнему числу a_n мы, таким образом, добавим число $(k-1)$. В результате получим цепочку строгих неравенств вида

$$1 \leqslant a_1 < a_2 + 1 < a_3 + 2 < a_4 + 3 < \ldots < a_k + (k-1) \leqslant n + (k-1).$$

В нашем примере

$$1 \leqslant (a_1 = 1) < (a_2 + 1 = 2) < (a_3 + 2 = 4) \leqslant (n + (3-1) = 4).$$

Заметим, что в результате этой операции мы получили некоторое k-элементное подмножество *различных* чисел вида $a_i + (i-1)$ множества $\widetilde{X} = [n+k-1]$ всех чисел от единицы до $n+k-1$. Иными словами, мы сопоставили любому k-мультимножеству над множеством $X = [n]$ вполне определенное k-подмножество множества $\widetilde{X} = [n+k-1]$. Очевидно, что это сопоставление взаимно-однозначно. Но количество всех k-подмножеств данного множества мы знаем — оно равно $\binom{n+k-1}{k}$. Следовательно, этому числу равно, по принципу биекции, и количество $\left(\!\binom{n}{k}\!\right)$ всех k-мультимножеств над множеством $X = [n]$:

$$\left(\!\binom{n}{k}\!\right) = \binom{n+k-1}{k}.$$

Справедливость этого равенства для произвольного n-элементного множества X вновь следует из принципа биекции.

3 k-перестановки из n элементов. Урновые схемы и схемы раскладки предметов по ящикам.

3.1. Перейдем теперь к подсчету количества k-перестановок из n элементов.

3.1.1. Напомним, что k-перестановкой из n элементов называется *упорядоченный* набор

$$(a_1, a_2, \ldots, a_k)$$

элементов, в котором все a_i принадлежат одному и тому же n-элементному множеству X.

Элементы a_i в наборе $(a_1, a_2, \ldots, a_k)$ могут как повторяться, так и не повторяться. В первом случае говорят о k-перестановках с повторениями, во втором — о k-перестановках без повторений.

Номер паспорта — это типичный пример k-перестановки с повторениями над множеством из десяти цифр $X = \{0, 1, \ldots, 9\}$. Классическим примером 3-перестановки без повторений является упорядоченный список спортсменов, занявших призовые места в любых спортивных соревнованиях.

3.1.2. В литературе встречается довольно много альтернативных названий для данного объекта. Именно, упорядоченный набор $(a_1, a_2, \ldots, a_k)$, $a_i \in X$, также иногда называется

- k-размещением из n элементов;
- кортежем из k элементов множества X;
- упорядоченной k-выборкой из n элементов;
- k-мерным вектором над множеством X;
- k-элементным словом над n-элементным алфавитом.

3.1.3. Сосчитаем количество k-перестановок с повторениями.

Утверждение 3.1. *Количество k-перестановок с повторениями из n элементов равно n^k.*

Для доказательства можно либо просто сослаться на правило произведения, либо рассмотреть $(a_1, a_2, \ldots, a_k)$, $a_i \in X$ как слово из k элементов над алфавитом из $n = |X|$ букв. На первое место в слове мы можем поставить любую из n букв, на второе — также любую из n букв и так далее. Всего же получаем n^k вариантов записать данное слово.

3.1.4. Перейдем теперь к подсчету количества перестановок без повторений.

Утверждение 3.2. *Количество $P(n, k)$ k-перестановок из n элементов без повторений равно*

$$P(n, k) = n \cdot (n-1) \cdot \ldots \cdot (n-k+1) =: (n)_k.$$

Доказательство очевидно — на первое место в строке длины k я могу поставить любой из n элементов, на второе — любой из оставшихся $(n-1)$ элементов и так далее.

3.1.5. В частном случае $k = n$ k-перестановки из n элементов без повторений называются просто перестановками n-элементного множества X. Их количество равно

$$P_n \equiv P(n) = n!, \qquad P(0) = 0! = 1.$$

3.1.6. Любую k-перестановку из n элементов без повторений можно рассматривать и как упорядоченное k-подмножество n-множества. Количество таких подмножеств мы можем сосчитать следующим образом: мы можем $\binom{n}{k}$ способами выбрать k-подмножество n-элементного множества, а затем $k!$ способами его упорядочить. Действительно, на первое место мы можем поставить любой из k элементов подмножества, на второе — любой из оставшихся $k-1$ элементов и так далее. Таким образом, мы получаем некоторое новое выражение для количества всех упорядоченных k-элементных подмножеств n-множества, то есть количества k-перестановок из n элементов без повторений, равно

$$(n)_k = k! \cdot \binom{n}{k} \qquad \Longrightarrow \qquad \binom{n}{k} = \frac{(n)_k}{k!}.$$

Последняя формула часто используется как некомбинаторное определение биномиальных коэффициентов $\binom{n}{k}$ в случае, когда $k \in \mathbb{Z}$, а n принадлежит $\mathbb{Z}$, $\mathbb{R}$ или даже $\mathbb{C}$. Именно, по определению,

$$\binom{q}{k} := \begin{cases} \dfrac{q\,(q-1)\,\ldots\,(q-k+1)}{k!} =: \dfrac{(q)_k}{k!}, & \text{если } k \in \mathbb{N}, \\ 1, & \text{если } k = 0, \\ 0, & \text{если } k < 0, \end{cases} \qquad \forall\ q \in \mathbb{C}.$$

Например,

$$\binom{-1}{3} = \frac{(-1)\cdot(-2)\cdot(-3)}{3\cdot 2\cdot 1} = -1.$$

3.1.7. Функцию $(q)_k$ часто также обозначают через $q^{\underline{k}}$ и называют *убывающей факториальной степенью*. Наряду с убывающей можно ввести и так называемую *возрастающую факториальную степень*

$$q^{(k)} \equiv q^{\overline{k}} := q \cdot (q+1) \cdot \ldots \cdot (q+k-1).$$

В частности, с ее помощью получается удобное для вычислений выражение для количества $\left(\!\binom{n}{k}\!\right)$ сочетаний с повторениями:

$$\left(\!\binom{n}{k}\!\right) = \binom{n+k-1}{k} = \frac{n^{(k)}}{k!}. \tag{14}$$

3.2. Итак, мы получили простые соотношения для подсчета количества четырех основных объектов элементарной комбинаторики — k-сочетаний и k-перестановок из n элементов с повторениями и без повторений. Эти объекты встречаются в огромном количестве внешне не очень похожих друг на друга задач элементарной комбинаторики. Оказывается, однако, что большинство этих задач можно свести к одной из двух простейших схем — либо к так называемой урновой схеме, либо к схеме раскладки предметов по ящикам.

3.2.1. В урновой схеме имеется урна, в которой находятся n *различимых* предметов. Из урны последовательно вытаскивается k предметов. Задача состоит в подсчете количества различных способов выбора этих предметов, или, как еще говорят, в подсчете различных k-элементных выборок из n предметов, находящихся в урне.

На практике наиболее часто встречаются четыре модификации этой задачи, различающиеся способами формирования k-элементной выборки. Прежде всего, мы можем возвращать или не возвращать вытаскиваемые предметы обратно в урну. В первом случае говорят о *выборке с повторениями*, во втором — о *выборке без повторений.* Далее, в некоторых задачах нам

важен порядок вытаскиваемых предметов. В этом случае имеем так называемые *упорядоченные* выборки. В противном случае выборки называются *неупорядоченными*.

Нетрудно понять, что задачи о подсчете k-элементных выборок представляют собой, по сути, те же самые задачи о подсчете k-перестановок или k-сочетаний из n элементов. Действительно, любая неупорядоченная k-элементная выборка представляет собой либо k-элементное подмножество n-множества, либо k-мультимножество над n-элементным множеством в зависимости от того, возвращаем мы вытаскиваемые предметы обратно в урну или не возвращаем. Следовательно, количество таких неупорядоченных выборок совпадает с коэффициентами $\binom{n}{k}$ или $\left(\!\binom{n}{k}\!\right)$. Очевидно также, что любая упорядоченная k-элементная выборка есть просто некоторая k-перестановка n-элементного множества. Поэтому количество таких выборок равно n^k или $(n)_k$ в зависимости от того, говорим ли мы о выборке с повторениями или без повторений.

В результате получаем следующую таблицу решений задач, связанных с урновыми схемами:

Предметы на выходе	с возвращением	без возвращения
упорядоченные	n^k	$(n)_k$
неупорядоченные	$\left(\!\binom{n}{k}\!\right)$	$\binom{n}{k}$

3.2.2. Приведем несколько характерных примеров, достаточно естественно сводящихся к одной из описанных выше урновых схем.

Пример 3.3. Предположим, что у нас имеется некоторое общество, состоящее из двадцати членов. Сколькими способами можно выбрать президента, вице-президента, секретаря и казначея этого общества?

Решение. Очевидно, что любой способ выбора представляет собой упорядоченное 4-элементное подмножество 20-элементного множества. Следовательно, существует ровно $(20)_4$ различных способов выбора членов общества на эти должности.

Пример 3.4. Для того, чтобы открыть сейф, нужно набрать код из пяти символов с помощью вращающихся дисков. На каждом из этих дисков нанесено 12 символов, одинаковых для каждого из дисков. Сколько вариантов различных кодов существует?

Решение. Любой код представляет собой упорядоченную 5-элементную выборку с повторениями или, иначе, строку из пяти символов над алфавитом из 12 букв. Следовательно, имеется 12^5 вариантов различных кодов.

Пример 3.5. На почте продаются открытки десяти различных видов. Сколькими способами можно купить восемь открыток? А восемь открыток разных видов?

Решение. Понятно, что в первом случае любой набор из восьми открыток представляет собой неупорядоченную выборку с повторениями, а во втором — выборку без повторений из 10 элементов. Следовательно, в первом случае имеем $\left(\!\binom{10}{8}\!\right)$, а во втором — $\binom{10}{8}$ способов покупки восьми открыток.

3.2.3. Второй, не менее популярной в элементарной комбинаторике схемой, связанной с подсчетом количества k-перестановок и k-сочетаний, является схема раскладки предметов по ящикам. В этой схеме имеется n *различимых* ящиков, по которым нужно разложить k *различимых* или

неразличимых предметов. При этом мы можем накладывать определенные ограничения на количество предметов в каждом ящике.

Рассмотрим, к примеру, задачу о подсчете количества способов раскладки k различимых предметов по n различимым ящикам при условии, что в любой ящик можно класть любое количество предметов. Количество способов совершить эти действия равно, очевидно, n^k. Действительно, любой предмет мы можем положить в любой из n ящиков вне зависимости от того, куда мы положили оставшиеся предметы. Поэтому, согласно правилу произведения, это количество равно n^k. Иными словами, данная задача представляет собой переформулировку задачи о подсчете количества k-перестановок из n элементов с повторениями.

Теперь предположим, что в той же схеме мы не имеем права класть более одного предмета в один ящик. Тогда первый предмет мы можем поместить в любой из n ящиков, второй — в любой из оставшихся свободными $(n-1)$ ящиков и так далее. Всего же получаем $(n)_k$ способов раскладки. Следовательно, данная схема соответствует подсчету k-перестановок без повторений.

3.2.4. Пусть теперь у нас имеются n различимых ящиков и k неразличимых предметов. Тогда подсчет количества различных способов раскладки этих предметов по ящикам сводится к задаче о подсчете количества k-сочетаний из n элементов.

Действительно, в данной схеме в качестве n-элементного множества выступает множество, состоящее из n различимых ящиков. В случае, когда в каждый ящик можно класть не более одного предмета, мы, раскладывая предметы по ящикам, выделяем в этом множестве некоторое k-элементное подмножество. Следовательно, количество таких раскладок совпадает с количеством различных k-элементных подмножеств n-множества и равно $\binom{n}{k}$.

В случае же, когда никаких ограничений на количество предметов в ящике не накладывается, мы, раскладывая по ящикам k неразличимых предметов, задаем тем самым некоторое k-мультимножество n-множества X. Поэтому количество различных способов такой раскладки равно количеству $\left(\!\binom{n}{k}\!\right)$ k-сочетаний из n элементов с повторениями.

Подводя итоги, построим таблицу рассмотренных схем раскладок k предметов по n ящикам:

Предметы	Ящики	Произвольное количество предметов в ящике	Не более одного предмета в ящике
различимые	различимые	n^k	$(n)_k$
неразличимые	различимые	$\left(\!\binom{n}{k}\!\right)$	$\binom{n}{k}$

3.2.5. Проиллюстрируем некоторые характерные примеры задач, которые довольно естественно сводятся к схеме схем раскладки предметов по ящикам.

Пример 3.6. Сколькими способами можно разложить по двум *различимым* карманам (например, левому и правому) девять монет *различного* достоинства?

Решение. Рассматриваемый пример является типичной задачей, которая естественным образом сводится к схеме раскладки предметов по ящикам. Действительно, в роли ящиков здесь высту-

пают левый и правый карман, а в роли предметов — девять различных монет. Поэтому ответ в этой задаче — 2^9 способов.

Замечание 3.7. Рассмотренная задача, однако, не всегда решается верно: в качестве ответа иногда выдают 9^2 способов. Путаница, как правило, происходит потому, что эту задачу пытаются свести к урновой схеме, считая, что имеются урна, в которой расположены 9 различных предметов, а также 2 различимые позиции на выходе.

Для того, чтобы этой путаницы избежать, полезно сформулировать следующие основные отличия схемы раскладки по ящикам от соответствующей ей урновой схемы. Во-первых, в схеме раскладки предметов по ящикам предметы обратно не возвращаются, они остаются в ящике. Во-вторых, в этой схеме в любой ящик можно класть любое количество предметов. В аналогичной урновой схеме на любую позицию помещается ровно один предмет.

Приведем теперь два характерных примера, связанных с раскладкой неразличимых предметов по различимым ящикам.

Пример 3.8. У отца имеется 5 (неразличимых) апельсинов, которые он может раздать восьми своим сыновьям. Если его задача состоит в том, чтобы раздать их максимальному количеству сыновей, то он должен поставить дополнительное условие — любой из его сыновей не должен получить более одного апельсина. В этом случае количество способов, которыми он может раздать своим сыновьям эти пять апельсинов, равно, очевидно, $\binom{8}{5}$. Если же он раздает их по каким-то заслугам, и может, таким образом, любому сыну отдать любое количество апельсинов, то количество способов это сделать равно $\left(\!\binom{8}{5}\!\right) = \binom{12}{5}$.

Пример 3.9. В физике встречаются задачи, в которых имеются n различных уровней энергии и k неразличимых элементарных частиц. Если эти частицы — фермионы, то для них действует так называемый принцип запрета Паули, согласно которому на любом энергетическом уровне может находиться не более одной элементарной частицы. Как следствие, количество различных распределений k фермионов по n энергетическим уровням равно $\binom{n}{k}$. Наряду с фермионами существуют и частицы иного сорта — бозоны, для которых не существует ограничений на количество частиц, занимающих один и тот же уровень энергии. Для бозонов количество таких распределений равно, очевидно, $\left(\!\binom{n}{k}\!\right)$.

3.2.6. К задачам раскладки неразличимых предметов по различимым ящикам, связанным с подсчетом количества k-сочетаний, сводятся также задачи о так называемом *разбиении* натурального числа k на n слагаемых. Данная задача формулируется следующим образом: сколькими способами можно представить натуральное число k в виде суммы n слагаемых вида

$$a_1 + a_2 + \ldots + a_n = k$$

при условии, что порядок слагаемых важен, то есть при условии, что разбиения вида

$$1 + 3 + 3 + 3 = 10 \qquad \text{и} \qquad 3 + 1 + 3 + 3 = 10$$

считаются различными?

Если на числа a_i накладывается единственное условие вида $a_i \geqslant 0$, то количество разбиений равно количеству $\left(\!\binom{n}{k}\!\right)$ k-мультимножеств над n-элементным множеством. Действительно, в упорядоченной сумме $a_1 + a_2 + \ldots + a_n$ любой индекс i слагаемого a_i можно рассматривать как i-й ящик, в который мы складываем a_i единиц. Следовательно, эту задачу можно трактовать как задачу о раскладке k "неразличимых" единиц по n различимым ящикам.

Пример 3.10. Подсчитать количество разбиений числа $k = 4$ на два слагаемых:

$$a_1 + a_2 = 4, \qquad a_1, a_2 \geqslant 0.$$

Ответ: $\left(\!\binom{2}{4}\!\right) = \binom{5}{4} = 5$ разбиений: $0 + 4 = 1 + 3 = 2 + 2 = 3 + 1 = 4 + 0 = 4$.

К подсчету числа k-сочетаний из n элементов без повторений задача о разбиении числа k сводится в случае, когда на числа a_i накладываются следующие условия:

$$a_i = 0 \qquad \text{или} \qquad a_i = 1.$$

В этом случае индекс i также можно трактовать как i-й ящик; его можно выбрать (положив $a_i = 1$) или не выбрать (положив $a_i = 0$). Всего же нужно выбрать k таких ящиков. Это можно сделать $\binom{n}{k}$ способами.

Пример 3.11. Подсчитать количество разбиений числа 2 на три слагаемых при условии, что любое слагаемое может принимать значения 0 или 1.

Ответ: $\binom{3}{2} = 3$ разбиения: $1 + 1 + 0 = 1 + 0 + 1 = 0 + 1 + 1 = 2$.

3.2.7. Достаточно часто на практике встречаются ситуации, когда одну и ту же задачу можно свести и к урновой схеме, и к схеме раскладки предметов по ящикам.

Пример 3.12. В кондитерском магазине продаются пирожные трех разных видов. Сколькими различными способами можно купить семь пирожных?

Решение. Ответ в этой задаче, очевидно, равен $\left(\!\binom{3}{7}\!\right) = \binom{9}{7}$. Этот ответ можно, например, получить, представляя себе коробку с тремя отделениями, в каждое из которых кладется пирожное только одного вида; в этом случае мы сводим задачу к подсчету количества раскладок семи неразличимых предметов по трем различимым ящикам. Другой способ получить тот же ответ — это представлять себе урну, в которой находятся три разных пирожных, и считать количество способов выбора из урны семи пирожных с возвращениями любого выбранного пирожного обратно в урну. Наконец, можно вообще забыть о любых схемах, если понимать, что любые семь купленных пирожных трех различных видов представляют собой 7-мультимножество над 3-элементным множеством различных видов пирожных.

4 Подсчет количества отображений конечных множеств. Числа Стирлинга второго рода

4.1. Оказывается, задачи о раскладке различимых предметов по различимым же ящикам имеют и еще одну, чрезвычайно важную комбинаторную интерпретацию — они эквивалентны задачам о подсчете количества отображений конечных множеств.

4.1.1. Напомним определение произвольного отображения $f\colon X \to Y$.

Определение 4.1. Пусть X, Y — пара конечных множеств. Отображением f из X в Y называется правило, согласно которому любому элементу $x \in X$ ставится в соответствие единственный элемент $y \in Y$:

$$\forall\, x \in X \qquad \exists!\, y \in Y\colon \qquad y = f(x).$$

С комбинаторной точки зрения любое отображение f из n-элементного множества X в k-элементное множество Y можно рассматривать как некоторый вариант раскладки n различимых предметов по k различимым ящиками при отсутствии каких-либо ограничений на количество предметов в каждом ящике.

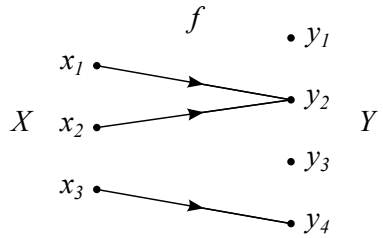


Рис. 6

Пример 4.2. Рассмотрим отображение f из трехэлементного множества X в четырехэлементное множество Y вида

$$f(x_1) = y_2, \qquad f(x_2) = y_2, \qquad f(x_3) = y_4$$

(смотри рис.6). Этому отображению отвечает раскладка трех различимых предметов по четырем различимым ящикам, при которой первые два предмета размещаются во втором ящике, а третий предмет — в четвертом ящике.

Как следствие, общее количество всех отображений n-элементного множества X в k-элементное множество Y равно k^n.

4.1.2. Напомним теперь определение *инъективного* отображения.

Определение 4.3. Отображение $f\colon X \to Y$ называется инъективным, если из условия

$$f(x_1) = f(x_2) \qquad \Longrightarrow \qquad x_1 = x_2.$$

Иными словами, отображение называется инъективным, если у любого элемента $y \in Y$ имеется не более одного прообраза, т.е. элемента $x \in X$, такого, что $y = f(x)$.

Понятно, что любому инъективному отображению $f\colon X \to Y$ отвечает такая раскладка n элементов множества X, при которой в каждом из k ящиков находится не более одного элемента. Как следствие, количество всевозможных инъективных отображений равно $(k)_n$.

4.1.3. Наконец, рассмотрим случай биективного и сюръективного отображений.

Определение 4.4. Отображение $f\colon X \to Y$ называется *биективным,* если

$$\forall\, y \in Y \qquad \exists!\, x \in X\colon \qquad y = f(x).$$

Количество таких отображений равно, очевидно, $n!$, где $n = |X| = |Y|$.

Определение 4.5. Отображение $f\colon X \to Y$ называется сюръективным, если

$$\forall\, y \in Y \qquad \exists\, x \in X\colon \qquad y = f(x).$$

Другими словами, отображение f сюръективно, если у любого элемента $y \in Y$ найдется хотя бы один прообраз $x \in X$, то есть такой x, что $f(x) = y$.

Комбинаторная интерпретация сюръективного отображения такова: это есть некоторая раскладка n различимых предметов по k различимым ящикам при условии, что в каждом ящике находится хотя бы один предмет. Количество таких раскладок при $k > n$ равно, очевидно, нулю. Задача следующего пункта данного параграфа — сосчитать количество этих раскладок для случая $0 \leqslant k \leqslant n$.

4.2. Обозначим через $\widehat{S}(n,k)$ количество всех сюръективных отображений n-элементного множества X в k-элементное множество Y. Сосчитаем $\widehat{S}(n,k)$ для случая $n \geqslant 0$, $0 \leqslant k \leqslant n$.

4.2.1. Рассмотрим множество *всех* отображений из n-множества X в k-множество Y. Как мы знаем, количество таких отображений равно k^n. Наша задача состоит в том, чтобы подсчитать это количество по-другому, выразив k^n через числа $\widehat{S}(n,k)$.

4.2.2. Заметим, что *любое* отображение $f\colon X \to Y$ можно рассматривать как *сюръективное* отображение множества X на множество

$$\operatorname{Im}(f) = \{y \in Y \mid \exists x\colon\ y = f(x)\},$$

являющееся образом множества X при отображении f.

Так, для отображения f из примера 4.2 образ $\operatorname{Im}(f) = \{y_2, y_4\}$, а отображение $f\colon X \to Y$ является сюръективным отображением множества X на подмножество $\operatorname{Im}(f) \subset Y$.

4.2.3. Разобъем теперь все множество отображений $f\colon X \to Y$ на блоки, включив в i-й блок все отображения, образ $\operatorname{Im}(f)$ которых содержит ровно i элементов: $|\operatorname{Im}(f)| = i$, $i = 1, \ldots, k$. Все, что нам остается — это сосчитать количество элементов в каждом блоке, а затем воспользоваться правилом суммы для того, чтобы получить общее количество k^n всех отображений.

4.2.4. Заметим, что существует $\binom{k}{i}$ способов выбрать i-элементное подмножество k-множества Y. Для каждого из этих подмножеств имеется $\widehat{S}(n,i)$ различных сюръективных отображений из n-элементного множества X в выбранное i-элементное подмножество множества Y. Таким образом, по правилу произведения, общее количество элементов в i-м блоке равно

$$\binom{k}{i} \cdot \widehat{S}(n,i).$$

Тогда по правилу суммы можем записать, что

$$k^n = \sum_{i=0}^{k} \binom{k}{i} \cdot \widehat{S}(n,i). \tag{15}$$

При этом мы суммируем не от 1 до k, а от 0 до k, учитывая, что $\widehat{S}(n,0) = 0$ для всех $n > 0$.

Замечание 4.6. Формулу (15) полезно иногда записывать в виде

$$k^n = \sum_{i=0}^{n} \binom{k}{i} \cdot \widehat{S}(n,i). \tag{16}$$

Несложно убедиться в том, что эта формула непосредственно следует из (15), а также в том, что она оказывается справедливой как для случая $n \geqslant k$, так и для случая $n < k$.

4.2.5. Мы выразили количество всех отображений n-элементного множества X в k-элементное множество Y через количество $\widehat{S}(n,i)$ сюръективных отображений. Нам же нужна обратная формула, выражающая количество $\widehat{S}(n,k)$ сюръективных отображений через число i^n всех отображений. Для ее получения воспользуемся так называемыми *формулами обращения.*

Утверждение 4.7. *Пусть* $(f_0, f_1, f_2, \ldots)$ *и* $(g_0, g_1, g_2, \ldots)$ *— две числовые последовательности, и пусть одна из них выражается через вторую по формулам*

$$f_k = \sum_{i=0}^{k} \binom{k}{i} g_i, \qquad k \geqslant 0. \tag{17}$$

Тогда справедлива следующая формула обращения:

$$g_k = \sum_{i=0}^{k} (-1)^{k-i} \binom{k}{i} f_i, \qquad k \geqslant 0. \tag{18}$$

С учетом этих формул обращения можно, считая n параметром, из соотношения (15) получить следующую явную формулу для вычисления чисел $\widehat{S}(n,k)$:

$$\widehat{S}(n,k) = \sum_{i=0}^{k} (-1)^{k-i} \binom{k}{i} \cdot i^n.$$

4.3. Задачи подсчета количества отображений n-элементного множества X в k-элементное множество Y имеют еще одну важную комбинаторную интерпретацию.

4.3.1. Начнем с простого примера.

Пример 4.8. Для трехэлементного множества $X = \{x_1, x_2, x_3\}$ и двухэлементного множества $Y = \{y_1, y_2\}$ имеется, как мы знаем, $2^3 = 8$ различных отображений множества X в множество Y. Запишем все эти отображения как упорядоченные пары подмножеств множества X:

$$(\{x_1, x_2, x_3\}, \varnothing), \qquad (\{x_1, x_2\}, \{x_3\}), \qquad (\{x_1, x_3\}, \{x_2\}), \qquad (\{x_2, x_3\}, \{x_1\}),$$

$$(\{x_1\}, \{x_2, x_3\}), \qquad (\{x_2\}, \{x_1, x_3\}), \qquad (\{x_3\}, \{x_1, x_2\}), \qquad (\varnothing, \{x_1, x_2, x_3\}).$$

Видно, что записанное в таком виде решение представляет собой не что иное, как список всех возможных *разделений* множества X, то есть упорядоченных разбиений X на два блока, один из которых может быть и пустым.

4.3.2. Очевидно, что данный результат справедлив и в общем случае. Именно, любое отображение $f\colon X \to Y$ задает нам некоторое разделение множества X, то есть разбиение этого множества на k упорядоченных блоков, часть из которых могут быть пустыми. Как следствие, количество таких разделений совпадает с количеством всех отображений f и равно k^n.

Аналогичные рассуждения показывают, что любое сюръективное отображение $f\colon X \to Y$ задает нам некоторое *упорядоченное разбиение* множества X на блоки. Поэтому количество всех упорядоченных разбиений n-элементного множества X на k блоков равно числу $\widehat{S}(n,k)$.

4.3.3. Рассмотрим теперь некоторый специальный вид k-разделений множества X, а именно, такие k-разделения, в которых в первом блоке содержится a_1 элемент, во втором блоке — a_2

элемента, в k-м блоке — a_k элементов. Очевидно, что при этом общая сумма всех элементов должна быть равна мощности $|X| = n$:

$$a_1 + a_2 + \ldots + a_k = n, \qquad a_i \geqslant 0.$$

Утверждение 4.9. *Количество всех таких k-разделений n-множества X равно*

$$P(n; a_1, a_2, \ldots, a_k) := \binom{n}{a_1} \cdot \binom{n-a_1}{a_2} \cdot \ldots \cdot \binom{n-a_1-a_2-\ldots-a_{k-1}}{a_k} = \frac{n!}{a_1! \cdot a_2! \cdot \ldots \cdot a_k!}. \quad (19)$$

Доказательство. Действительно, из любого n-элементного множества X мы $\binom{n}{a_1}$ способами можем выбрать a_1 элементов и положить их в первый ящик (отнести к первому блоку разбиения). Затем для каждого такого выбора мы $\binom{n-a_1}{a_2}$ способами можем из оставшегося ($n - a_1$)-элементного множества выбрать a_2 элементов и положить их во второй ящик (отнести ко второму блоку разбиения), и так далее. Формула (19), описывающая общее количество способов совершить все эти действия, следует теперь из правила произведения. □

Следствие 4.10. *Общее количество k^n всех k-разделений n-множества X выражается через числа $P(n; a_1, a_2, \ldots, a_k)$ по формуле*

$$k^n = \sum_{\substack{a_1+a_2+\ldots+a_k=n \\ a_i \geqslant 0}} P(n; a_1, a_2, \ldots, a_k) = \sum_{\substack{a_1+a_2+\ldots+a_k=n \\ a_i \geqslant 0}} \frac{n!}{a_1! \cdot a_2! \cdot \ldots \cdot a_k!}. \quad (20)$$

Замечание 4.11. Если в условии рассматриваемой задачи заменить нестрогие неравенства $a_i \geqslant 0$ на строгие, то есть на неравенства $a_i > 0$, то вместо разделения мы получим упорядоченное разбиение специального вида. Количество таких упорядоченных разбиений также описывается формулой (19), а вместо формулы (20) получается не менее полезное соотношение вида

$$\widehat{S}(n, k) = \sum_{\substack{a_1+a_2+\ldots+a_k=n \\ a_i > 0}} \frac{n!}{a_1! \cdot a_2! \cdot \ldots \cdot a_k!}. \quad (21)$$

4.4. Числа $P(n; a_1, a_2, \ldots, a_k)$ имеют и еще один важный комбинаторный смысл. Именно, они перечисляют так называемые *перестановки n-множества X с повторениями.*

4.4.1. Рассмотрим в качестве элементарного примера следующую задачу: на полке имеются 15 различных книг по математике, 16 по информатике и 12 по физике; каково количество способов перестановки этих книг на полке? Ответ очевиден: $(15 + 16 + 12)! = 43!$ способов.

Предположим теперь, что мы перестали различать книги, посвященные одному и тому же предмету. В этом случае количество различных способов перестановки таких книг уменьшится. Обозначим это количество через λ_n. Так как существует 15! способов упорядочить книги по математике, 16! – по информатике и 12! – по физике, то по правилу произведения мы можем записать, что

$$43! = \lambda_n \cdot 16! \cdot 15! \cdot 12! \qquad \Longrightarrow \qquad \lambda_n = \frac{43!}{16! \cdot 15! \cdot 12!} = P(43; 15, 16, 12).$$

4.4.2. Аналогичные рассуждения справедливы и в общем случае. Именно, пусть среди n переставляемых предметов имеется a_1 неразличимых предметов первого сорта, a_2 неразличимых

предметов второго сорта и так далее, причем $a_1 + a_2 + \ldots + a_k = n$. Тогда для количества перестановок таких предметов с повторениями получаем уже знакомую нам формулу

$$\frac{n!}{a_1! \cdot a_2! \cdot \ldots \cdot a_k!} = P(n; a_1, a_2, \ldots, a_k).$$

4.4.3. В частном случае перестановки n предметов двух различных сортов получаем

$$P(n; k, n-k) = \frac{n!}{k! \cdot (n-k)!} = \binom{n}{k}.$$

Иными словами, количество таких перестановок совпадает с количеством различных k-элементных подмножеств n-элементного множества. Для комбинаторного доказательства данного факта можно воспользоваться рассуждениями, которые мы проводили при подсчете количества всех подмножеств данного множества. Напомним, что там мы любое подмножество кодировали упорядоченной битовой строкой длины n, состоящей из k единиц и $(n-k)$ нулей. Осталось заметить, что любая такая строка представляет собой некоторую перестановку k элементов первого сорта (единиц) и $(n-k)$ элементов второго сорта (нулей).

4.4.4. В заключение отметим еще одну полезную биекцию, позволяющую несколько по-другому сосчитать количество k-мультимножеств n-элементного множества. Мы знаем, что любому k-мультимножеству над n-множеством отвечает некоторая раскладка k неразличимых предметов по n различимым ящикам. В свою очередь, любую такую раскладку можно рассматривать как упорядоченный набор, состоящий из k неразличимых предметов одного сорта (например, k неразличимых шаров), и $(n-1)$-го предмета второго сорта ($(n-1)$-й неразличимой перегородки между этими шарами). Как следствие, количество всех k-мультимножеств

$$\left(\!\binom{n}{k}\!\right) = P(k+n-1; k, n-1) = \frac{(n+k-1)!}{(n-1)! \cdot k!} = \binom{n+k-1}{k}.$$

4.5. Вернемся к числам $\widehat{S}(n,k)$, описывающим, в частности, количество всех упорядоченных разбиений n-множества на k блоков. Обозначим теперь через $S(n,k)$ количество обычных, неупорядоченных разбиений n-множества на k блоков. Так как для любого неупорядоченного разбиения существует $k!$ способов упорядочить k его блоков, то

$$\widehat{S}(n,k) = k!\, S(n,k).$$

Отсюда с учетом (15) и (21) получаем следующие две явные формулы, позволяющие вычислять числа $S(n,k)$:

$$S(n,k) = \frac{1}{k!}\sum_{i=0}^{k}(-1)^i\binom{k}{i}(k-i)^n = \frac{1}{k!}\sum_{\substack{a_1+a_2+\ldots+a_k=n \\ a_i>0}} \frac{n!}{a_1! \cdot a_2! \cdot \ldots \cdot a_k!}. \tag{22}$$

Числа $S(n,k)$ называются *числами Стирлинга второго рода* и встречаются в большом количестве комбинаторных приложений. Исследуем эти числа поподробнее.

4.5.1. Для практического расчета чисел $S(n,k)$ удобно использовать рекуррентные соотношения.

Утверждение 4.12. *Числа Стирлинга второго рода удовлетворяют следующим рекуррентным соотношениям:*

$$S(n,k) = S(n-1,k-1) + k \cdot S(n-1,k), \qquad k = 1,\dots,n; \tag{23}$$

$$S(0,0) = 1; \qquad S(n,0) = 0 \quad \forall\ n > 0; \qquad S(n,k) = 0 \quad \forall\ k > n.$$

Доказательство. Граничные условия $S(n,0) = 0$ для всех $n > 0$ и $S(n,k) = 0$ для всех $k > n$ очевидны — n-элементное множество в случае $n > 0$ нельзя разбить на 0 блоков, а также на k блоков в случае, когда $k > n$. Равенство $S(0,0) = 1$ введено просто для удобства.

Докажем теперь соотношение (23). Разобьем для этого множество всех k-разбиений на два блока. К первому блоку Σ_1 отнесем все разбиения, содержащие одноэлементное подмножество $\{x_1\}$. Ко второму блоку Σ_2 отнесем все оставшиеся k-разбиения, т.е. разбиения, в которых элемент x_1 входит в подмножества, содержащие как минимум два элемента.

Рассмотрим, к примеру, все 2-разбиения множества $X = \{x_1, x_2, x_3, x_4\}$:

$$\{\{x_1,x_2,x_3\},\{x_4\}\}, \quad \{\{x_1,x_2,x_4\},\{x_3\}\}, \quad \{\{x_1,x_3,x_4\},\{x_2\}\}, \quad \{\{x_2,x_3,x_4\},\{x_1\}\},$$

$$\{\{x_1,x_2\},\{x_3,x_4\}\}, \quad \{\{x_1,x_3\},\{x_2,x_4\}\}, \quad \{\{x_1,x_4\},\{x_2,x_3\}\}.$$

Для этого примера к первому блоку относится единственное разбиение вида $\{\{x_2,x_3,x_4\},\{x_1\}\}$. Остальные шесть разбиений относятся в данном случае ко второму блоку.

Довольно очевидно, что количество элементов в первом блоке равно количеству $S(n-1,k-1)$ всех $(k-1)$-разбиений оставшегося $(n-1)$-элементного множества. В примере это число равно $S(3,1) = 1$ — любое множество можно только одним способом разбить на один блок.

Для подсчета количества элементов во втором блоке заметим, что, по сути дела, это число равно количеству всевозможных разбиений $(n-1)$-элементного множества на k непустых подмножеств с поочередным добавлением элемента x_1 в каждое из этих подмножеств. Действительно, для разобранного выше примера имеется три разбиения трехэлементного множества $\{x_2,x_3,x_4\}$ на два непустых подмножества, а именно, разбиения вида

$$\{\{x_2\},\{x_3,x_4\}\}, \quad \{\{x_3\},\{x_2,x_4\}\}, \quad \{\{x_4\},\{x_2,x_3\}\}.$$

Добавляя к каждому из этих подмножеств элемент x_1, получим $2 \cdot 3 = 6$ выписанных выше разбиений четырехэлементного множества X на блоки требуемого вида. Количество таких разбиений в общем случае равно, очевидно, $k \cdot S(n-1,k)$. □

4.5.2. Как и биномиальные коэффициенты, числа Стирлинга второго рода удобно представлять в виде треугольного массива на плоскости (n,k). На рис.7 показаны первые несколько строк такого массива, вычисленных с помощью рекуррентных соотношений (23). Такой рисунок дает нам еще одну комбинаторную интерпретацию чисел $S(n,k)$ — это есть количество *взвешенных* путей, идущих из начала координат в точку с координатами (n,k). Любой участок такого пути, имеющий вес i, можно рассматривать как набор из i путей, соединяющих соответствующие точки на плоскости.

4.5.3. Числа Стирлинга второго рода встречается в очень большом количестве самых разнообразных задач. В качестве примера вернемся к формуле (16) и перепишем ее через числа Стирлинга второго рода:

$$k^n = \sum_{i=0}^{n} \binom{k}{i} \cdot i! \cdot S(n,i) = \sum_{i=0}^{n} (k)_i \cdot S(n,i). \tag{24}$$

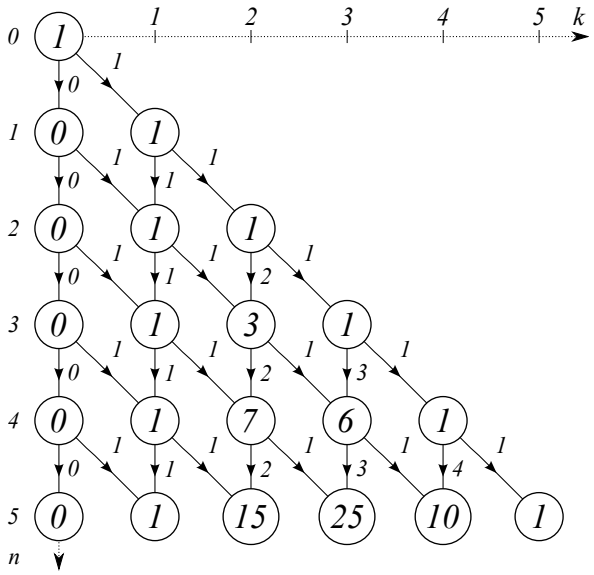


Рис. 7: Графическое представление чисел $S(n,k)$

Оказывается, эта формула справедлива для любых вещественных и даже комплексных значений k. Именно, справедливо равенство

$$x^n = \sum_{i=0}^{n} (x)_i \cdot S(n,i).$$

Эта формула активно используется в теории конечных операторов — она позволяет перейти от базиса x^n к базису $(x)_n$.

4.5.4. С точки зрения схемы раскладки предметов по ящикам числа Стирлинга второго рода описывают количество способов разложить n различимых предметов по k неразличимым ящикам так, чтобы в любом ящике содержался хотя бы один предмет. Сосчитаем, зная $S(n,k)$, количество $B(n,k)$ различных раскладок n различимых предметов по k неразличимым ящикам в случае, когда ограничения на количество предметов в любом ящике отсутствуют.

Для этого вновь разобьем множество всех таких раскладок на блоки, поместив в i-й блок все раскладки, в которых занято ровно i ящиков. В случае различимых ящиков в процессе аналогичного разбиения множества на блоки мы в i-м блоке имели $\binom{k}{i} \cdot \widehat{S}(n,i)$ элементов. В случае неразличимых ящиков мы i из k ящиков выбираем ровно одним способом, а затем $S(n,i)$ способами заполняем эти ящики n предметами. Используя правило суммы, получаем отсюда следующее выражение для чисел $B(n,k)$:

$$B(n,k) = \sum_{i=1}^{k} S(n,i).$$

4.5.5. Числа $B(n,k)$ в случае $n=k$ называются числами Белла $B(n)$. Эти числа перечисляют количество *всех* возможных разбиений n-элементного множества X.

Заметим, что любое разбиение множества X можно получить, введя на этом множестве некоторое отношение эквивалентности. Как следствие, количество всех возможных отношений эквивалентности на n-элементном множестве X описывается числами Белла B_n.

4.5.6. Покажем, что для чисел Белла $B(n)$ справедливо следующее рекуррентное соотношение:

$$B_{n+1} = \sum_{k=0}^{n} \binom{n}{k} B_k. \tag{25}$$

Действительно, рассмотрим блок разбиения $(n+1)$-элементного множества, содержащий число $n+1$. Все возможные разбиения мы можем разбить на $n+1$ группу (блок) в зависимости от того, сколько чисел содержится вместе с $n+1$. Предоположим, что вместе с $n+1$ находятся i чисел, $i = 0, \ldots, n$. Эти числа мы можем выбрать $\binom{n}{i} = \binom{n}{n-i}$ способами. Оставшиеся $k = n - i$ чисел, $k = 0, \ldots, n$, мы можем $B(k)$ способами разбить на блоки. Суммируя по всем k, мы и получим формулу (25).

4.6. Подведем итоги, построив таблицу различных схем раскладок n предметов по k ящикам:

Элементы множества X (предметы)	Элементы множества Y (ящики)	Произвольное количество предметов в ящике	Не более 1 предмета в ящике	Как минимум 1 предмет в ящике
различимые	различимые	k^n	$(k)_n$	$\widehat{S}(n,k)$
неразличимые	различимые	$\left(\!\binom{k}{n}\!\right)$	$\binom{k}{n}$	$\binom{n-1}{k-1}$
различимые	неразличимые	$B(n,k)$	$\begin{matrix} 0, & n > k \\ 1, & n \leqslant k \end{matrix}$	$S(n,k)$

5 Рекуррентные соотношения

5.1. Мы уже несколько раз получали решения комбинаторных задач, записанные в виде тех или иных рекуррентных соотношений. Настало время поговорить о них немного поподробнее.

5.1.1. Начнем с простого примера.

Пример 5.1. Популяция лягушек в озере увеличивается в четыре раза каждый год. В последний день каждого года 100 лягушек отлавливают и переправляют на другие озера. Предполагая, что в начале первого года в озере было 50 лягушек, найти количество лягушек в начале любого последующего года.

Решение. Обозначим через a_n количество лягушек в начале $(n+1)$-го года. По условию задачи, $a_0 = 50$. Тогда, очевидно,

$$a_1 = 4 \cdot 50 - 100 = 100, \qquad a_2 = 4 \cdot 100 - 100 = 300,$$

а в общем случае

$$a_{n+1} = 4 \cdot a_n - 100, \qquad n = 0, 1, 2, \ldots$$

Полученное равенство является простейшим примером *рекуррентного соотношения.*

5.1.2. Перейдем теперь к формальным определениям.

Определение 5.2. Пусть $a_0, a_1, a_2, \ldots$ — произвольная числовая последовательность. Если для любого $n \geqslant 0$ число a_{n+m} является некоторой функцией от m предыдущих членов последовательности, т.е.

$$a_{n+m} = f_n(a_n, a_{n+1}, \ldots, a_{n+m-1}), \tag{26}$$

то такая последовательность называется рекуррентной последовательностью, а соотношение (26) — рекуррентным соотношением m-го порядка.

В частном случае линейной функции f имеем так называемое *линейное рекуррентное соотношение*

$$a_{n+m} = b_1(n)\, a_{n+m-1} + b_2(n)\, a_{n+m-2} + \ldots + b_{m-1}(n)\, a_{n+1} + b_m(n)\, a_n + u(n). \tag{27}$$

В случае $u(n) = 0$ оно называется *однородным,* в противном случае — *неоднородным.*

Самый простой случай рекуррентного соотношения (27) — это *линейное однородное рекуррентное соотношение с постоянными коэффициентами*

$$a_{n+m} = b_1\, a_{n+m-1} + b_2\, a_{n+m-2} + \ldots + b_{m-1}\, a_{n+1} + b_m\, a_n. \tag{28}$$

Очевидно, что для однозначного определения всех a_n необходимо наряду с самим рекуррентным соотношением (26) задать и первые m членов $a_0, a_1, \ldots, a_{m-1}$ данной последовательности, то есть, как говорят, *задать начальные условия* для рекуррентного соотношения.

5.1.3. Итак, наличие рекуррентного соотношения и начальных условий позволяет нам последовательно, шаг за шагом, определить любое наперед заданное количество n членов рекуррентной числовой последовательности. Зачастую это все, что нам требуется от задачи. Иными словами, получение рекуррентного соотношения для искомой числовой последовательности a_n является вполне приемлемым, а зачастую — и наиболее удобным с вычислительной точки зрения ответом для поставленной комбинаторной задачи.

Однако иногда нам полезно иметь явную формулу, которая для любого n позволяет вычислять значение a_n. Такую формулу называют *решением* рассматриваемого рекуррентного соотношения. Мы в данном параграфе покажем, как строить такие решения для случая линейного однородного рекуррентного соотношения с постоянными коэффициентами (28).

5.2. Прежде чем рассматривать общий случай уравнения (28), рассмотрим наиболее простой его вариант — линейное однородное рекуррентное соотношение первого порядка

$$a_{n+1} = b_1 \cdot a_n, \qquad n = 0, 1, 2, \ldots; \qquad a_0 \text{ — заданное число.} \tag{29}$$

5.2.1. Решение этого уравнения построить легко. Действительно,

$$a_1 = b_1 \cdot a_0; \qquad a_2 = b_1 \cdot a_1 = b_1^2 \cdot a_0; \qquad \ldots \qquad a_n = b_1^n \cdot a_0.$$

5.2.2. Предположим теперь, что нам кто-то сразу подсказал вид решения, а именно, что решение нашего уравнения (29) степенным образом зависит от n:

$$a_n = r^n \qquad \text{для некоторого } r.$$

Воспользовавшись этой подсказкой, подставим это выражение в исходное рекуррентное соотношение (29). В результате получается равенство вида

$$r^{n+1} = b_1 \cdot r^n,$$

из которого сразу же следует, что $r = b_1$; при этом a_n оказывается равным b_1^n. Это означает, что при $n = 0$ число $a_0 = b_1^0 = 1$. Иными словами, $a_n = b_1^n$ есть решение исходной задачи (29) в частном случае $a_0 = 1$. Поэтому такое решение называется *частным решением* уравнения (29).

5.2.3. Теперь попытаемся, зная частное решение, построить общее решение задачи (29). Для этого заметим, что в силу однородности уравнения (29) любое его частное решение, умноженное на произвольную постоянную c, по-прежнему этому уравнению удовлетворяет:

$$c \cdot b_1^{n+1} \equiv c \cdot b_1 \cdot b_1^n.$$

Решение же вида $c \cdot b_1^n$ позволяет удовлетворить любому начальному условию. Действительно, подставляя его в начальное условие для уравнения (29), получим:

$$c \cdot b_1^0 = a_0 \qquad \Longrightarrow \qquad c = a_0 \qquad \Longrightarrow \qquad a_n = a_0 \cdot b_1^n.$$

По этой причине решение вида $c \cdot b_1^n$ называется *общим решением* уравнения (29).

5.3. Подведем предварительные итоги. Так как исходное уравнение (29) было очень простым, то нам удалось сразу построить его решение и выяснить, что оно степенным образом зависит от параметра n. Затем мы заметили, что если бы нам кто-то заранее подсказал степенной характер решения уравнения (29), то нам хватило бы этой информации для построения как общего, так и частного решения нашего рекуррентного уравнения.

Возникает вопрос: зачем же нам нужен для столь простого уравнения столь сложный алгоритм построения его решения? Оказывается, что этот алгоритм практически без изменений работает и для построения решения линейного однородного уравнения произвольного порядка.

5.3.1. Рассмотрим в качестве примера линейное однородное рекуррентное соотношение второго порядка

$$a_{n+2} = b_1 \cdot a_{n+1} + b_2 \cdot a_n, \qquad n = 0, 1, 2, \ldots, \qquad a_0,\, a_1 \text{ — заданные числа}, \tag{30}$$

и попытаемся применить к этому уравнению алгоритм, описанный в конце предыдущего пункта.

Для этого предположим, что частное решение уравнения (30) по прежнему степенным образом зависит от параметра n, то есть предположим, что существуют такие значения параметров a_0 и a_1, при которых $a_n = r^n$. Подставляя это выражение в рекуррентное соотношение, получим

$$r^{n+2} = b_1 \cdot r^{n+1} + b_2 \cdot r^n \qquad \Longrightarrow \qquad r^2 - b_1\, r - b_2 = 0, \tag{31}$$

т.е. квадратное уравнение на r. Любое его решение r_0 дает нам некоторое частное решение уравнения (30). По этой причине данное уравнение называется *характеристическим уравнением* для рекуррентного соотношения (30).

5.3.2. Предположим, что характеристическое уравнение (31) имеет два различных вещественных корня r_1 и r_2. Покажем, что в таком случае выражение

$$a_n = c_1\, r_1^n + c_2\, r_2^n, \tag{32}$$

где c_1 и c_2 — произвольные постоянные, является *общим решением* соотношения (30) в том смысле, что любое решение (30) с заданными начальными условиями в нем содержится.

Действительно, покажем вначале, что выражение (32) действительно удовлетворяет рекуррентному соотношению (30):

$$c_1\, r_1^{n+2} + c_2\, r_2^{n+2} = b_1\, c_1\, r_1^{n+1} + b_1\, c_2\, r_2^{n+1} + b_2\, c_1\, r_1^n + b_2\, c_2\, r_2^n \qquad \Longleftrightarrow$$

$$\iff \qquad c_1\, r_1^n\,(r_1^2 - b_1\, r_1 - b_2) + c_2\, r_2^n\,(r_2^2 - b_1\, r_2 - b_2) \equiv 0.$$

Покажем теперь, что это — действительно общее решение, т.е. что мы всегда можем подобрать константы c_1 и c_2 так, чтобы решение вида (32) удовлетворяло любым заданным начальным условиям. Для этого рассмотрим это выражение при $n = 0$ и $n = 1$:

$$\begin{aligned} a_0 &= c_1\, r_1^0 + c_2\, r_2^0 = c_1 + c_2, \\ a_1 &= c_1\, r_1^1 + c_2\, r_2^1 = c_1\, r_1 + c_2\, r_2. \end{aligned}$$

Эти выражения следует рассматривать как систему линейных уравнений для определения неизвестных постоянных c_1 и c_2. Определитель этой системы

$$\begin{vmatrix} 1 & 1 \\ r_1 & r_2 \end{vmatrix} = r_2 - r_1 \neq 0,$$

поэтому система всегда имеет единственное решение.

5.3.3. Пожалуй, самым известным и важным примером рекуррентного соотношения (30) является соотношение вида

$$F_{n+2} = F_{n+1} + F_n, \qquad n = 0, 1, 2, \ldots, \qquad F_0 = 0, \;\; F_1 = 1,$$

определяющее так называемые числа Фибоначчи F_n. Характеристическое уравнение для этого рекуррентного уравнения имеет вид

$$r^2 = r + 1 \qquad \Longrightarrow \qquad r_{1,2} = \frac{1 \pm \sqrt{5}}{2} \qquad \Longrightarrow \qquad F_n = c_1 \left(\frac{1+\sqrt{5}}{2}\right)^n + c_2 \left(\frac{1-\sqrt{5}}{2}\right)^n.$$

Константы c_1 и c_2 определим из начальных условий:

$$\begin{aligned} &F_0 = 0 = c_1 + c_2 \\ &F_1 = 1 = c_1\, \frac{1+\sqrt{5}}{2} + c_2\, \frac{1-\sqrt{5}}{2} = c_1 \left[\frac{1+\sqrt{5}-1+\sqrt{5}}{2}\right] = c_1\, \sqrt{5} \qquad \Longrightarrow \end{aligned}$$

$$\Longrightarrow \qquad c_1 = +\frac{1}{\sqrt{5}}, \qquad c_2 = -\frac{1}{\sqrt{5}}.$$

Таким образом, окончательно получаем следующее явное выражение для чисел Фибоначчи:

$$F_n = \frac{1}{\sqrt{5}} \left(\frac{1+\sqrt{5}}{2}\right)^n - \frac{1}{\sqrt{5}} \left(\frac{1-\sqrt{5}}{2}\right)^n. \tag{33}$$

5.3.4. Предположим теперь, что характеристическое уравнение (31) имеет ровно один кратный корень $r_1 = r_2 =: \rho$. Покажем, что в этом случае общее решение уравнения (30) имеет вид

$$a_n = c_1\, \rho^n + c_2\, n\, \rho^n.$$

Мы уже показали, что частное решение вида ρ^n удовлетворяет уравнению (30). С учетом линейности и однородности этого уравнения нам осталось показать, что этому уравнению удовлетворяет и частное решение вида $n\, \rho^n$. Подставляя в уравнение (30) выражение $n\, \rho^n$, получим

$$(n+2)\, \rho^{n+2} = b_1\, (n+1)\, \rho^{n+1} + b_2\, n\, \rho^n \qquad \Longrightarrow \qquad n\,(\rho^2 - b_1\, \rho - b_2) + \rho\,(2\,\rho - b_1) = 0.$$

Первое слагаемое в левой части этого выражения равно нулю в силу характеристического уравнения. Для того, чтобы понять, что равно нулю и последнее слагаемое, заметим, что в случае совпадающих корней $r_1 = r_2 = \rho$ имеем

$$\rho = \frac{b_1 \pm 0}{2} \qquad \Longrightarrow \qquad 2\rho - b_1 = 0.$$

Осталось показать, что такой вид решения может удовлетворить любым начальным условиям. Подставив начальные условия в выражение для a_n, получим следующую систему двух линейных алгебраических уравнений относительно параметров c_1 и c_2:

$$c_1 = a_0,$$

$$\rho\,(c_1 + c_2) = a_1.$$

Понятно, что такая система имеет решение при любых a_0, a_1 и $\rho \neq 0$. Случай же $\rho = 0$ тривиален — в этом случае $b_1 = b_2 = 0$, и потому $a_n = 0$ для любых n.

5.3.5. Наконец, давайте рассмотрим случай, когда уравнение (31) имеет два комплексно сопряжённых корня

$$r_1 = x + iy = \rho\, e^{i\,\vartheta}, \qquad r_2 = x - iy = \rho\, e^{-i\,\vartheta}, \qquad \rho, \vartheta \neq 0.$$

Покажем, что в таком случае общее решение линейного однородного рекуррентного соотношения с постоянными коэффициентами второго порядка (30) можно записать так:

$$a_n = \widetilde{c}_1\, \rho^n\, \cos(n\,\vartheta) + \widetilde{c}_2\, \rho^n\, \sin(n\,\vartheta). \tag{34}$$

Действительно, рассуждения, аналогичные проведенным для случая двух различных вещественных корней, показывают, что линейная комбинация частных решений вида

$$a_n = c_1\, r_1^n + c_2\, r_2^n$$

удовлетворяет рекуррентному соотношению (30) при любых c_1 и c_2. Рассмотрим теперь систему линейных уравнений

$$\begin{aligned} &a_0 = c_1 + c_2, \\ &a_1 = c_1\,\rho\, e^{i\,\vartheta} + c_2\,\rho\, e^{-i\,\vartheta} = \rho\,(c_1 + c_2)\cos\vartheta + \rho\, i\,(c_1 - c_2)\sin\vartheta. \end{aligned} \tag{35}$$

Определитель этой системы по-прежнему отличен от нуля, так что мы с помощью общего решения по-прежнему можем удовлетворить любым начальным условиям. Из (35) также следует, что коэффициенты $\widetilde{c}_1 = c_1 + c_2$ и $\widetilde{c}_2 = i\,(c_1 - c_2)$ являются вещественными числами. Действительно, первое уравнение (35) гарантирует нам, что сумма $c_1 + c_2$ является вещественным числом. Но тогда вещественным числом обязано быть и выражение $i\,(c_1 - c_2)$.

Заметим теперь, что линейную комбинацию частных решений мы в рассматриваемом случае можем переписать так:

$$a_n = c_1\, r_1^n + c_2\, r_2^n = c_1\,\rho^n\, e^{i\,\vartheta\, n} + c_2\,\rho^n\, e^{-i\,\vartheta\, n} =$$

$$= (c_1 + c_2)\,\rho^n\, \cos(n\,\vartheta) + i\,(c_1 - c_2)\,\rho^n\, \sin(n\,\vartheta).$$

Обозначив через $\widetilde{c}_1 = c_1 + c_2$ и $\widetilde{c}_2 = i\,(c_1 - c_2)$, мы получаем для a_n формулу (34).

5.3.6. Описанная выше техника решения линейных однородных рекуррентных соотношений с постоянными коэффициентами обобщается и на случай соотношений более высокого порядка. Именно, рассмотрим линейное однородное рекуррентное соотношение m-го порядка (28)

$$a_{n+m} = b_1 \cdot a_{n+m-1} + b_2 \cdot a_{n+m-2} + \ldots + b_m \cdot a_n, \qquad a_0, \ldots, a_{m-1} \text{ — заданы.}$$

Предположим, что частное решение этого соотношения имеет вид $a_n = r^n$. Подставляя это выражение в исходное рекуррентное соотношение, получаем для параметра r следующее характеристическое уравнение:

$$r^m = b_1 \cdot r^{m-1} + b_2 \cdot r^{m-2} + \ldots + b_m \cdot r.$$

Пусть $r_1, \ldots, r_k$ есть корни этого уравнения. Каждый такой корень дает свой вклад в общее решение рассматриваемого рекуррентного соотношения. В частности, если r_j есть корень кратности q_j, то вклад этого корня в общее решение задается выражением вида

$$c_{1,j} \cdot r_j^n + c_{2,j} \cdot n \cdot r_j^n + \ldots + c_{q_j,j} \cdot n^{q_j-1} \cdot r_j^n.$$

Так, если характеристическое уравнение для некоторого рекуррентного соотношения шестого порядка имеет вид

$$(r-1)^3 \cdot (r-2)^2 \cdot (r-3) = 0,$$

то общее решение такого соотношения записывается так:

$$a_n = c_1 + c_2 \cdot n + c_3 \cdot n^2 + c_4 \cdot 2^n + c_5 \cdot n \cdot 2^n + c_6 \cdot 3^n.$$

5.4. Рассмотренные ранее рекуррентные соотношения были линейными. Однако на практике довольно часто встречаются и более сложные, нелинейные рекуррентные соотношения. В качестве характерного примера рассмотрим задачи, связанные с так называемыми числами Каталана.

5.4.1. Начнем с характерного примера — с подсчета так называемых правильных скобочных последовательностей. Как известно, порядок вычислений в любом арифметическом выражении можно однозначно задать расстановкой скобок. Давайте возьмем какое-то достаточно произвольное арифметическое выражение, например,

$$(3-1) \cdot (4 + (15-9) \cdot (2+6)),$$

и сотрем в нем все числа и знаки арифметических операций. В результате такого действия мы получим последовательность открывающихся и закрывающихся скобок

$$()(()()),$$

представляющую собой так называемую правильную скобочную последовательность.

Определение 5.3. Правильная скобочная последовательность — это строка, состоящая из n открывающихся и n закрывающихся скобок, обладающая следующим свойством: при проходе вдоль этой структуры слева направо количество открывающихся скобок всегда больше или равно количеству закрывающихся скобок.

Перечислим все правильные скобочные последовательности с числом *пар* скобок $n = 1, 2, 3$:

$n = 1:$	$()$	– 1 последовательность;
$n = 2:$	$()(),\ (())$	– 2 последовательности;
$n = 3:$	$()()(),\ ()(()),\ (())(),\ (()()),\ ((()))$	– 5 последовательностей.

Числа C_n, описывающие количество таких последовательностей, называются *числами Каталана.* Как мы увидим несколько позднее, удобно по определению положить $C_0 = 1$.

Последовательность чисел Каталана

$$1, 1, 2, 5, 14, 42, 132, 429, 1430, \ldots$$

(последовательность $A000108$ в `OEIS` (`oeis.org`)) встречается в огромном количестве различных комбинаторных задач. К настоящему времени известно порядка 100 задач, в которых эти числа появляются. Приведем лишь несколько наиболее важных из них.

5.4.2. Рассмотрим вначале очень простую интерпретацию правильной скобочной последовательности — задачу об очереди в кассу. Предположим, что у кассы кинотеатра стоит очередь, состоящая из $2n$ человек. У половины из них имеется по 100 рублей, у второй половины — по 50 рублей. Билет в кино стоит 50 рублей. В начале продажи билетов касса кинотеатра пуста. Спрашивается, сколькими способами можно расставить людей в очереди правильно, т.е. так, чтобы никому не пришлось ждать у кассы сдачу.

Очевидно, что между этой задачей и задачей о количестве правильных скобочных последовательностей имеется биекция: любому человеку, имеющему 50 рублей, отвечает открывающаяся скобка в правильной скобочной последовательности, а человеку со 100 рублями — закрывающаяся скобка.

5.4.3. Задачу об очереди в кассу часто формулируют более формально, а именно, как задачу о подсчете количества различных *слов Дика* длины $2n$. Словом Дика называют строку длины $2n$ над алфавитом, состоящим из двух символов (например, X и Y), в которой количество символов X и Y совпадает, и в которой никакой начальный сегмент строки не содержит символов Y больше, чем символов X. В этой формулировке биекция с задачей о подсчете правильных скобочных последовательностей имеет, очевидно, вид

$$(\ \longleftrightarrow \ X, \qquad\qquad) \ \longleftrightarrow \ Y.$$

5.4.4. К задаче о подсчете правильных скобочных последовательностей сводится, очевидно, и задача о перечислении путей на плоскости, выходящих из начала координат, состоящих из отрезков $(1, 1)$ и $(1, -1)$, заканчивающихся в точке $(2n, 0)$ на оси абсцисс и нигде не пересекающих эту ось. Такие пути на плоскости называются *путями Дика* (рис 8). Биекция с правильными скобочными последовательностями здесь такова:

$$(\ \longleftrightarrow \ \text{вектор } (1, 1), \qquad\qquad) \ \longleftrightarrow \ \text{вектор } (1, -1).$$

5.4.5. Одной из основных дискретных структур, использующихся в теории алгоритмов, является плоское корневое бинарное дерево, иногда называющаяся просто бинарным или двоичным деревом. Неформально корневое дерево — это дерево, в котором любая вершина имеет ровно двух (возможно, пустых) потомков — левого и правого (рис. 9). Можно доказать, что количество таких деревьев на n вершинах описывается числами Каталана C_n.

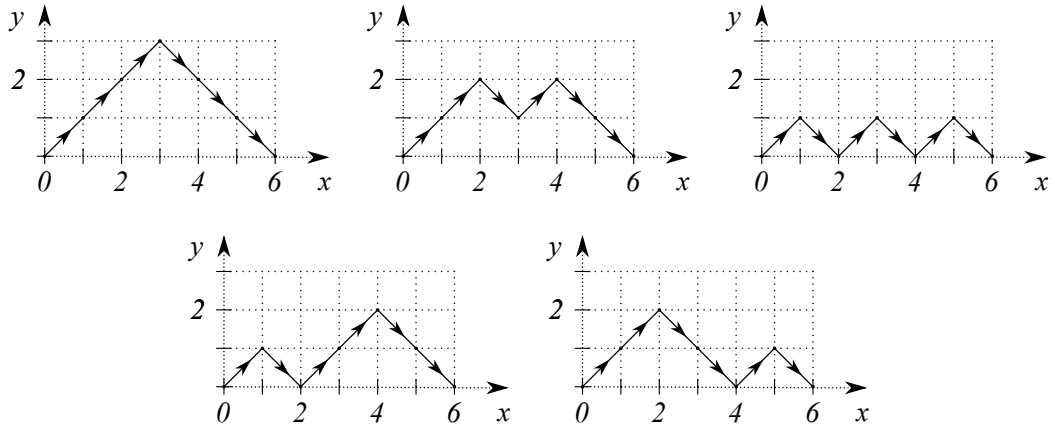


Рис. 8: Все пути Дика из трёх пар шагов.

Рис. 9: Все бинарные деревья на трёх вершинах.

5.4.6. Рассмотрим выражение вида

$$a_1 \cdot a_2 \cdot a_3 \cdot \ldots \cdot a_n \cdot a_{n+1}, \tag{36}$$

где элементы a_i принадлежат множеству S с введенной на нем неассоциативной бинарной операцией $' \cdot '$. Очевидно, что это выражение не имеет смысла до тех пор, пока мы не расставим скобки так, чтобы указать на последовательность проводимых операций. Несложно убедиться, что количество расстановки скобок в описанном выше выражении есть число Каталана C_n. Так, для случая $n = 3$ мы имеем следующие 5 различных вариантов расстановки скобок:

$$((a_1 \cdot a_2) \cdot a_3) \cdot a_4, \quad (a_1 \cdot (a_2 \cdot a_3)) \cdot a_4, \quad (a_1 \cdot a_2) \cdot (a_3 \cdot a_4), \quad a_1 \cdot ((a_2 \cdot a_3) \cdot a_4), \quad a_1 \cdot (a_2 \cdot (a_3 \cdot a_4)).$$

5.4.7. Еще одна важная комбинаторная интерпретация чисел Каталана появилась впервые в работах Леонарда Эйлера (и, кстати сказать, задолго до работ самого Эжена Каталана). Эйлер рассмотрел количество разбиений выпуклого $(n + 2)$-угольника с занумерованными (т.е. различимыми) вершинами на треугольники непересекающимися между собой диагоналями этого $(n + 2)$-угольника (рис. 10).

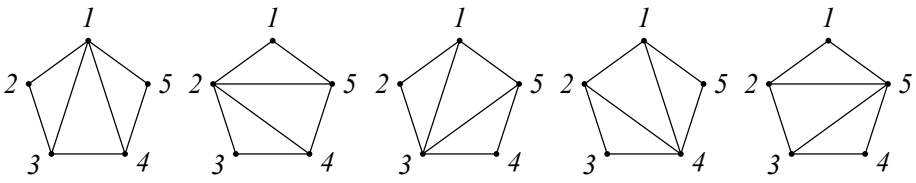


Рис. 10: Все триангуляции пятиугольника.

Можно доказать, что это количество описывается числами Каталана C_n, установив биекцию между всеми триангуляциями выпуклого $(n+2)$-угольника и плоскими корневыми бинарными деревьями, построенными на n вершинах.

5.4.8. Рассмотрим теперь все плоские корневые деревья (рис 11), построенные на $(n + 1)$-й вершине, $n = 0, 1, 2, \ldots$. Количество таких деревьев также равно числу Каталана C_n. Для того,

чтобы это понять, осуществим в любом таком дереве поиск в глубину. При движении вдоль некоторого ребра вниз, т.е. от корня, сопоставим этому ребру левую открывающуюся скобку. При движении в обратном направлении сопоставим этому же ребру закрывающуюся скобку. Тем самым мы устанавливаем биекцию между всеми такими деревьями и всеми правильными скобочными последовательностями.

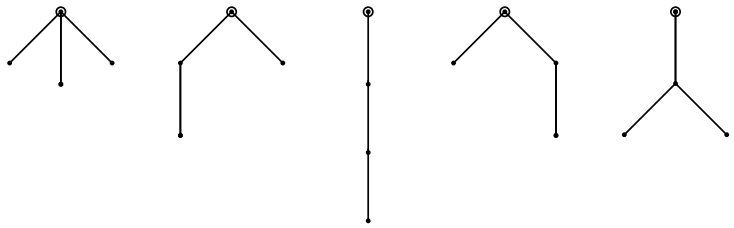


Рис. 11: Все плоские деревья на четырёх вершинах.

5.4.9. Наконец, перейдем к еще одному полезному понятию — стек-сортируемой последовательности. По определению, стек-сортируемая последовательность — это линейно упорядоченный набор из n чисел, который можно получить из линейно упорядоченной последовательности $(1, 2, \ldots, n)$ с помощью единственного стека.

В качестве примера рассмотрим последовательность чисел $(1, 2, 3, 4)$. Разместим ее справа от стека и будем сдвигать ее влево, помещая числа в стек и вынимая их из этого стека с помощью следующей последовательности действий:

– поместить число 1 в стек;
– поместить число 2 в стек;
– извлечь число 2 из стека;
– поместить число 3 в стек;
– поместить число 4 в стек;
– извлечь число 4 из стека;
– извлечь число 3 из стека;
– извлечь число 1 из стека.

В результате этих действий мы получим линейно упорядоченный набор чисел $(2, 4, 3, 1)$.

Возникает вопрос: сколько различных наборов чисел можно получить из последовательности $(1, 2, \ldots, n)$ с помощью подобных операций? Ответ, конечно же, вполне ожидаем — это количество равно числу Каталана C_n.

Для того, чтобы убедиться в этом, достаточно установить следующую биекцию между парой используемых в алгоритме операций и парой скобок:

$($ $\longleftrightarrow$ поместить число в стек, $\qquad\qquad$ $)$ $\longleftrightarrow$ извлечь число из стека.

В случае $n = 3$ имеется пять стек-сортируемых последовательностей:

$$(1, 2, 3), \qquad (3, 2, 1), \qquad (2, 1, 3), \qquad (1, 3, 2), \qquad (3, 1, 2).$$

Единственной перестановкой, которую нельзя перевести в последовательность $(1, 2, 3)$ с помощью стека, является перестановка вида $(2, 3, 1)$.

5.5. Теперь, после стольких примеров, пора научиться вычислять числа Каталана. Начнем с вывода рекуррентного соотношения для этих чисел. В качестве основного объекта мы выберем множество всех правильных скобочных последовательностей. Рекуррентное соотношение

для чисел C_n получим, воспользовавшись хорошо нам уже знакомым принципом разбиения множества на блоки и подсчетом количества элементов в каждом из блоков в отдельности.

5.5.1. Рассмотрим произвольную правильную скобочную последовательность. Для любой открывающейся скобки в такой последовательности можно ввести понятие парной ей закрывающейся скобки. Для этого будем идти от открывающейся скобки вправо и для каждой закрывающейся скобки будем проверять условие "количество закрывающихся скобок равно количеству открывающихся скобок". Первая закрывающаяся скобка, для которой это правило выполнится, и будет парной для нашей открывающейся скобки.

5.5.2. Разобъем теперь множество всех правильных скобочных последовательностей на блоки. Для этого возьмем крайнюю левую открывающуюся скобку в правильной скобочной последовательности и поместим в k-й блок все правильные скобочные последовательности, для которых парная ей закрывающаяся скобка стоит на $2k$-м месте.

Так, в случае $n = 3$ имеем разбиение множества, состоящего из пяти различных правильных скобочных последовательностей, на три блока:

$$\underbrace{()()(),\qquad ()(())}_{k=1};\qquad \underbrace{(())()}_{k=2};\qquad \underbrace{(()()),\qquad ((()))}_{k=3}.$$

5.5.3. Рассмотрим крайнюю левую открывающуюся и парную ей закрывающуюся скобки — выделенную пару скобок в нашей последовательности. Основное наблюдение здесь состоит в следующем: как внутри, так и снаружи указанной пары скобок стоят правильные скобочные последовательности.

Действительно, рассмотрим вначале последовательность скобок, находящуюся внутри выделенной пары скобок. Мы выбирали выделенную пару из того условия, что в подпоследовательности скобок, начинающейся с крайней левой открывающейся скобки и заканчивающейся парной ей закрывающейся скобкой, количество открывающихся скобок равно количеству закрывающихся скобок. Но, если мы крайнюю пару скобок удалим, то это условие для оставшейся скобочной подпоследовательности сохранится. Условие "количество открывающихся скобок больше или равно количеству закрывающихся скобок" в этой подпоследовательности также выполняется — в противном случае оно бы было нарушено и для всей последовательности скобок в целом.

Проверка того, что и правая подпоследовательность является правильной, проводится с помощью аналогичных рассуждений.

5.5.4. Подсчитаем теперь количество элементов в k-м блоке. Для этого заметим, что количество способов построить правильную скобочную последовательность внутри выделенной пары скобок равно, очевидно, C_{k-1}. Вне зависимости от выбора этой последовательности мы C_{n-k} способами можем построить правильную скобочную подпоследовательность справа от выделенной пары скобок. Следовательно, по правилу произведения в каждом блоке существует $C_{k-1} \cdot C_{n-k}$ способов построить правильную скобочную последовательность длины $2n$. Общее же число способов получить такую последовательность согласно правилу суммы равно

$$C_n = \sum_{k=1}^{n} C_{k-1}\, C_{n-k}, \qquad n = 1, 2, \ldots; \qquad C_0 = 1. \tag{37}$$

Заметим, что рекуррентное соотношение для чисел Каталана является нелинейным. Кроме того, n-й член последовательности C_n зависит от всех n предыдущих членов этой последовательности.

6 Основные понятия и определения теории графов

6.1. Начнем данный параграф с определения неориентированных и ориентированных графов.

6.1.1. Формальное и достаточно общее определение неориентированного графа таково.

Определение 6.1. *Неориентированным графом* G называется тройка

$$G = (V, E, I),$$

состоящая из

(1) (конечного) множества вершин $V = V(G)$, например,

$$V = \{1, 2, 3, 4\},$$

(2) (конечного) множества ребер $E = E(G)$, например,

$$E = \{a, b, c, d, e, f, g, h\},$$

(3) а также отображения $I\colon E \to V_2$, сопоставляющего любому ребру $e \in E$ неупорядоченную пару вершин $\{x, y\} \in V_2$, которую это ребро соединяет.

Вершины x и y называются *концевыми вершинами* ребра e. При этом говорят, что ребро e *инцидентно* своим концевым вершинам. Про любую из двух вершин x, y, в свою очередь, говорят, что она инцидентна ребру e.

В принципе, возможен случай $x = y$. Ребро $e \in E$, соответствующее паре $\{x, x\}$, называется обычно *петлей.* Кроме того, в общем случае у нас могут быть несколько различных ребер, соединяющих одну и ту же пару вершин $\{x, y\}$. Такие ребра называют кратными ребрами и говорят, что они образуют *мультиребро* графа G. Если пара вершин $\{x, y\}$ соединена между собой единственным ребром, то такое ребро иногда называют простым ребром графа.

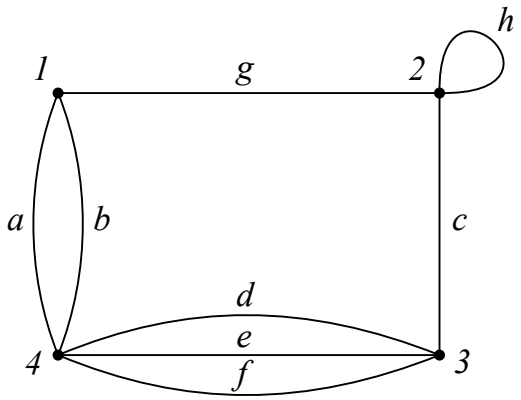


Рис. 12: Пример графа на четырех вершинах

Пример 6.2. Зададим отображение I в виде следующей таблицы:

E	a	b	c	d	e	f	g	h
V_2	$\{1,4\}$	$\{1,4\}$	$\{2,3\}$	$\{3,4\}$	$\{3,4\}$	$\{3,4\}$	$\{1,2\}$	$\{2,2\}$

Ей соответствует граф G, изображенный на рис.12. В этом графе ребра c и g являются простыми, кратные ребра a и b образуют мультиребро, соединяющее вершины 1 и 4, а ребро h представляет собой петлю.

6.1.2. С понятием инцидентности тесно связано важное понятие степени вершины графа G.

Определение 6.3. В неориентированном графе G *степенью* $\deg(x)$ или *валентностью* вершины x называется количество ребер, инцидентных x. При этом считается, что петля дает вклад, равный двум, в степень любой вершины.

Так, вершина 1 на рис.12 имеет степень, равную трем, а вершина 2 — степень, равную четырем.

Следующее утверждение часто называют первой теоремой теории графов.

Теорема 6.4. *В неориентированном графе G сумма степеней всех вершин равна удвоенному количеству всех ребер графа:*

$$\sum_{x\in V(G)} \deg(x) = 2|E(G)|. \tag{38}$$

Доказательство практически очевидно — любое ребро дает вклад, равный двум, в стоящую слева сумму.

Следствие 6.5. *Количество вершин в графе G, имеющих нечетную степень, четно.*

6.1.3. Несмотря на достаточную очевидность доказательства теоремы 6.4, оно использует чрезвычайно важный прием, очень часто встречающийся и в теории графов, и в комбинаторике — так называемый двойной подсчет (double counting).

Неформально его можно себе представлять следующим образом. Предположим, что у нас имеется какое-то множество C элементарных объектов C_i, любой из которых мы можем рассматривать как составную часть двух различных, более сложно устроенных объектов A_j и B_k. Тогда количество $c = |C|$ всех элементарных объектов C_i мы всегда сможем сосчитать двумя способами. Именно, для любого сложного объекта первого типа A_j мы можем сосчитать количество a_j элементарных объектов, из которых состоит A_j, а затем просуммировать полученные числа a_j по j от 1 до n, где n есть количество всех сложных объектов первого типа. С другой стороны, мы можем для любого B_k сосчитать количество b_k элементарных объектов, из которых состоит сложный объект второго типа B_k, а затем просуммировать b_k по всем k от 1 до m, где m — количество объектов второго типа. Так как общее количество c всех элементарных объектов C_i, от способа подсчета не зависит, то мы в итоге получаем равенство вида

$$\sum_{j=1}^{n} a_j = c = \sum_{k=1}^{m} b_k, \tag{39}$$

которое и отражает некоторый конечный результат процедуры double counting.

При доказательстве первой теоремы теории графов мы в качестве элементарного объекта C_i можем взять так называемое *полуребро* графа G (semiedge или dart), то есть половинку любого ребра, инцидентного вершине x (см.рис.13, на котором полуребра изображены синим цветом). Каждое ребро состоит ровно из двух полуребер, поэтому количество c всех полуребер равно удвоенному количеству $2m$, $m = |E(G)|$, всех ребер. С другой стороны, любую вершину x_j мы можем представить в виде "ежа", то есть вершины вместе с инцидентными ей полуребрами. Количество a_j таких полуребер совпадает со степенью $\deg(x_j)$ вершины x_j. Суммируя эти числа a_j по всем $j = 1, \ldots, n$, где n — количество всех вершин в графе, мы вновь получаем общее количество полуребер c. Формула (38) при этом есть не что иное, как соотношение (39), переписанное для рассматриваемого частного случая.

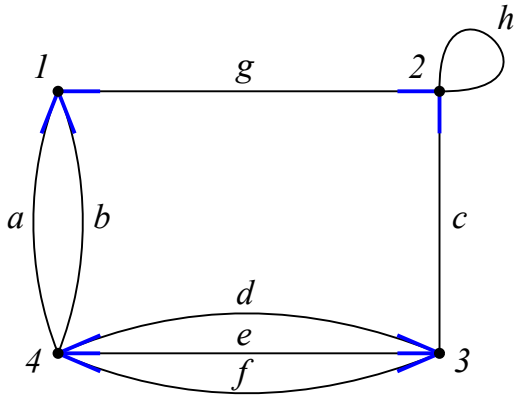


Рис. 13

6.1.4. Описанную выше процедуру double counting можно формализовать с помощью так называемой матрицы инцидентности M_i. Именно, рассмотрим матрицу M_i размерами $n \times m$, строки которой отвечают объектам A_j, $j = 1, \ldots, n$ первого типа, а столбцы — объектам B_k, $k = 1, \ldots, m$ второго типа. Назовем объекты A_j и B_k *инцидентными* друг другу, если у нас существует хотя бы один элементарный объект, являющийся частью как A_j, так и B_k. Любой элемент $m_{j,k}$ матрицы M_i равен количеству элементарных объектов, входящих как в A_j, так и в B_k. При таком подходе левая часть равенства (39) подсчитывает количество элементарных объектов суммированием по строкам матрицы M_i, а правая часть (39) — суммированием по столбцам.

В случае неориентированного графа G строки матрицы M_i инцидентности графа G отвечают вершинам этого графа, а столбцы — его ребрам. Элемент $m_{j,k}$ матрицы M_i равен нулю, если вершина x_j и ребро e_k не инцидентны друг другу, $m_{j,k} = 2$ в случае, если e_k есть инцидентная вершине x_j петля, и $m_{j,k} = 1$ в остальных случаях.

Рассмотрим, к примеру, граф G, показанный на рисунках 12 и 13. Для такого графа матрица инцидентности имеет следующий вид:

$$M_i = \left(\begin{array}{c|cccccccc} V\backslash E & a & b & c & d & e & f & g & h \\ \hline 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 2 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 2 \\ 3 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 4 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 \end{array}\right).$$

Видно, что сумма элементов в любом столбце этой матрицы равна двум, а сумма элементов в любой строке совпадает со степенью вершины x этого графа. Суммирование по строкам и суммирование по столбцам снова приводит нас к равенству (38).

6.2. Чаще всего на практике встречаются так называемые простые графы, к описанию которых мы и перейдем.

6.2.1. Начнем с определения простого графа.

Определение 6.6. Граф G называется *простым,* если он не содержит петель и мультиребер. Граф, не являющийся простым, часто называют *мультиграфом.*

В качестве примера рассмотрим изображенный на рис.14,а простой граф G, построенный на четырех вершинах и имеющий два ребра. Видно, что в таком графе мы можем не вводить какое-

(a) Граф G (b) Граф $\bar{G}$

Рис. 14: Примеры простых графов

то дополнительное специальное обозначение для ребер — любое ребро однозначно задается парой вершин, которые оно соединяет. Как следствие, для описания простого графа G нам достаточно задать множество V его вершин, а также множество E его ребер в виде некоторого подмножества множества всевозможных неупорядоченных пар множества вершин. Так, для графа G на рис.14,a мы имеем

$$V = \{1, 2, 3, 4\}, \qquad E = \{\{1, 2\}, \{1, 3\}\}.$$

Итак, с формальной точки зрения любой простой неориентированный граф G, построенный на n вершинах, можно рассматривать как некоторое подмножество множества $V^{(2)}$ всех двухэлементных подмножеств множества $V(G)$ его вершин:

$$G \subseteq V^{(2)}.$$

Если же говорить менее формально, то простой граф — это граф, построенный на n вершинах, любая пара которых может быть соединена или не соединена ребром.

Сразу заметим, что матрица M_i инцидентности простого графа состоит только из нулей и единиц.

(a) Полный граф K_5 (b) Граф $\bar{K}_5$

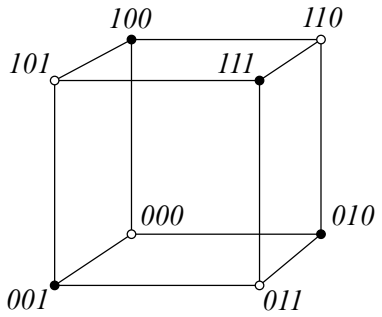
Рис. 15: Полный граф и пустой граф

6.2.2. Разберем некоторые важные примеры простых графов. В качестве первого примера рассмотрим так называемый *полный* граф K_n (см.рис.15,a, на котором показан полный граф K_5), в котором любая вершина соединена со всеми оставшимися вершинами графа. Такой граф отвечает всему множеству $V^{(2)}$ двухэлементных подмножеств множества n вершин. Пустому подмножеству множества $V^{(2)}$ отвечает так называемый *пустой* граф, то есть граф, состоящий из n изолированных вершин (рис.15,b), то есть вершин, степени которых равны нулю.

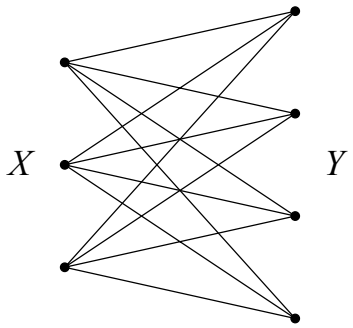
Пустой граф является дополнением к полному графу K_n (и обозначается он как $\bar{K}_n$) в смысле следующего определения.

Определение 6.7. Граф $\bar{G}$ называется *дополнением* к графу G, если множества вершин этих двух графов совпадают, а множество ребер графа $\bar{G}$ дополняет множество ребер $E(G)$ исходного графа G до множества ребер полного графа K_n.

На рис.14 в качестве примера приведены два графа, построенные на четырех вершинах — граф G и граф $\bar{G}$. Граф $\bar{G}$ из графа G можно получить, например, так: взять полный граф K_4, построенный на том же количестве вершин, что и графы G и $\bar{G}$, и удалить из него ребра, принадлежащие графу G. Полученный в результате этой операции граф $\bar{G}$ и будет являться дополнением к графу G в смысле данного выше определения.



(a) Граф Q_3



(b) Полный двудольный граф $K_{3,4}$

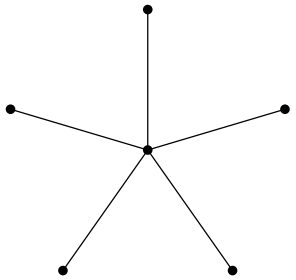

(c) Граф "звезда" $K_{5,1}$

Рис. 16: Двудольные графы

6.2.3. При изучении теории графов нам часто будут встречаться еще несколько важных подклассов простых графов.

Определение 6.8. Граф G называется *двудольным* (рис.16), если множество $V(G)$ его вершин можно разбить на два блока X и Y так, что концы x и y любого ребра $e = \{x, y\} \in E(G)$ лежат в разных блоках этого разбиения. Обозначается двудольный граф с разбиением (X, Y) через $G[X, Y]$.

Простой граф $G[X, Y]$, в котором любая вершина из X соединена ребром с каждой вершиной из блока Y и наоборот, называется *полным двудольным графом* (рис.16,b). Часто такой граф обозначается через $K_{n,m}$, где $n = |X|$, а $m = |Y|$. Граф $K_{n,1}$ называется *звездой* (рис.16,c).

(a) Путь P_4

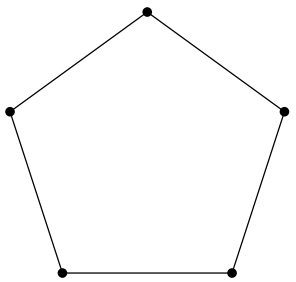
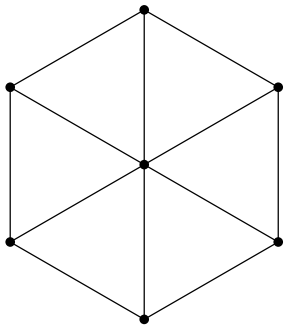
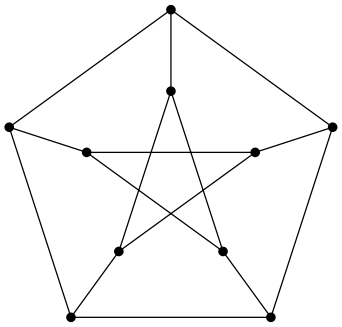

(b) Цикл C_5

Рис. 17: Путь и цикл

Еще два важных подкласса простого графа, называемые путем P_n и циклом C_n, показаны на рис.17,a и рис.17,b. Количество ребер в графах P_n и C_n характеризует длину таких графов. Также часто нам будет встречаться граф W_n, называемый колесом (см.рис.18,a). Индекс n в обозначении W_n такого графа обозначает либо количество вершин в его внешнем цикле, либо количество всех вершин. Мы будем считать, что n — это количество вершин во внешнем цикле графа W_n.



(a) Колесо W_6



(b) Граф Петерсена

Рис. 18: Колесо и граф Петерсена

Важный класс простых подграфов представляют собой так называемые *регулярные графы,* то есть графы, у которых все вершины имеют одну и ту же степень k. Также их называют k-регулярными графами или регулярными графами степени k. В случае $k = 3$ такие графы называются кубическими. На рис.18,b показан пример кубического графа, который достаточно часто будет встречаться у нас в приложениях — так называемый граф Петерсена (Petersen graph). Графы C_n и K_n также являются регулярными графами степени 2 и $n-1$ соответственно.

Наконец, имеется важный класс графов, являющихся одновременно k-регулярными и двудольными — так называемые k-кубы Q_k. Вершины графа Q_k можно пометить бинарными строками длины k (см.рис.16,a). Ребра же в таком графе проводятся только между теми вершинами, бинарные последовательности которых отличаются только в одной из k позиций. Пример графа Q_3 показан на рис.16,a.

6.3. Наряду с неориентированными, в теории графов также изучаются и так называемые ориентированные графы (или орграфы).

6.3.1. Начнем с мы вновь с достаточно формального и общего определения ориентированного мультиграфа D.

Определение 6.9. Если в тройке

$$D = (V, E, I),$$

в которой V есть множество вершин, а E — множество ребер, отображение I ставит в соответствие любому ребру e *упорядоченную* пару вершин $(x, y) \in V \times V$, то такая тройка называется *ориентированным* графом (или *орграфом*). В таком случае говорят, что ребро e *выходит* из вершины x и *входит* в вершину y. На рисунке такое ребро помечается стрелкой, указывающей направление данного ребра.

В качестве примера на рис.19,a показан ориентированный граф D, построенный на множестве вершин $V = \{1, 2, 3\}$ и имеющий три ребра. Ребро a выходит из вершины 1 и входит в вершину 2 (то есть отвечает упорядоченной паре $(1, 2)$), а ребра b и c исходят из вершины 2 и приходят в

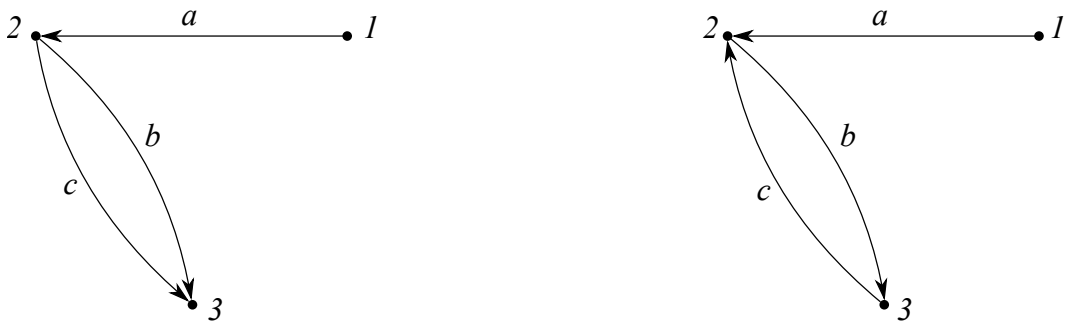


Рис. 19: Примеры ориентированных графов

вершину 3 (то есть им соответствует упорядоченная пара $(2,3)$ вершин графа D). Таким образом, орграф D на рис. задается тройкой (V,E,I), в которой множество V вершин и множество E ребер имеют вид

$$V=\{1,2,3\}, \qquad E=\{a,b,c\},$$

а отображение $I\colon E\to V\times V$ задается таблицей вида

E	a	b	c
$V\times V$	$(1,2)$	$(2,3)$	$(2,3)$

6.3.2. Для беспетлевых орграфов можно ввести понятие матрицы инцидентности $M_i(D)$. Каждая строка такой матрицы отвечает некоторой вершине орграфа, каждый столбец — некоторому ребру. В случае, если из вершины x_i исходит ребро e_j, элемент $m_{i,j}$ матрицы инцидентности считается равным $+1$, а в случае, если в вершину x_i входит ребро e_j, то $m_{i,j}$ полагают равным -1.

Так, для изображенного на рис.19 орграфа D матрица инцидентности $M_i(D)$ имеет следующий вид:

$$M_i=\left(\begin{array}{c|ccc} V\backslash E & a & b & c \\ \hline 1 & 1 & 0 & 0 \\ 2 & -1 & 1 & 1 \\ 3 & 0 & -1 & -1 \end{array}\right).$$

6.3.3. В орграфе различают исходящую ($\mathrm{outdeg}(x)$) и входящую ($\mathrm{indeg}(x)$) степень любой вершины $x\in V(D)$. Так, в графе, изображенном на рис.19,а, вершина 2 имеет входящую степень, равную единице, и исходящую степень, равную двойке.

Так как любое ориентированное ребро в орграфе D вносит вклад, равный единице, в сумму всех исходящих степеней вершин орграфа D, а также вклад, равный единице, в сумму всех входящих степеней вершин орграфа D, то для орграфа справедливо равенство вида

$$\sum_{x\in V(D)}\mathrm{indeg}(x)=|E(D)|=\sum_{x\in V(D)}\mathrm{outdeg}(x), \tag{40}$$

являющееся аналогом первой теоремы теории графов (6.4).

6.3.4. Как и для неориентированного графа, важным частным случаем орграфа является простой орграф.

Определение 6.10. Орграф D называется *простым,* если он не содержит петель, а также кратных *упорядоченных* ребер, то есть ребер, отвечающих одинаковым упорядоченным парам вершин.

Упорядоченность вершин в этом определении важна. Так, граф из рис.19,a простым не является — в нем ребра b и c отвечают одной и той же упорядоченной паре $(2, 3)$ вершин. Изображенный же на рис.19,b граф D' является простым. Несмотря на то, что в этом графе по-прежнему имеются два ребра, соединяющих вершины 2 и 3, направлены эти ребра в разные стороны. Иными словами, эти ребра отвечают различным упорядоченным парам вершин — ребро b отвечает упорядоченной паре $(2, 3)$, а ребро c — упорядоченной паре $(3, 2)$.

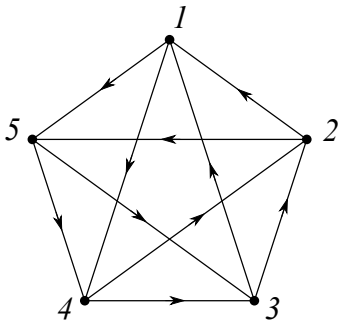


Рис. 20

6.3.5. Важным частным случаем простого орграфа, довольно часто встречающимся на практике, является турнир T — орграф, полученный ориентацией ребер полного графа K_n (рис.20).

Определение 6.11. Ориентацией графа G называется орграф D, полученный из G ориентацией каждого из ребер G, то есть заменой любого ребра $\{x, y\} \in E(G)$ неориентированного графа либо на ребро $(x, y) \in E(D)$, либо на ребро $(y, x) \in E(D)$.

Определение 6.12. Турниром T называется орграф, полученный ориентацией полного графа K_n.

Любой орграф T моделирует результат так называемого кругового турнира (round-robin tournament), в котором каждый игрок встречается с каждым другим участником ровно один раз. Считается при этом, что результатом встречи может быть победа или поражение, ничьи правилами турнира не предусматриваются. Ребро (x, y) в турнире направлено от выигравшего игрока x к проигравшему y. Как следствие, исходящая степень $\mathrm{outdeg}(x)$ произвольного турнира T определяет количество очков (score), набранных в турнире соответствующей командой.

6.4. Следующим важным понятием в теории графов является понятие смежности вершин.

6.4.1. Начнем с определения смежных вершин в неориентированном графе G.

Определение 6.13. Говорят, что в неориентированном графе G вершина y смежна с вершиной x, если в этом графе существует ребро $\{x, y\}$.

На множестве вершин V смежность задает некоторое отношение. Для неориентированного графа G это отношение является симметричным: если вершина x смежна с вершиной y, то и вершина y смежна с вершиной x.

6.4.2. Для ориентированного графа ситуация несколько сложнее.

Определение 6.14. Говорят, что в ориентированном графе D вершина y смежна с вершиной x, если в этом графе существует ребро (x, y), исходящее из вершины x и входящее в вершину y. (см.рис.19,a). Как следствие, все смежные с x вершины — это вершины, в которые ведут ребра из вершины x.

Для изображенного на рис.19,a орграфа G вершина 2 смежна с вершиной 1.

Сразу заметим, что если существует ребро (x, y) и не существует ребро (y, x), то вершина y смежна с вершиной x, а вот x вершиной, смежной с y, уже не является. Так, для изображенного на рис.19,a орграфа вершина 1 смежной с вершиной 2 не является. Вершина x будет смежной с y только в случае, когда в орграфе D существует ребро (y, x) (см. вершины 2 и 3 на рис.19,b). Иными словами, в ориентированном графе D отношение смежности на множестве V вершин симметричным не является.

6.5. Для хранения графа в памяти компьютера, как правило, используются две структуры, тесно связанные с понятием смежности — матрица смежности и список смежности.

6.5.1. Матрица смежности — это матрица M_a размерами $n \times n$, любой элемент a_{ij} которой описывает количество ребер, соединяющих вершины i и j. Так, для примера 6.2 соответствующая графу G матрица смежности имеет следующий вид:

$$M_a = \begin{bmatrix} 0 & 1 & 0 & 2 \\ 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 3 \\ 2 & 0 & 3 & 0 \end{bmatrix}$$

В случае ориентированных графов элементу a_{ij} отвечает количество ребер, идущих из вершины i в вершину j. Так, для ориентированных графов, показанных на рис.19, матрицы смежности равны

$$M_a = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 2 \\ 0 & 0 & 0 \end{bmatrix} \qquad \text{и} \qquad M_a = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

Заметим, что для неориентированного графа матрица смежности всегда симметрична. Как следствие, все собственные значения этой матрицы являются вещественными числами.

В случае простого графа или орграфа все диагональные элементы $a_{ii} = 0$, так что сумма собственных значений такой матрицы, совпадающая со следом матрицы M_a, также равняется нулю. Элементы, не лежащие на диагонали, равны единице в случае, если существует ребро, идущее из вершины i в вершину j, и нулю в случае, если такового ребра не существует.

6.5.2. Список смежности — это линейный массив L_a размера n, каждый элемент a_i которого содержит список (мультимножество) вершин, смежных с вершиной i. Для примера 6.2 соответствующий список имеет следующий вид:

1 (смежные с 1 вершины): $2, 4, 4$;
2 (смежные с 2 вершины): $1, 2, 3$;
3 (смежные с 3 вершины): $2, 4, 4, 4$;
4 (смежные с 4 вершины): $1, 1, 3, 3, 3$.

Для ориентированного графа напротив каждой вершины с номером i стоят вершины (возможно, повторяющиеся), в которые идут ребра из i. Как следствие, для показанного на рис.19,a

орграфа список смежности записывается так:

1 (смежные с 1 вершины): 2;
2 (смежные с 2 вершины): 3, 3;
3 (смежные с 3 вершины): $\varnothing$.

7 Маршруты, пути, циклы в графе. Связные графы и орграфы

7.1. Следующая важная группа понятий теории графов связана с обходом графа вдоль некоторой последовательности его вершин и ребер. В этой связи нам понадобятся такие понятия, как маршруты и пути в графе, а также некоторые другие связанные с ними характеристики графа.

7.1.1. Начнем с определения маршрута в мультиграфе G.

Определение 7.1. *Маршрутом* (walk) в графе G из вершины x_0 в вершину x_k называется чередующаяся последовательность

$$W := x_0, e_1, x_1, e_2, x_2, \ldots, x_{k-1}, e_k, x_k$$

вершин $x_i \in V$ и ребер $e_i \in E$, соединяющих вершины x_{i-1} и x_i. И вершины, и ребра в такой последовательности могут повторяться. Количество ребер в маршруте W называется *длиной* k этого маршрута.

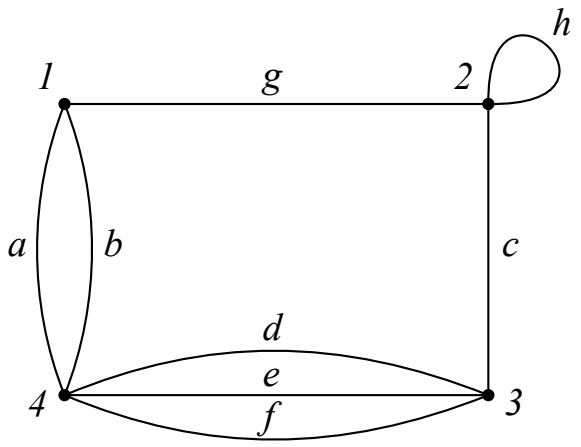


Рис. 21

В случае простого графа любой маршрут W полностью определяется последовательностью

$$x_0, x_1, x_2, \ldots, x_{k-1}, x_k,$$

вершин $x_i \in V(G)$, любые два последовательных элемента x_{i-1}, x_i которой являются смежными вершинами (т.е. соединены между собой ребром $e_i = \{x_{i-1}, x_i\} \in E(G)$).

Вершины x_0 и x_k называют часто начальной и конечной вершинами маршрута W, а остальные вершины — внутренними его вершинами. Говорят также, что вершины x_0 и x_k связаны маршрутом W, а сам маршрут называют x_0x_k-маршрутом.

Определение 7.2. Если все ребра $e_1, \ldots, e_k$ в маршруте различны, то такой маршрут называется *путем* (в английской литературе — trail) из вершины x_0 в вершину x_k. Если также и все вершины в данном пути различны, то такой путь называется *простым* (в английской литературе — path).

В качестве примера рассмотрим разные маршруты в графе, показанном на рис.21. Маршрут

$$1, a, 4, d, 3, e, 4, d, 3, c, 2, h, 2$$

путем не является — в нем повторяется ребро d. Маршрут

$$1, a, 4, d, 3, e, 4, f, 3, c, 2, h, 2$$

представляет собой путь (trail), который не является простым — в нем повторяются вершины $2, 3, 4$. Наконец, маршрут

$$1, a, 4, e, 3, b, 2$$

является простым путем (path) из вершины 1 в вершину 2.

7.1.2. Понятие пути позволяет нам дать крайне важное определение связного графа.

Определение 7.3. Если вершины $x, y \in V$ графа G соединены хотя бы одним путем, то такие вершины называются *связанными.*

Несложно проверить, что связанность задает на множестве V вершин графа G отношение эквивалентности. Это отношение делит граф на классы эквивалентности, называющиеся *компонентами связности графа.*

Определение 7.4. В случае, когда в графе G существует лишь одна компонента связности, то есть в случае, когда любые две вершины x, y графа соединены хотя бы одним путем, граф называется *связным.* В противном случае граф называется *несвязным.*

7.1.3. Перейдем теперь к понятиям, характеризующим метрические свойства графа.

Определение 7.5. *Расстоянием* $d(x, y)$ между двумя связанными вершинами $x, y \in V(G)$ называется длина наименьшего пути между ними.

Понятно, что такой путь обязательно является простым. В случае, когда вершины не являются связанными, полагают по определению, что $d(x, y) = \infty$.

Определение 7.6. *Диаметром* графа называется максимальное расстояние между его вершинами:

$$\operatorname{diam}(G) := \max_{x,y \in V(G)} d(x, y).$$

В случае несвязного графа считается, что $\operatorname{diam}(G) = \infty$.

Определение 7.7. *Эксцентриситетом* $\varepsilon(x)$ вершины $x \in V(G)$ называется максимальное расстояние от x до любой другой вершины графа G:

$$\varepsilon(x) := \max_{y \in V(G)} d(x, y).$$

Определение 7.8. *Радиусом* $r(G)$ графа G называется минимальный из эксцентриситетов вершин графа G. Вершины, на которых этот минимум достигается, называются *центральными вершинами* графа G. Множество всех центральных вершин называется *центром* графа.

Некоторые несложные факты, связанные с введенными выше понятиями, приведены в упражнениях **??** и **??**.

7.1.4. Следующая серия определений связана с замкнутыми маршрутами в графах, то есть маршрутами, в которых начальная x_0 и конечная x_k вершины совпадают.

Определение 7.9. Замкнутым путем (closed trail) или составным циклом (circuit) в графе G называется путь, в котором $x_0 = x_k$. Путь, в котором совпадают только начальная и конечная вершины, называется простым циклом (cycle). Часто слово "простой" опускают, понимая под циклом замкнутый простой путь.

Так, в графе G, показанном на рис.21, путь $1, a, 4, e, 3, d, 4, b, 1$ является составным циклом, а путь вида $1, a, 4, e, 3, c, 2, g, 1$ — простым циклом. Замкнутый маршрут вида $1, a, 4, e, 3, e, 4, b, 1$ составным циклом не является — в нем повторяется ребро e.

В простом графе невозможны циклы длины меньшей, чем три. Циклы длины три часто называются треугольниками. Графы, в которых такие циклы отсутствуют, называются графами, свободными от треугольников (triangle-free graphs).

Определение 7.10. Обхватом графа (girth) называется длина наименьшего цикла в нем. Если в графе циклы отсутствуют, то обхват такого графа считают равным бесконечности.

С учетом данного определения простые графы, свободные от треугольников — это графы, обхват которых больше трех.

В случае мультиграфов возможны и циклы длины 2, и циклы длины 1. Циклы длины 2 — это циклы типа $3, d, 4, e, 3$ на рис.21, появляющиеся при обходе двух ребер одного и того же мультиребра. Циклы длины 1 — это петли графа G. Так, петлю h на рис.21 можно рассматривать как цикл $2, h, 2$ длины 1.

Изолированная вершина по определению является замкнутым путем длины 0, однако циклом она не является.

7.1.5. В предыдущем параграфе мы ввели важный подкласс простых графов — так называемые двудольные графы $G[X, Y]$. Оказывается, что такие графы допускают характеризацию в терминах циклов.

Теорема 7.11 (König)**.** *Граф G является двудольным тогда и только тогда, когда в нем отсутствуют циклы нечетной длины (так называемые нечетные циклы).*

Доказательство. То, что в любом двудольном графе нечетные циклы отсутствуют, достаточно очевидно. Действительно, для того, чтобы, выйдя из произвольной вершины x блока X, затем в нее же и вернуться, нам необходимо сделать четное число шагов. Следовательно, любой цикл в двудольном графе $G[X, Y]$ обязан иметь четную длину.

Предположим теперь, что в графе G циклы нечетной длины отсутствуют. Сразу заметим, что при доказательстве нам достаточно ограничиться связными графами — любой несвязный граф является двудольным тогда и только тогда, когда двудольной является любая из его компонент

связности. Кроме того, мы можем считать, что граф G построен на $n > 1$ вершинах — в случае $n = 1$ граф $G = K_1$ по определению является двудольным.

Выберем в G произвольную вершину $x \in V(G)$ и разобъем множество $V(G)$ вершин на два блока. К блоку Y отнесем все вершины $y \in V(G)$, для которых длина кратчайшего пути из z в y нечетна, а к блоку X отнесем все оставшиеся вершины. В частности, сама вершина x принадлежит блоку X, а все смежные с ней вершины принадлежат подмножеству Y. Докажем, что любые две вершины x_p, x_q из подмножества X смежными не являются (рис. 22).

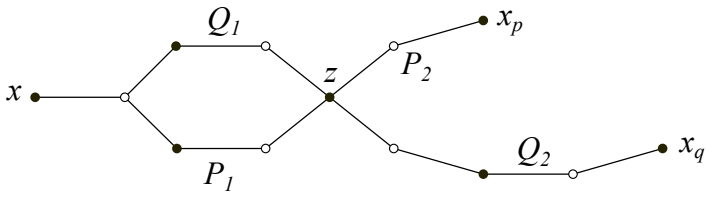


Рис. 22

Рассмотрим для этого произвольные кратчайшие пути P и Q, соединяющие эти вершины с вершиной x. Обозначим через $z \in V(G)$ последнюю общую вершину этих путей. Вершина z разделяет пути P и Q на два подпути — участки P_1 и Q_1 этих путей от x до вершины z, а также подпути P_2 и Q_2, соединяющие z с вершинами x_p и x_q соответственно. Сразу заметим, что длины путей P_1 и Q_1 обязаны совпадать — в противном случае мы могли бы сократить путь от x до одной из двух вершин x_p, x_q, что невозможно. Кроме того, так как x_p и x_q принадлежат блоку X, длины путей P и Q имеют одинаковую четность. Как следствие, одинаковую четность имеют и участки P_2 и Q_2 путей P и Q. Кроме того, P_2 и Q_2 не имеют никаких других общих вершин, помимо z. Добавление к ним ребра $e = \{x_p, x_q\}$ приводит к образованию в G цикла нечетной длины, чего быть не может.

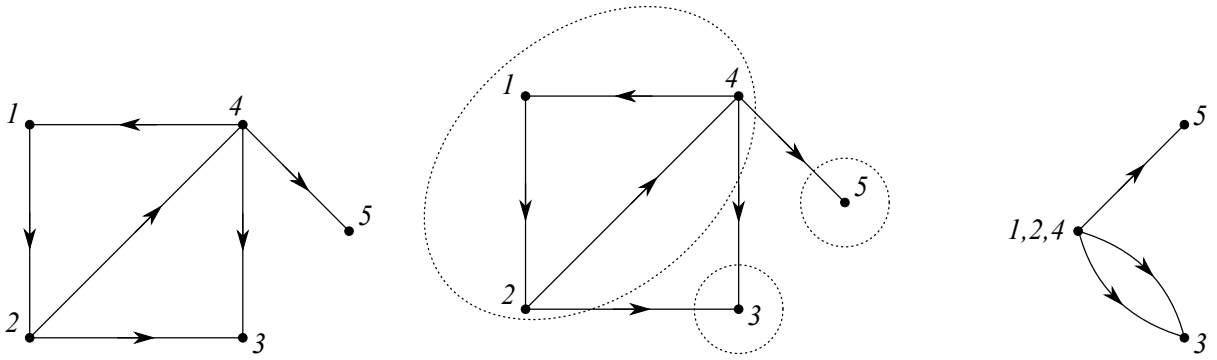
Аналогично доказывается, что и любые две вершины $y_p, y_q \in Y$ смежными не являются. Следовательно, граф G является двудольным. □

Замечание 7.12. Идея, описанная при доказательстве теоремы 7.11, лежит в основе довольно элементарного алгоритма проверки графа на двудольность, основанная на поиске в ширину (см. первый параграф второй главы). Именно, выберем произвольную вершину x графа G и окрасим ее в черный цвет. Затем запустим из этой вершины поиск в ширину и начнем оставшиеся вершины окрашивать в два цвета (черный или белый) в зависимости от четности расстояния от этих вершин до вершины x. В случае, если из какой-то вершины x' имеется ребро в уже окрашенную вершину x'', нам следует проверить, что эти две вершины окрашены в разные цвета. Если это не так, граф двудольным не является. Если же в процессе работы алгоритма конфликтов не возникнет, граф является двудольным. Более формальное описание данного алгоритма, равно как и его реализацию на языке C++, можно посмотреть, например, в книге [?].

7.2. Рассмотрим теперь понятие связности в ориентированном графе D.

7.2.1. Заметим, прежде всего, что понятия маршрута, пути, простого пути и цикла достаточно естественно переносятся на ориентированные графы. В качестве примера рассмотрим ориентированный граф D, показанный на рис.23,а. В таком графе последовательность

$$(1, 2, 4, 1, 2, 3)$$



(a) Орграф D (b) Компоненты орграфа D (c) Граф $C(D)$

Рис. 23

задает нам маршрут из вершины 1 в вершину 3.

7.2.2. Теперь мы можем перейти к определению понятия связанных вершин в орграфе.

Определение 7.13. Вершины x и y орграфа D называются *связанными,* если в D существует хотя бы один путь из вершины x в вершину y, а также хотя бы один путь из вершины y в вершину x.

Вернемся к орграфу D на рис.23,a. В таком графе вершины 1 и 4 связаны между собой — мы можем пройти из 4 в 1 по ребру $(4,1)$, а также из вершины 1 в вершину 4 по простому пути $(1,2,4)$. Вершины же 1 и 3 связанными между собой не являются — из вершины 1 в вершину 3 путь существует, однако из вершины 3 в вершину 1 добраться невозможно.

7.2.3. Как и в случае неориентированных графов, в случае орграфов отношение связанности является отношением эквивалентности. Как всякое отношение эквивалентности, оно разбивает множество $V(D)$ вершин орграфа D на классы эквивалентности, называемые *компонентами сильной связности* орграфа D.

В качестве примера рассмотрим орграф, показанный на рис.23,a. Множество вершин такого орграфа разбивается отношением связанности на три блока — блок, состоящий из вершин $1, 2, 4$, блок, состоящий из единственной вершины 3, а также блок, состоящий из единственной вершины 5 (рис.23,b).

Определение 7.14. Орграф D называется *сильно связным,* если он состоит из единственной компоненты сильной связности, то есть если любые две его вершины являются связанными.

Иногда наряду с этим понятием для орграфа вводят понятие слабой связности.

Определение 7.15. Орграф D называется *слабо связным,* если соответствующий ему неориентированный граф G, получающийся заменой всех ориентированных ребер на неориентированные, является связным.

Показанный на рис.23,a орграф D является примером слабо связного орграфа, не являющегося сильно связным.

7.2.4. Вернемся к отношению связанности в орграфе, разбивающему множество $V(D)$ вершин орграфа D на классы эквивалентности — компоненты сильной связности. Когда мы рассматривали отношение связанности в неориентированном графе G, мы говорили, что между компонентами связности такого графа ребра отсутствуют. В случае ориентированного графа это не

так (см., например, рис.23,a) — в орграфе такие ребра могут существовать, однако направлены все они будут лишь в одну сторону, от одной компоненты связности к другой.

Именно, справедливо следующее достаточно очевидное утверждение.

Лемма 7.16. *Пусть H_1, H_2 есть две различные компоненты сильной связности графа D, и пусть существует ребро $e \in E(D)$ из H_1 в H_2. Тогда ребра из H_2 в H_1 отсутствуют.*

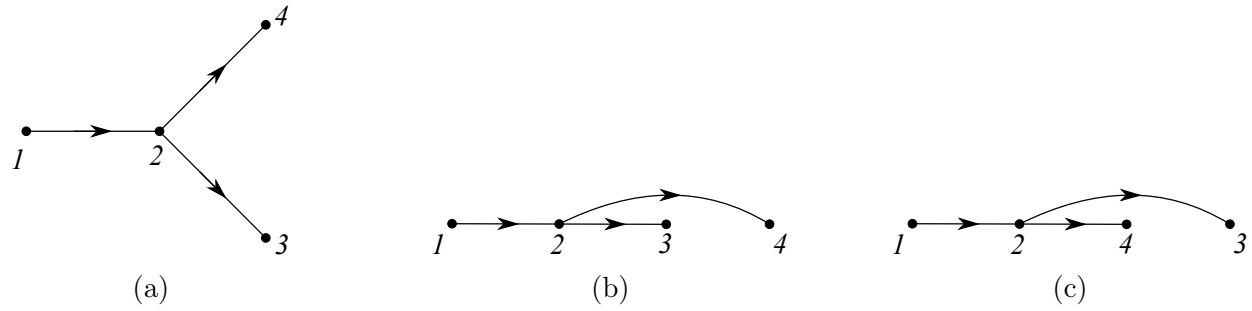
Доказательство. Действительно, если бы ребро из H_2 в H_1 присутствовало в D, то тогда любые две вершины в множестве $H_1 \cup H_2$ вершин оказались бы связанными. Но тогда объединение $H_1 \cup H_2$ представляло бы собой компоненту сильной связности графа D, что невозможно — мы изначально предполагали, что H_1 и H_2 представляют собой две различные компоненты сильной связности. □

7.2.5. По любому орграфу D можно построить так называемый граф $C(D)$ компонент сильной связности графа D, вершинами которого будут компоненты сильной связности графа D, а ребрами — ребра графа D, направленные из одной компоненты сильной связности D к другой (рис.23,c). Основное свойство такого орграфа $C(D)$ состоит в том, что в таком графе нет циклов.

Теорема 7.17. *В орграфе $C(D)$ циклы отсутствуют, то есть он, как еще говорят, представляет собой ациклический орграф (DAG — directed acyclic graph).*

Доказательство. Если бы в таком графе существовал цикл вида $H_1 \to H_2 \to \ldots \to H_n \to H_1$, то любые две вершины в объединении $H_1 \cup H_2 \cup \ldots \cup H_k$ оказались бы связанными. Действительно, внутри каждой компоненты H_i мы, по определению сильной связности, можем попасть из любой вершины в любую вершину H_i. Вершины же из разных компонент H_i и H_j мы также можем всегда связать с помощью пути, идущего из H_i в H_j, а также пути, соединяющего компоненты H_j и H_i. □

7.2.6. Ориентированные ациклические графы достаточно часто встречаются в практических приложениях. В качестве характерного примера можно рассмотреть орграф D, моделирующий порядок изложения материала в учебнике или онлайн-курсе. Любой параграф (урок) можно изображать вершиной орграфа D. Ребро между вершинами x и y в D проводится в случае, если соответствующий вершине x параграф является пререквизитом к параграфу, моделируемому вершиной y, то есть в случае, когда для изучения материала, изложенного в параграфе y, необходим материал, изложенный в параграфе x. Довольно очевидно, что в таком орграфе D циклов быть не должно — в противном случае параграф x ссылался бы на параграф y и наоборот, и было бы не очень понятно, как изучать подобным образом организованный материал.



(a) (b) (c)

Рис. 24: Различные способы топологической сортировки вершин ациклического орграфа

Теперь предположим, что мы хотим организовать учебный курс, основанный на данном учебнике. Первый вопрос, который при этом возникает — это в каком порядке рассказывать изложенный в нем материал. В простейшем случае в учебнике информация подается линейно

— за первым параграфом излагается второй, за вторым — третий, и так далее. Такого рода учебник моделируется простым ориентированным путем P, и проблем с организацией курса по этому учебнику не возникает. Однако в более сложно устроенных учебниках могут встречаться параграфы, которые можно проходить в различном порядке. Так, в учебнике, модель подачи материала в котором описывается орграфом D, показанным на рис.24,a, материал первых двух параграфов излагается линейно. Затем в книге идут два параграфа, не зависящие друг от друга. Как следствие, у лектора, использующего такой учебник в своем учебном процессе, возникает дилемма — он может либо рассказывать вначале материал параграфа 3, а затем — материал, изложенный в параграфе 4 (рис.24,b), либо начать с параграфа 4 и закончить параграфом 3 (рис.24,c).

С точки зрения теории графов тот или иной способ линейного упорядочивания вершин ациклического орграфа носит название *топологической сортировки* его вершин. Топологическая сортировка называется *правильной,* если после сортировки все ребра направлены слева направо. Иными словами, вершины орграфа отсортированы правильно в случае, если для любого ребра (x, y) вершина x имеет более ранний номер в линейном порядке по сравнению с номером, присвоенном вершине y. Как мы с вами видели на примере орграфа, показанного на рис.24, для одного и того же орграфа в общем случае существует довольно много различных способов правильно отсортировать его вершины.

8 Подграф графа G. Основные операции над графами

8.1. Вернемся к основным определениям теории графов и введем очень важное понятие подграфа графа G. Заодно мы введем две основные операции над графами — операцию удаления ребра и операцию удаления вершины в графе G.

8.1.1. Начнем с формального определения подграфа H графа G.

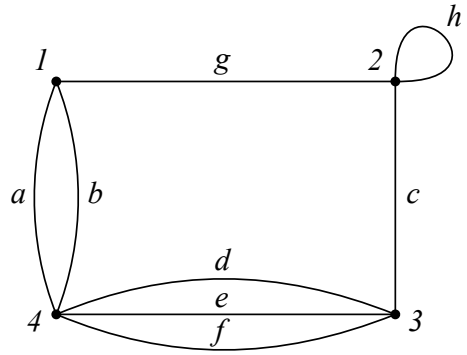
Определение 8.1. *Подграфом* графа G называется граф H, для которого выполнены следующие три условия:

1. $V(H) \subseteq V(G)$;
2. $E(H) \subseteq E(G)$;
3. любое ребро $e \in E(H)$, соединяющее пару вершин x и y в H, должно соединять ту же самую пару вершин в графе G.

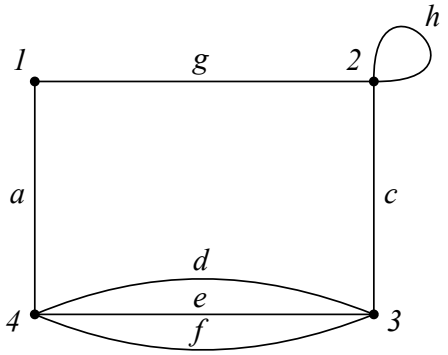
По отношению к графу H граф G иногда называют *надграфом* или *суперграфом.*

Первые два условия, данные в определении подграфа, более-менее очевидны. Третье же условие требует некоторых комментариев. Рассмотрим в качестве первого примера знакомый нам граф G (рис.25,a), а также граф H, показанный на рис.25,b. У обоих этих графов множества вершин совпадают. Кроме того, множество $E(H)$ ребер графа H представляет собой подмножество множества $E(G)$ ребер графа G (в графе H ребро b отсутствует). Наконец, любое ребро, соединяющее вершины в графе H (например, ребро $g \in E(H)$), соединяет те же самые две вершины и в графе G (например, ребро g соединяет и в H, и в G одну и ту же пару вершин — вершины 1 и 2). Как следствие, граф H является подграфом графа G.

Давайте теперь посмотрим на графы G и H', показанные на рис.26. Множества $V(G)$ и $V(H')$ у таких графов по-прежнему совпадают, множество $E(H')$ является подмножеством множества



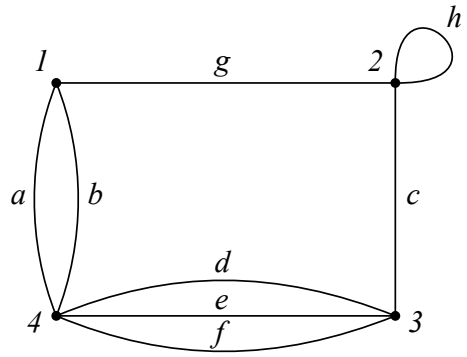
(a) Граф G



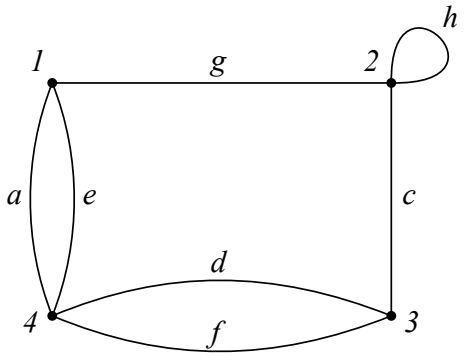
(b) Подграф H графа G

Рис. 25

$E(G)$ ребер графа G. Однако у графа H' ребро e соединяет вершины 1 и 4, тогда как в графе G это же ребро соединяет вершины 3 и 4. Как следствие, условие 3 определения подграфа для H' нарушается, так что H' подграфом графа G не является.



(a) Граф G



(b) Граф H', не являющийся подграфом графа G

Рис. 26

8.1.2. Данное выше определение можно сделать несколько более конструктивным, если ввести две основные операции над графами — операцию удаления ребра и операцию удаления вершины. Оказывается, любой подграф H графа G — это граф, полученный из исходного графа G с помощью этих двух операций.

Начнем с более простой операции удаления ребра e. При такой операции множество вершин графа G не меняется, а из множества ребер удаляется элемент $e \in E(G)$. Полученный в результате этой операции граф обозначается $G - e$. Очевидно, что он является подграфом графа G. Действительно, в этом случае $V(H) = V(G)$, и $E(H) \subset E(G)$. Кроме того, так как мы никакие другие ребра графа G мы в процессе удаления ребра e не трогали, то в этом случае условие 3 определения подграфа у нас выполняется автоматически. Так, в приведенном на рис.25 примере мы в графе G удалили ребро b, соединяющее вершины 1 и 4, и получили подграф $H = G - b$.

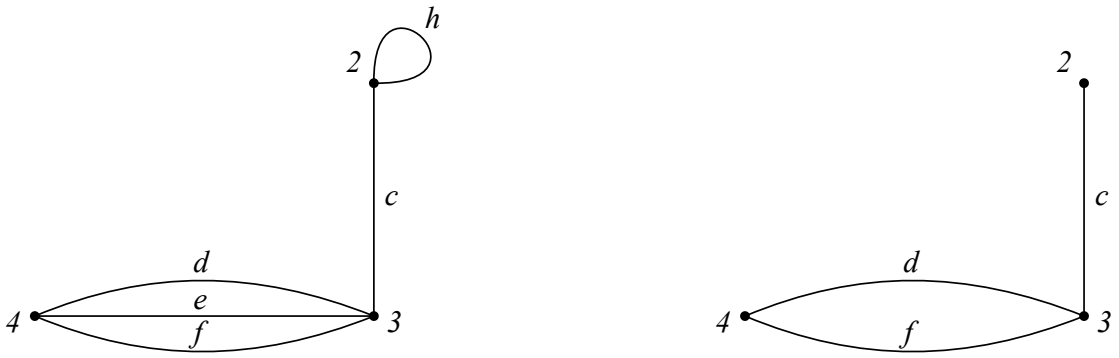
В более общем случае мы таким образом можем удалить сразу несколько ребер, принадлежащих некоторому подмножеству S множества $E(G)$ ребер графа G. Полученный в результате этих операций подграф H обозначается $G - S$.

Перейдем теперь к чуть более сложной операции удаления вершины. Предположим, что мы хотим удалить в графе G вершину x. Если эта вершина является изолированной, то нам ничего

больше делать не нужно. Если же этой вершине инцидентны какие-то ребра, то мы обязаны будем также вместе с вершиной x удалить и их. Действительно, ранее эти ребра вели в вершину x. Вести в никуда ребра не могут, так что нам вместе с вершиной x приходится удалять и все ребра, инцидентные данной вершине. Так, в показанном на рис.27,a примере мы удалили в графе G вершину 1. Вместе с ней мы вынуждены были удалить и все три инцидентных ей ребра a, b, g.

Полученный в результате удаления вершины x граф обозначается обычно $G - x$. Очевидно, что граф $H = G - x$ также является подграфом исходного графа G. Действительно, в этом случае $V(H) \subset V(G)$, $E(H) \subseteq E(G)$, а третье условие в определении подграфа у нас вновь выполняется автоматически — мы вновь ничего с оставшимися ребрами графа G не делали в процессе удаления вершины x.

В более общем случае $S \in V(G)$ мы с помощью операции удаления вершин из подмножества S получаем подграф $G - S$, в котором по сравнению с исходным графом G удалены все вершины подмножества S вместе со всеми ребрами, инцидентными этим вершинам.



(a) Граф, полученный из G удалением вершины 1

(b) Подграф H графа G

Рис. 27

Теперь рассмотрим граф H, полученный из графа G последовательным выполнением двух операций — операции удаления вершины и операции удаления ребра (см.рис.27,b). Так как для каждой операции условие 3 в определении подграфа выполняется автоматически, то мы тем самым получаем, что граф H является подграфом графа G. Иными словами, мы получаем конструктивное определение подграфа H графа G как графа, полученного из G удалением некоторого количества вершин и/или ребер.

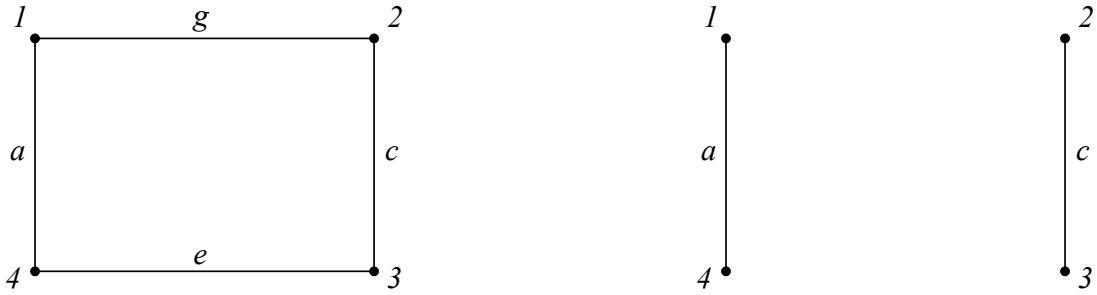
В качестве примера на рис.27,b показан граф H. Так как он получен из графа G, показанного на рис.26,a, удалением вершины 1 и ребер e, h, то граф H является подграфом графа G.

8.1.3. Как мы уже заметили, любой подграф графа G получается из G последовательным выполнением двух операций — удаления вершин и удаления ребер. Естественным кажется рассмотреть два частных случая этой ситуации. Первый — это случай, когда мы в графе G удаляем только ребра, а второй — когда в G мы удаляем только вершины.

Если мы при получении из графа G подграфа H используем лишь операцию удаления ребер, то мы получаем подграф, множество вершин которого совпадает с множеством $V(G)$ вершин исходного графа. Такой подграф называется *остовным* подграфом (spanning subgraph) графа G (смотри рис.26,b). В частности, любой простой граф G, построенный на n вершинах, является остовным подграфом полного графа K_n.

Второй частный случай — когда мы в графе G удаляем одну или несколько вершин. В результате такой операции мы получаем подграф H, *индуцированным подмножеством оставшихся вершин графа* G. Иными словами, подграфом H графа G, индуцированным подмножеством вершин S, называется граф, полученный из G удалением всех вершин, не принадлежащих множеству S, вместе со всеми инцидентными этим вершинам ребрами (смотри рис.27,a).

Иногда нам будет удобно использовать понятие подграфа, индуцированного некоторым подмножеством F множества $E(G)$ ребер графа G. Такой подграф состоит из ребер, принадлежащих F, а также из вершин, являющихся концевыми вершинами этих ребер.



(a) 2-фактор графа G (b) 1-фактор графа G

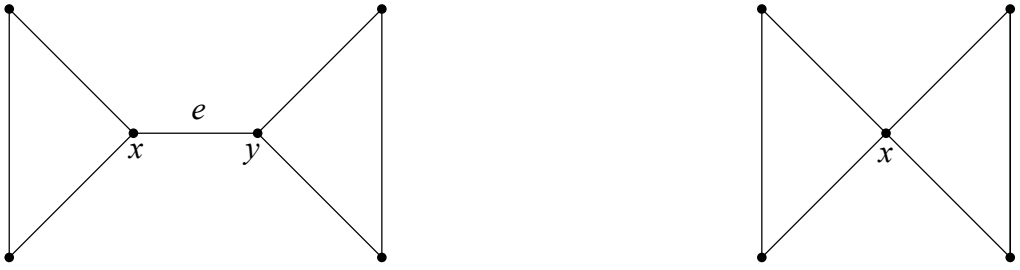
Рис. 28: k-факторы графа G

8.1.4. Достаточно часто в приложениях встречаются важные с практической точки зрения подклассы остовных подграфов. По историческим причинам некоторые из них имеют специальные названия. Так, остовный 1-регулярный подграф носит называние 1-фактором графа G, а набор ребер в таком подграфе называется *совершенным паросочетанием* в исходном графе G. Остовный k-регулярный подграф для произвольного натурального $k \geqslant 1$ называется *k-фактором* графа G.

В качестве примера на рис.28 показаны графы, являющиеся 2-фактором (рис.28,a) и 1-фактором (рис.28,b) графа G, показанного на рис.26,a.

В случае мультиграфа G очень часто рассматривают простой граф, получающийся из исходного графа G удалением всех петель и заменой мультиребер на простые ребра. Такой граф, очевидно, является остовным подграфом исходного графа.

8.2. Вернемся к понятию связности в графах. Операции удаления ребра и вершины позволяют нам ввести важные понятия моста и точки сочленения в графе.



(a) Граф G_1 (b) Граф G_2

Рис. 29

8.2.1. Рассмотрим графы, показанные на рис.29. Видно, что связный граф G_1 (рис.29,a) перестает быть таковым после удаления ребра e, связывающего вершины x и y.

Определение 8.2. Ребро $e \in E(G)$ в связном графе G называется *мостом,* если получающийся после его удаления граф $G - e$ становится несвязным. В случае несвязного графа G мостом называется ребро, после удаления которого количество компонент связности увеличивается на единицу.

Теперь рассмотрим связный граф G, показанный на рис.29,b. У этого графа мостов нет, однако имеется вершина x, после удаления которой граф $G - x$ становится несвязным.

Определение 8.3. Вершина x называется точкой сочленения графа G, если после ее удаления количество компонент связности графа $G - x$ увеличивается по сравнению с количеством компонент связности исходного графа G.

Заметим, что даже в случае связного графа G удаление в нем точки сочленения может приводить к появлению как двух, так и большего количества связных компонент.

8.2.2. Докажем простое утверждение, характеризующее точку сочленения в связном графе G.

Утверждение 8.4. *Вершина x в связном графе G, построенном на $n \geqslant 3$ вершинах, есть точка сочленения графа G тогда и только тогда, когда в G существуют отличные от x вершины y и z, такие, что x содержится в каждом пути из y в z.*

Доказательство. Предположим вначале, что x есть точка сочленения в связном графе G. Предположим, что утверждение неверно, то есть предположим, что для любой пары вершин y, z найдется путь, не проходящий через x. Но тогда в графе $G - x$ любая пара вершин осталась бы связанной, что невозможно.

Теперь предположим, что в G существуют описанная в утверждении пара вершин y и z. Удаление вершины x в таком графе уничтожает любой путь из y в z, то есть превращает эти вершины в несвязанные. Это, в свою очередь, означает, что вершины y и z, лежащие ранее в одной компоненте связности графа G, стали принадлежать разным компонентам связности графа $G - x$. Иными словами, удаление x увеличивает количество связных компонент графа, то есть x есть точка сочленения в G. □

8.2.3. Теперь докажем наиболее важное утверждение, характеризующее мосты в графе G.

Утверждение 8.5. *Ребро $e = \{x, y\}$ в простом связном графе G является мостом тогда и только тогда, когда e не принадлежит ни одному из циклов графа G.*

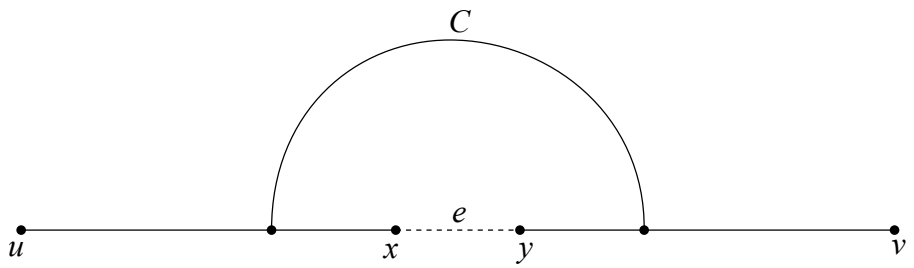


Рис. 30

Доказательство. Пусть ребро e является мостом в графе G. По определению, удаление e приводит к появлению в графе $G - e$ хотя бы одной пары $\{u, v\}$ несвязных между собой вершин (см.рис.30). В исходном графе G эти вершины связаны между собой простым путем $P = (u, v)$, который с необходимостью проходит через $e = \{x, y\}$. В графе $G - e$ вершины u и x связаны между собой участком $P_1 = (u, x)$ пути P, а вершины y и v — участком $P_2 = (y, v)$ этого пути. Если бы ребро e принадлежало циклу C в графе G, то x и y оказались бы связанными в графе $G - e$ путем $C - e$. Но тогда, по транзитивности, связанными бы оказались между собой и вершины u, v, что невозможно.

Обратно, предположим, что e не является мостом в графе G, то есть его удаление оставляет граф $G - e$ связным. Но тогда концы x и y ребра e в графе $G - e$ оказываются связанными некоторым xy-путем. Добавление к этому пути ребра e превращает этот путь в цикл. □

Замечание 8.6. Доказанные выше утверждения являются, по сути, аналогами вершинной и реберной теорем Менгера, описывающих k-связные графы. Изучению таких графов будет посвящена отдельная глава.

8.3. Мы ввели операции удаления ребра и вершины. Нам понадобятся еще несколько операций над графами.

8.3.1. Начнем с чуть более сложной по сравнению с удалением ребра операции — операции стягивания ребра.

(a) Исходный граф G (b) Граф $G \setminus e$

Рис. 31: Операция стягивания ребра

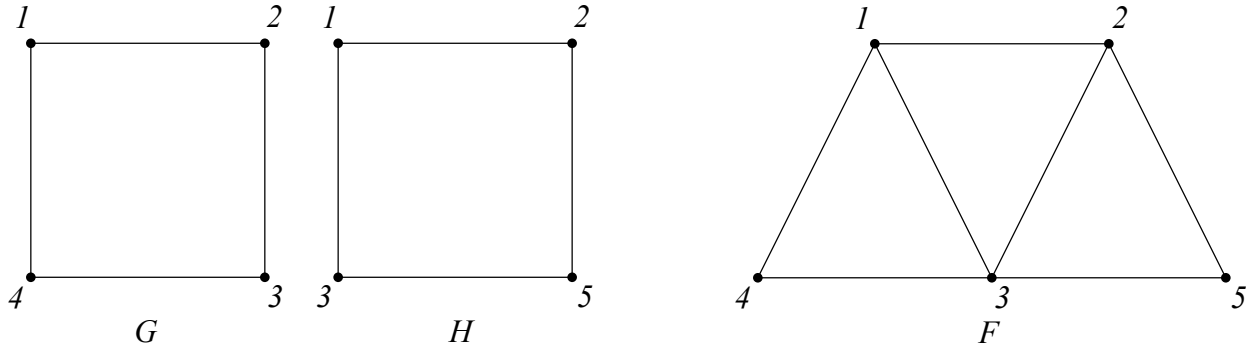
Определение 8.7. Говорят, что граф $G \setminus e$ получен из графа G стягиванием ребра e, если граф $G \setminus e$ получается из G удалением ребра e и стягиванием инцидентных e вершин x и y в одну (см.рис.31).

8.3.2. Следующие две полезные операции — это операции объединения и пересечения графов. Пусть G и H есть пара простых графов.

Определение 8.8. Объединением графов G и H называется граф $F = G \cup H$, множество вершин и множество ребер которого представляют собой объединение соответствующих множеств графов G и H:

$$V(F) = V(G) \cup V(H), \qquad E(F) = E(G) \cup E(H).$$

На рис.32,b в качестве примера показан граф F, полученный в результате объединения графов G и H, показанных на рис.32,a.



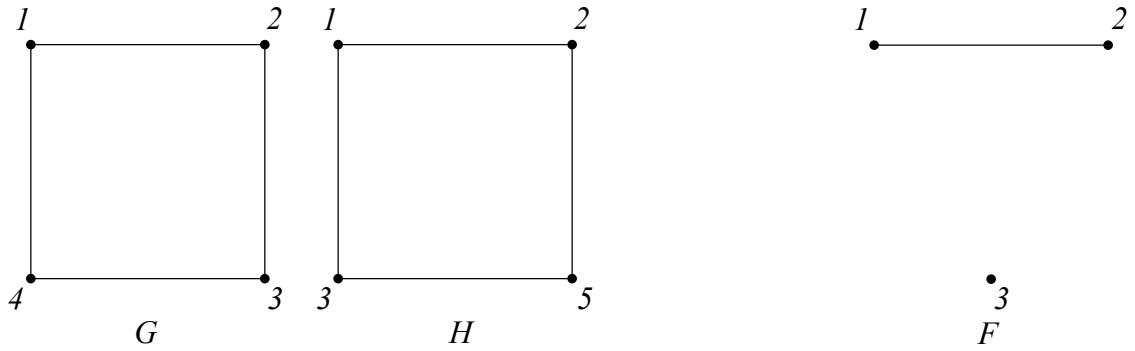
(a) (b)

Рис. 32: Операция объединения графов G и H

В случае, если множества вершин (а следовательно, и ребер) графов G и H не пересекаются, то мы получаем так называемое несвязное объединение $G + H$ таких графов. По сути, любой несвязный граф представляет собой несвязное объединение двух или более связных компонент.

Определение 8.9. Пересечением графов G и H называется граф $F = G \cap H$, множество вершин и множество ребер которого представляют собой пересечение соответствующих множеств графов G и H:

$$V(F) = V(G) \cap V(H), \qquad E(F) = E(G) \cap E(H).$$



(a) (b)

Рис. 33: Операция пересечения графов G и H

На рис.33,b приведен граф F, полученный в результате пересечения графов G и H, показанных на рис.33,a.

Заметим, что мы уже, по сути, встречались с операциями объединения и пересечения графов, когда говорили о графе G и о его дополнении — графе $\bar{G}$. Действительно, в частном случае $H = \bar{G}$ объединение графов G и $\bar{G}$ есть полный граф K_n на n вершинах, а пересечение графов G и $\bar{G}$ — это пустой граф $\bar{K}_n$, построенный на n изолированных вершинах.

8.3.3. При изучении остовных подграфов H_1 и H_2 одного и того же графа G достаточно часто встречается операция Δ их симметрической разности. С помощью такой операции мы из остовных подграфов H_1 и H_2 получаем остовный подграф $H = H_1 \Delta H_2$, состоящий из ребер, принадлежащих H_1, но не принадлежащих H_2, и наоборот.

Рассмотрим в качестве примера граф G, показанный на рис. 34. Два его остовных подграфа H_1 и H_2 показаны на рис. 35(a) и 35(b), а их симметрическая разность — остовный подграф

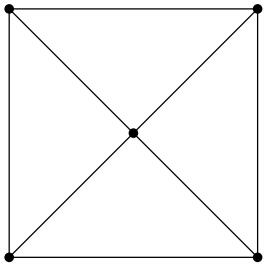
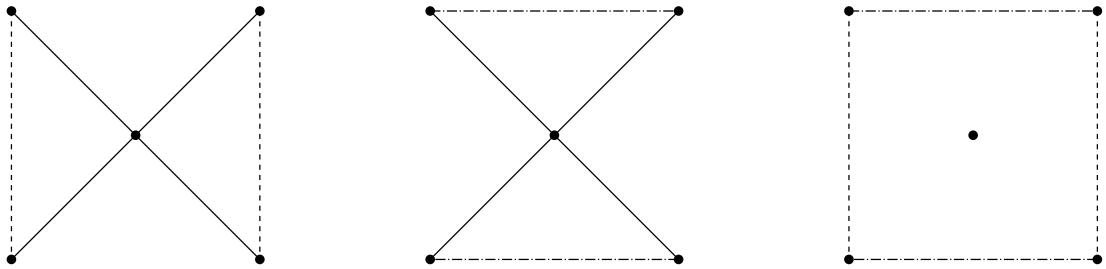


Рис. 34: Граф G

$H_3 = H_1 \Delta H_2$ — представлен на рис. 35(c). Общие для H_1 и H_2 ребра помечены на этих рисунках сплошными линиями, а ребра, встречающиеся в одном подграфе и не встречающиеся в другом, — пунктирными и штрихпунктирными линиями. Результирующий подграф H_3 состоит из ребер, помеченных пунктирными и штрихпунктирными линиями.



(a) (b) (c)

Рис. 35: Операция симметрической разности двух остовных подграфов

9 Изоморфизм и автоморфизм графов

9.1. Постараемся вначале ответить на вопрос, сколько же существует различных графов, построенных на n вершинах.

9.1.1. Сразу заметим, что в случае мультиграфов мы наряду с количеством вершин должны также фиксировать и количество ребер. Действительно, любые две вершины мультиграфа соединить произвольным количеством ребер, поэтому количество мультиграфов, построенных на n вершинах, без ограничения на количество ребер может быть сколь угодно большим. Поэтому мы ограничимся пока что задачей подсчета простых графов.

9.1.2. Количество g_n простых неориентированных графов достаточно легко сосчитать, используя формальное определение простого графа как некоторого подмножества множества $V^{(2)}$. Действительно, количество всех двухэлементных подмножеств n-элементного множества вершин (то есть мощность множества $V^{(2)}$) равно $|V^{(2)}| = \binom{n}{2}$. Нас же интересует множество Σ всех подмножеств $V^{(2)}$. Количество элементов в этом множестве, как известно, равно

$$|\Sigma| = 2^{|V^{(2)}|} = 2^{\binom{n}{2}}.$$

Следовательно, количество g_n всех простых графов на n вершинах равняется $2^{\binom{n}{2}}$.

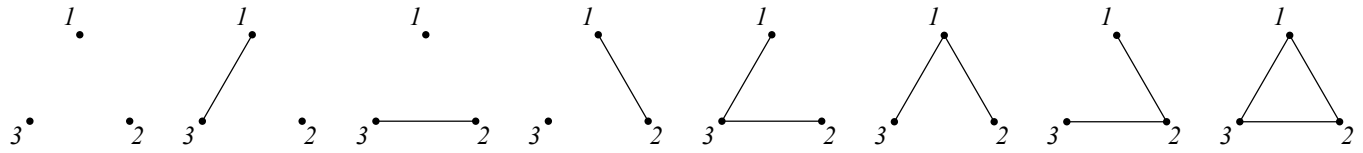


Рис. 36: Восемь простых графах на трех вершинах

Например, существует ровно $g_3 = 2^{\binom{3}{2}} = 2^3 = 8$ различных простых графов, построенных на трех вершинах (смотри рис.36).

9.1.3. Столь же просто подсчитать количество различных простых орграфов. Действительно, всего существует $n(n-1)$ упорядоченных пар отличных друг от друга вершин. Как следствие, количество различных простых орграфов, построенных на n вершинах, равно $2^{n(n-1)}$.

9.2. Заметим, что многие из приведенных на рис.36 графов похожи друг на друга в том смысле, что отличаются они друг от друга только перенумерацией вершин. Таковыми, в частности, являются три графа, имеющих только лишь одно ребро, а также три графа, имеющих ровно два ребра. Формализовать эту похожесть можно с помощью понятия изоморфизма графов.

9.2.1. Определим вначале понятие изоморфизма для простых неориентированных графов.

Определение 9.1. Говорят, что два простых графа G_1 и G_2 *изоморфны* друг другу, если существует взаимно-однозначное отображение $\varphi\colon V(G_1) \to V(G_2)$, которое сохраняет отношение смежности. Последнее означает, что если в графе G_1 некоторая пара вершин $\{x, y\}$ соединена ребром, то в графе G_2 соответствующая ей пара вершин $\{\varphi(x), \varphi(y)\}$ также соединена ребром, и наоборот.

Рис. 37: Два изоморфных графа

Пример 9.2. Рассмотрим графы G_1 и G_2, изображенные на рис.37. Они изоморфны друг другу, так как существует взаимно-однозначное отображение $\varphi\colon \{1,2,3\} \to \{1,2,3\}$ вида

$$\varphi(1) = 1, \qquad \varphi(2) = 3, \qquad \varphi(3) = 2,$$

при котором ребро $\{1,2\}$ графа G переходит в ребро $\{\varphi(1), \varphi(2)\} = \{1,3\} \in E(G_2)$, а ребро $\{2,3\}$ графа G переходит в ребро $\{\varphi(2), \varphi(3)\} = \{3,2\}$ графа G_2.

9.2.2. Как и любой изоморфизм, изоморфизм графов вводит отношение эквивалентности на множестве всех неориентированных простых графов на n вершинах. Отношение эквивалентности, в свою очередь, разбивает все множество таких графов на классы эквивалентности — классы изоморфных друг другу графов. В теории графов эти классы эквивалентности имеют специальное название.

Определение 9.3. Любой описанный выше класс эквивалентности графов называется *непомеченным* графом. В этом смысле обычный простой граф (то есть любой представитель класса эквивалентности) часто называется *помеченным* графом.

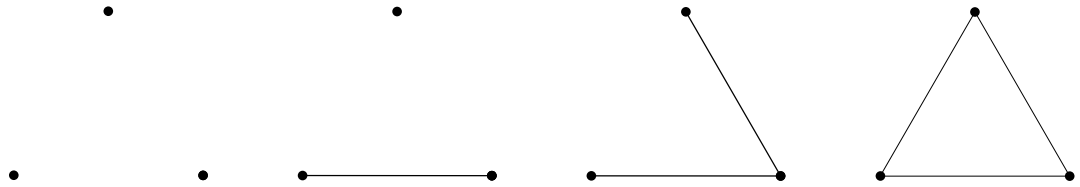


Рис. 38: Четыре простых непомеченных графа на трех вершинах

Так, из рис.36 видно, что существует четыре различных непомеченных графа на трех вершинах: один отвечает пустому графу, второй — трем помеченным графам с одним ребром, третий — трем помеченным графам с двумя ребрами, и наконец четвертый — полному графу K_3 (см.рис.38). Задача перечисления непомеченных графов на n вершинах для произвольного значения параметра n является значительно более сложной задачей по сравнению с задачей перечисления соответствующих помеченных графов.

9.2.3. Давайте теперь для полноты картины дадим определение изоморфизма для более общего случая неориентированных мультиграфов. Как мы помним, в общем случае любой неориентированный граф G задается тройкой, состоящей из множества $V(G)$ вершин, множества $E(G)$ ребер, а также отображения $I(G)$, сопоставляющего любому ребру $e \in E(G)$ неупорядоченную пару вершин $\{x, y\}$, которые это ребро e соединяет.

Определение 9.4. Говорят, что графы

$$G_1 = (V(G_1), E(G_1), I(G_1)) \qquad \text{и} \qquad G_2 = (V(G_2), E(G_2), I(G_2))$$

изоморфны друг другу, если существует пара взаимно-однозначных отображений

$$(\varphi, \vartheta), \quad \varphi\colon V(G_1) \to V(G_2), \qquad \text{и} \qquad \vartheta\colon E(G_1) \to E(G_2),$$

сохраняющих отношение смежности в графах G_1 и G_2. Последнее в данном случае означает, что если отображение $I(G_1)$ сопоставляет в графе G_1 ребру $e \in E(G_1)$ пару $\{x, y\}$, то в графе G_2 отображение $I(G_2)$ сопоставляет ребру $\vartheta(e) \in E(G_2)$ пару вершин $\{\varphi(x), \varphi(y)\}$, и наоборот.

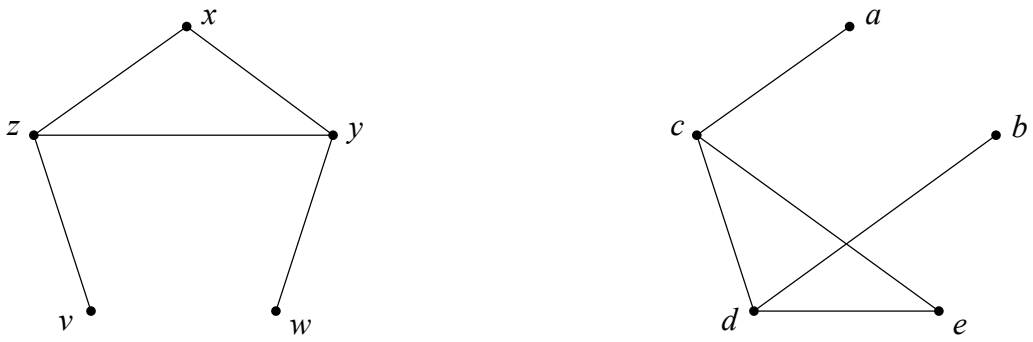
9.3. Часто на практике достаточно важно уметь отвечать на вопрос, являются ли два различных графа изоморфными друг другу или нет. Оказывается, проверка двух графов на изоморфизм является в общем случае довольно-таки нетривиальной задачей.

9.3.1. Начнем вначале с простого примера.

Пример 9.5. Рассмотрим два графа, изображенных на рис.39. Возникает вопрос, являются ли два таких графа изоморфными друг другу.

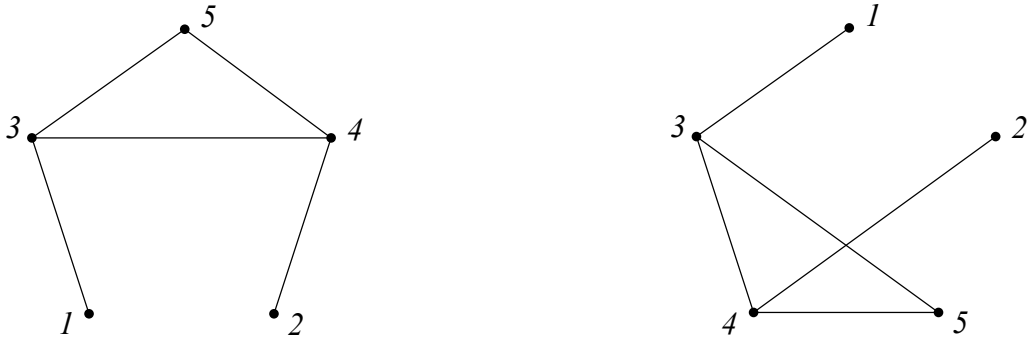
Для ответа на этот вопрос нам нужно, вообще говоря, найти взаимно-однозначное отображение φ из множества вершин $V(G_1)$ в множество $V(G_2)$, сохраняющее отношение смежности. Легче всего в таких простых задачах действовать следующим образом: нужно удалить пометки вершин у первого и у второго графов, пометить вершины первого графа числами от 1 до n, а затем попытаться разметить вершины второго графа теми же числами от 1 до n так, чтобы такой изоморфизм φ был бы очевиден.

В данном конкретном примере сделать это довольно просто (см.рис.40): у графов G_1 и G_2 имеются всего лишь две вершины, степень которых равна единице. В графе G_1 они помечены



(a) Граф G_1 (b) Граф G_2

Рис. 39: Установление изоморфизма между графами



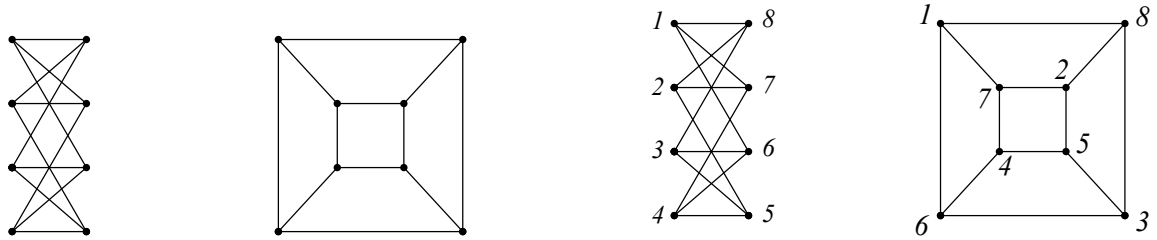
(a) Граф G_1 (b) Граф G_2

Рис. 40: Установление изоморфизма между графами

цифрами 1 и 2. Поэтому нам будет удобно теми же самыми цифрами пометить одновалентные вершины и в графе G_2. Далее, в графе G_1 эти вершины соединены с вершинами 3 и 4, имеющими степень, равную трем. Пометим аналогичные вершины в G_2 теми же цифрами. Нам остается пометить в графе G_2 оставшуюся вершину цифрой 5 и убедиться, что отображение $\varphi(i) = i$ задает изоморфизм этих графов.

Пример 9.6. Рассмотрим теперь графы, показанные на рис.41,a. Для таких графов уже не вполне очевидно то, что они изображают один и тот же непомеченный граф. Для доказательства этого факта можно, например, попытаться разметить вершины двух графов так, чтобы пара вершин $\{x, y\}$, связанная ребром в графе G_1, оказалась бы связанной ребром и в графе G_2, и наоборот. В данном примере это можно все же сделать достаточно просто, заметив, что граф G_2 состоит из двух непересекающихся циклов длины 4. Соответственно, выделив в графе G_1 замкнутый цикл аналогичной длины, мы можем уже однозначно пометить оставшиеся вершины графа G_1 так, чтобы списки смежности этих графов совпали (смотри рис.41,b).

9.3.2. В практических задачах довольно часто встречаются графы, в которых содержатся сотни тысяч вершин. Для таких графов никакие визуальные методы оценки уже не работают. Не существует пока и никаких достаточно простых критериев, позволяющих достаточно быстро ответить на вопрос, являются ли два заданных графа изоморфными или нет. Конечно же, есть очевидные необходимые условия изоморфности двух графов — так, например, у изоморфных графов последовательности степеней их вершин должны, очевидно, совпадать. Но в случае, если подобного рода очевидные условия выполнены, то не остается ничего другого, как проверять заданную пару графов на изоморфность простым перебором.



(a) (b)

Рис. 41: Установление изоморфизма между графами

Для этого вершины обоих графов помечают числами из одного и того же множества $[n]$, затем разметку первого графа фиксируют, а пометки второго графа начинают переставлять, вообще говоря, всеми $n!$ способами. Если на каком-то шаге матрицы смежности или же списки смежности этих двух графов совпадут, то графы являются изоморфными друг другу. В противном случае они неизоморфны.

Такого рода алгоритмы крайне неэффективны хотя бы потому, что при больших n значение $n!$ крайне велико. Существуют достаточно сложные алгоритмы, позволяющие более эффективно проверять на изоморфизм специальные подклассы графов, однако никакого быстрого алгоритма, позволяющего делать это в общем случае, пока не существует.

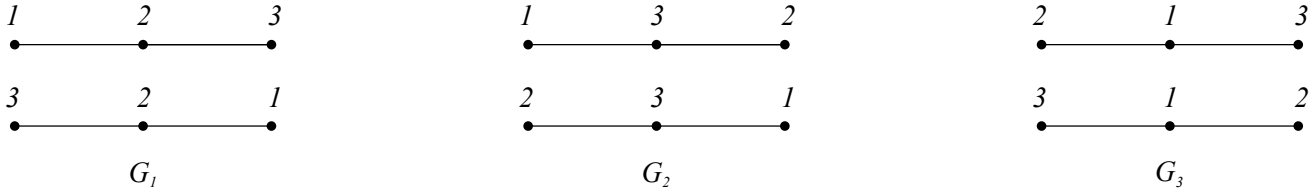


Рис. 42

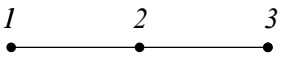
9.4. Наряду с изоморфизмом, в теории графов вводится также понятие автоморфизма — изоморфизма графа G в себя.

9.4.1. Прежде чем давать формальное определение автоморфизма, вернемся еще раз к очень важным понятиям непомеченных и помеченных графов. Давайте подумаем, сколькими способами мы можем разметить один и тот же непомеченный граф на n вершинах. Очевидно, что мы это можем сделать $n!$ способами. Однако при этом возникает вопрос — действительно ли все такие способы пометки дадут нам *различные* помеченные графы?

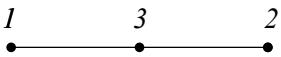
В качестве примера рассмотрим простой непомеченный граф на трех вершинах, имеющий два ребра (рис.42,a). Пометим для него 3! способами его вершины. В результате мы получим 6 помеченных графов, изображенных на рис.42,b. Однако мы с вами уже перебрали все помеченные графы, построенные на трех вершинах, и выяснили, что существует только лишь три *различных* помеченных графа, имеющих два ребра. Следовательно, среди шести графов, показанных на рис.42,b, половина лишняя в том смысле, что среди этих графов имеются графы, представляющие собой один и тот же помеченный граф.

Несложно, в частности, убедиться в том, что графы G_1 и G_3, показанные на рис.42,b, представляют собой один и тот же помеченный граф — первый из них переходит во второй при его развороте на 180°. То же касается и графов G_2 и G_6, а также графов G_4 и G_5.

9.4.2. Давайте теперь постараемся разобраться с причиной того, что при разметке вершин непомеченного графа $n!$ способами мы получаем лишние графы. Для этого возьмем непомеченный граф $\widetilde{G}$, как-то разметим его вершины (например, так, как это показано на рис.43,a), а затем рассмотрим поподробнее все функции $\varphi\colon [3] \to [3]$, задающие перестановки вершин полученного таким образом помеченного графа G_1.



(a) Граф G_1



(b) Граф G_2

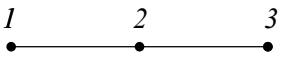
Рис. 43: Изоморфные помеченные графы на трех вершинах

Рассмотрим вначале функцию φ_1, оставляющую единицу на месте и меняющую местами вершины 2 и 3:

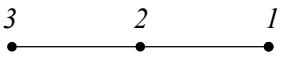
$$\varphi_1(1) = 1, \qquad \varphi_1(2) = 3, \qquad \varphi_1(3) = 2.$$

В результате мы из графа G_1 получим помеченный граф G_2 (рис.43,b). Естественно, что графы G_1 и G_2 изоморфны друг другу, так как они получены из одного и того же непомеченного графа. Однако важно то, что это — два различных графа: списки смежности этих двух графов различны:

$$G_1: \begin{array}{ll} 1 & \text{смежна с 2} \\ 2 & \text{смежна с 1 и 3;} \\ 3 & \text{смежна с 2} \end{array} \qquad G_2: \begin{array}{ll} 1 & \text{смежна с 3} \\ 2 & \text{смежна с 3} \\ 3 & \text{смежна с 1 и 2.} \end{array}$$



(a) Граф G_1



(b) Граф G_3

Рис. 44: Идентичные помеченные графы на трех вершинах

Рассмотрим теперь перестановку φ_2 вершин графа G_1 вида

$$\varphi_2(1) = 3, \qquad \varphi_2(2) = 2, \qquad \varphi_2(3) = 1.$$

В результате такой перестановки мы из графа G_1 (рис.44,a) получим граф G_3 (рис.44,b). Важно, что у такого графа G_3 список смежности полностью совпадает со списком смежности исходного графа. Следовательно, в результате подобной перестановки φ_2 мы получаем тот же самый помеченный граф, что у нас был до перестановки. Такая перестановка вершин, при которой граф как помеченный объект переходит сам в себя, как раз и называется *автоморфизмом* графа.

Более формальное определение автоморфизма графа звучит следующим образом.

Определение 9.7. Взаимно-однозначное отображение $\varphi : V(G) \to V(G)$ множества $V(G)$ вершин простого графа G в себя называется автоморфизмом графа G, если она сохраняет отношение смежности *в графе* G. Последнее означает, что если вершины x и y соединены в графе G ребром $\{x, y\}$, то в том же самом графе G и образы $\varphi(x)$, $\varphi(y)$ этих вершин соединены между собой ребром.

9.4.3. Давайте еще раз сравним определения изоморфизма и автоморфизма графов. На первый взгляд эти определения очень похожи друг на друга, и сразу не очень понятно, чем они

отличаются друг от друга. Казалось бы, отличие этих определений состоит в том, что в определении изоморфизма фигурируют различные множества $V_1(G_1)$ и $V_2(G_2)$ вершин, тогда как в определении автоморфизма эти множества совпадают. Однако это отличие в двух определениях абсолютно непринципиально. Как правило, вершины любого графа помечаются элементами одного и того же множества $[n] = \{1, 2, \ldots, n\}$ первых n натуральных чисел, так что и в определении изоморфизма, и в определении автоморфизма в роли отображения φ обычно выступает какая-то перестановка $\sigma\colon [n] \to [n]$ первых n натуральных чисел.

Ключевое же отличие этих двух определений заключается в следующем. В определении автоморфизма мы в результате отображения φ получаем *тот же самый* граф G, тогда как в случае изоморфизма, не являющегося автоморфизмом, граф G_1 под действием отображения φ переходит в некоторый *другой* граф G_2, отличный от исходного графа G_1, причем это верно даже в том случае, когда множества $V_1(G_1)$ и $V_2(G_2)$ вершин этих двух графов совпадают. Легче всего это отличие описать в терминах матрицы или списка смежности графов: в случае автоморфизма список смежности графа G не изменится, тогда как в результате действия изоморфизма списки смежности графов G_1 и G_2 будут, вообще говоря, отличаться друг от друга.

9.4.4. Рассмотрим теперь множество всех автоморфизмов какого-то простого помеченного графа G. Очевидно, что тождественная перестановка id принадлежит этому множеству. Далее, для любого автоморфизма φ обратный к нему автоморфизм φ^{-1} также не меняет граф G, то есть принадлежит тому же множеству автоморфизмов. Наконец, если φ_1 и φ_2 являются автоморфизмами графа, то и их композиция задает нам некоторый автоморфизм. Иными словами, операция композиции автоморфизмов задает на множестве всех автоморфизмов группу $\mathrm{Aut}(G)$, которая называется *группой автоморфизмов* помеченного графа G.

Так, в разобранном выше примере группа автоморфизмов помеченного графа G_1 состоит из двух перестановок — тождественной перестановки id, а также перестановки, меняющей местами вершины 1 и 3. Иными словами, эта группа изоморфна группе $\mathbb{Z}_2$.

9.4.5. Несмотря на то, что группа автоморфизмов определена для помеченного графа G, она, на самом-то деле, отражает внутреннюю симметрию исходного непомеченного графа $\widetilde{G}$. Так, например, мы видим, что показанный на рис.42,а непомеченный граф $\widetilde{G}$ на трех вершинах симметричен относительно поворота на 180°. Проблема состоит лишь в том, что не очень понятно, как у непомеченного графа такие симметрии искать. И здесь нам как раз и приходит на помощь помеченный граф: мы берем непомеченный граф $\widetilde{G}$, как-то размечаем его вершины, получаем помеченный граф G, вычисляем для него группу автоморфизмов $\mathrm{Aut}(G)$, а затем говорим, что эта группа и отражает внутреннюю симметрию исходного непомеченного графа $\widetilde{G}$.

9.4.6. После сделанных замечаний мы можем вернуться к проблеме, с которой мы начинали — к определению количества различных помеченных графов.

Утверждение 9.8. *Количество N различных помеченных графов, получаемых из непомеченного графа $\widetilde{G}$ разметкой его вершин, рассчитывается по формуле*

$$N = \frac{n!}{|\operatorname{Aut}(G_1)|}, \tag{41}$$

где G_1 — произвольно выбранный помеченный граф, полученный из $\widetilde{G}$ некоторой разметкой его вершин.

Доказательство. Выберем произвольный помеченный граф G_1 и сосчитаем для него группу автоморфизмов $\mathrm{Aut}(G_1)$. Предположим вначале, что порядок этой группы совпадает с $n!$. Это

означает, что любая перестановка вершин переводит граф G_1 в себя, то есть у нас получается лишь один помеченный граф, отвечающий исходному непомеченному графу $\widetilde{G}$. В этом случае формула (41) верна — она дает $n!/n! = 1$, то есть единственный помеченный граф.

Теперь предположим, что $|\operatorname{Aut}(G_1)| = k$, $k < n!$. В этом случае среди всех $n!$ перестановок вершин найдется перестановка $\widetilde{\varphi}$, не принадлежащая $\operatorname{Aut}(G_1)$. Иными словами, эта перестановка переведет нам граф G_1 в некоторый изоморфный, но отличный от G_1 граф G_2. Применим тогда к графу G_1 все перестановки вида

$$\widetilde{\varphi} \cdot \varphi_i, \qquad \varphi_i \in \operatorname{Aut}(G_1).$$

Количество таких перестановок совпадает, очевидно, с мощностью $|\operatorname{Aut}(G_1)| = k$ группы автоморфизмов графа G_1. Действительно, если $\widetilde{\varphi} \cdot \varphi_i = \widetilde{\varphi} \cdot \varphi_j$, то и $\varphi_i = \varphi_j$, поэтому для $\varphi_i \neq \varphi_j$ мы получаем и две различные перестановки вида $\widetilde{\varphi} \cdot \varphi_i$ и $\widetilde{\varphi} \cdot \varphi_j$. При этом все такие перестановки переведут граф G_1 в граф G_2: вначале мы с помощью перестановки φ_i переводим граф G_1 в себя, а затем с помощью $\widetilde{\varphi}$ из графа G_1 получаем граф G_2.

Предположим теперь, что у нас нашлась еще какая-то перестановка $\widehat{\varphi}$, отличная от $2k$ уже найденных перестановок и переводящая G_1 в некоторый граф G_3. Тогда все перестановки вида $\widehat{\varphi} \cdot \varphi$, $\varphi \in \operatorname{Aut}(G_1)$ переводят G_1 в G_3. При этом, так как $\widehat{\varphi} \notin \operatorname{Aut}(G_1)$, то G_3 отличен от G_1. Покажем, что этот граф отличен и от G_2. Действительно, если бы перестановки $\widetilde{\varphi} \cdot \varphi_i$ и $\widehat{\varphi} \cdot \varphi_j$ переводили граф G_1 в один и тот же граф G_2, то мы бы имели равенства вида

$$\widehat{\varphi} \cdot \varphi_j = \widetilde{\varphi} \cdot \varphi_i \qquad \Longleftrightarrow \qquad \widehat{\varphi} = \widetilde{\varphi} \cdot \varphi_i \cdot \varphi_j^{-1} = \widetilde{\varphi} \cdot \varphi_k, \qquad \varphi_k \in \operatorname{Aut}(G_1),$$

а это невозможно — мы предположили, что $\widehat{\varphi}$ отлична от перестановок вида $\widetilde{\varphi} \cdot \varphi_k$.

Продолжая указанный процесс, мы в конце концов исчерпаем все множество Φ перестановок. При этом все это множество у нас окажется разбитым на N блоков, каждый из которых имеет мощность k. Каждый такой блок соответствует перестановкам, переводящим граф G_1 в некоторый изоморфный, но отличный от G_1 помеченный граф G_i. Общее же количество N различных помеченных графов связано с количеством $n!$ всех перестановок множества $V(G_1)$ вершин графа G_1 формулой

$$N \cdot k = N \cdot |\operatorname{Aut}(G_1)| = n!,$$

из которой и следует формула (41). □

(a) Граф G_1 (b) Граф G_4

Рис. 45: Идентичные помеченные графы на трех вершинах

9.4.7. Переведем проведенные выше рассуждения на язык теории групп. Для этого рассмотрим множество $\mathcal{X}$ всех помеченных графов, построенных на n-множестве $\{1, 2, \ldots, n\}$ вершин, а также действие группы S_n перестановок на этом множестве $\mathcal{X}$. Действие перестановки σ на помеченный граф G при этом определим следующим образом: любая пометка i вершины графа переходит под действием перестановки σ в пометку $\sigma(i)$. В качестве примера рассмотрим граф G_1 на рис.45,a, а также перестановку $\sigma = (1, 2, 3)$. В этом случае пометка 1 вершины графа G_1 переходит под действием перестановки σ в пометку $\sigma(1) = 2$, пометка 2 — в пометку $\sigma(2) = 3$, а пометка 3 — в пометку $\sigma(3) = 1$. В результате такого действия мы получим граф G_4, показанный на рис.45,b.

Говорят, что графы G_1 и G_2 изоморфны друг другу, если существует такая перестановка σ множества вершин, которая переводит граф G_1 в граф $G_2 = \sigma \circ G_1$, то есть такая перестановка, которая любое ребро $\{i, j\}$ в графе G_1 переводит в ребро $\{\sigma(i), \sigma(j)\}$. На языке теории групп это означает, что эти графы эквивалентны относительно действия группы S_n, то есть принадлежат одной и той же *орбите*

$$S_n \circ G = \{\sigma \circ G \colon \sigma \in S_n\}$$

элемента G множества $\mathcal{X}$ всех графов, построенных на n вершинах, под действием группы S_n. Каждая такая орбита на языке теории графов представляет собой некоторый непомеченный граф на n-множестве вершин.

Рассмотрим теперь подмножество $\operatorname{Aut}(G)$ перестановок σ, любой элемент которого оставляет некоторый помеченный граф $G \in \mathcal{X}$ неподвижным:

$$\operatorname{Aut}(G) := \{\sigma \in S_n \colon \sigma \circ G = G\}.$$

Несложно понять, что такое подмножество перестановок образует подгруппу $\operatorname{Aut}(G)$ группы S_n перестановок. Такая подгруппа на языке теории групп называется *стабилизатором* элемента G множества $\mathcal{X}$. Эта подгруппа (как и любая подгруппа H группы S_n) задает разбиение группы S_n всех перестановок на блоки $\sigma \cdot \operatorname{Aut}(G)$, называемые смежными классами группы S_n по подгруппе $\operatorname{Aut}(G)$. Размер этих блоков одинаков и равен порядку $|\operatorname{Aut}(G)|$ группы $\operatorname{Aut}(G)$. Следовательно, порядок $|S_n|$ всей группы перестановок равен произведению порядка $|\operatorname{Aut}(G)|$ группы автоморфизмов графа G на количество смежных классов $S_n/\operatorname{Aut}(G)$ группы S_n по подгруппе $\operatorname{Aut}(G)$ (теорема Лагранжа):

$$|S_n| = n! = |\operatorname{Aut}(G)| \cdot |S_n/\operatorname{Aut}(G)|.$$

В рассмотренном нами примере мы разбили множество S_3 всех перестановок вершин графа G_1 на блоки, состоящие из перестановок вида $\widetilde{\varphi} \cdot \varphi$, где $\varphi \in \operatorname{Aut}(G_1)$, и показали, что размер каждого блока равен двум — порядку группы $\operatorname{Aut}(G_1)$.

Затем мы заметили, что любому такому блоку перестановок отвечает некоторый граф G_i, изоморфный исходному графу G. На языке теории групп данный факт формулируется в виде достаточно важного, и в то же время достаточно просто доказываемого утверждения — так называемой Orbit-Stabiliser Theorem. Согласно этой теореме, имеется взаимно-однозначное соответствие между всеми элементами орбиты $S_n \circ G$ графа G (то есть всеми изоморфными G графами на n вершинах) и фактор-множеством $S_n/\operatorname{Aut}(G)$, то есть множеством всех смежных классов группы S_n по стабилизатору $\operatorname{Aut}(G)$. Следствием этого утверждения как раз и является равенство

$$|S_n| = n! = |\operatorname{Aut}(G)| \cdot |S_n \circ G| = |\operatorname{Aut}(G)| \cdot N,$$

где N — количество графов на n вершинах, изоморфных G.

9.4.8. Дадим еще одно важное определение.

Определение 9.9. Граф G называется ассиметричным, если его группа автоморфизмов тривиальна, то есть состоит только из тождественной перестановки множества $[n]$ его вершин.

Важно заметить, что чем выше n, тем большее количество графов асимметрично. Следовательно, при больших n количество $\mathcal{M}_n$ непомеченных графов можно оценить по формуле

$$\mathcal{M}_n \sim \frac{2^{\binom{n}{2}}}{n!}.$$

10 Основные свойства деревьев

10.1. Одним из самых важных понятий теории графов является понятие дерева.

10.1.1. Начнем с формального определения этого понятия.

Определение 10.1. Деревом называется простой связный граф без циклов. Произвольный (не обязательно связный) простой граф без циклов называется лесом.

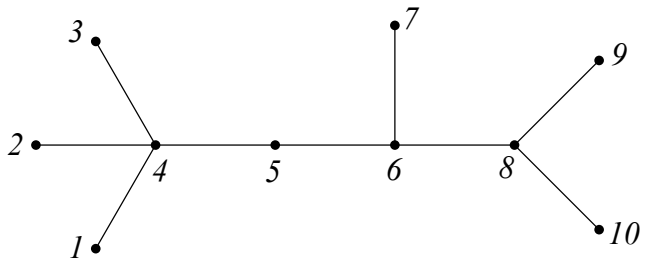


Рис. 46: Пример дерева на десяти вершинах

На рис.46 показан пример дерева T, построенного на 10 вершинах.

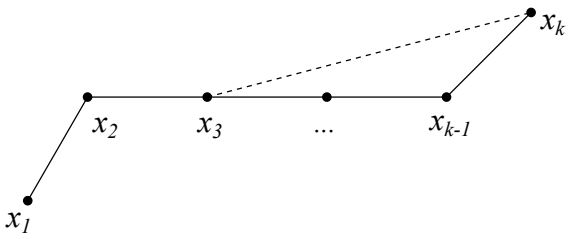
10.1.2. Изучим простейшие свойства деревьев. Начнем со следующего важного определения.

Определение 10.2. Вершина графа G, имеющая единичную степень, называется листом.

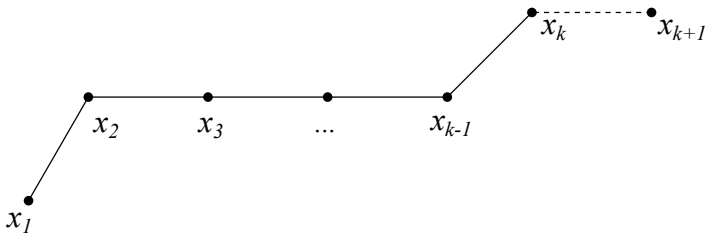
У дерева T, показанного на рис.46, имеется шесть листьев — вершины 1, 2, 3, 7, 9 и 10.

Лемма 10.3. *У любого дерева T, построенного на $n \geqslant 2$ вершинах, имеется как минимум два листа.*

Доказательство. Рассмотрим произвольный простой путь $P = (x_1, x_2, \ldots, x_k)$ максимальной длины в T (рис.47). Для изображенного на рис.46 дерева таким путем является, например, путь $P = (1, 4, 5, 6, 8, 9)$. Очевидно, что такой путь в T всегда существует, причем его длина k равна диаметру $\mathrm{diam}(T)$ дерева T. Покажем, что концы этого пути – вершины x_1 и x_k – обязаны быть листьями.



(a) Граф G



(b) Граф G'

Рис. 47

Предположим, что это не так, то есть предположим, например, что степень вершины x_k больше или равна двум. Это означает, что из вершины x_k помимо ребра $\{x_{k-1}, x_k\}$ исходит еще хотя бы одно ребро. Всего имеет место два варианта развития событий. В первом из них исходящее из x_k ребро приходит в одну из вершин пути P, отличную от x_{k-1} (рис.47,a). Но в этом случае в дереве

T появляется цикл, а это невозможно. Во втором варианте инцидентное x_k ребро приходит в какую-то вершину x_{k+1}, отличную от вершин пути P (рис.47,b). Но тогда в графе G имеется путь $(x_1, x_2, \ldots, x_k, x_{k+1})$, длина которого на единицу больше длины исходного пути P. А это, в свою очередь, противоречит максимальности пути P. Полученное противоречие доказывает наше утверждение. □

10.1.3. Из доказанной леммы достаточно легко следует одно из основных свойств дерева T.

Теорема 10.4. *В дереве T, построенном на n вершинах, имеется ровно $(n-1)$ ребро:*

$$|E| = |V| - 1 = n - 1.$$

Доказательство проведем индукцией по количеству n вершин в графе. База индукции очевидна — дерево, состоящее из $n = 1$ вершины, не имеет ни одного ребра. Предположим теперь, что утверждение доказано для деревьев, построенных на n вершинах, и покажем, что оно остается верным для произвольного дерева с $(n+1)$-й вершиной.

Действительно, по доказанной выше лемме 10.3 у любого такого дерева имеется хотя бы один лист x. Удалим теперь этот лист x вместе с инцидентным ему ребром e. Полученный в результате такой операции граф T' останется, очевидно, связным. Действительно, любой путь, соединяющий в T произвольные две вершины, отличные от x, ребро e содержать не может. Поэтому удаление e на связанность оставшихся в T' вершин никак не повлияет. Наконец, от удаления ребра e никаких дополнительных циклов также появиться не может. Следовательно, граф T' является деревом, построенным на n вершинах. Но у такого дерева по индукционному предположению имеется $(n-1)$ ребро. Следовательно, у исходного дерева T имеется ровно n ребер. □

Достаточно очевидно и обратное утверждение.

Теорема 10.5. *Любой простой связный граф G, построенный на n вершинах и имеющий $(n-1)$ ребро, является деревом.*

Доказательство. Действительно, пусть это не так. Тогда в графе G имеется цикл C. В параграфе, посвященном основным операциям над графами, мы ввели важное понятие моста как ребра, после удаления которого простой связный граф теряет связность. В том же параграфе мы предъявили важную характеризацию моста — мы сказали, что мостом в графе G является ребро, не принадлежащее никакому циклу в графе. Это утверждение можно еще переформулировать так: любое ребро e, принадлежащее циклу C, мостом в графе G не является. Следовательно, удаление такого ребра не приводит к потере связности графа G.

Рассмотрим тогда простой связный граф $G_1 = G - e$. Если в нем остались циклы, мы выберем один из них и удалим любое ребро, принадлежащее этому циклу. Продолжая далее, мы на каком-то шаге получим, наконец, простой связный граф без циклов, то есть дерево. Но мы только что доказали, что в дереве на n вершинах имеется ровно $n-1$ ребро. Следовательно, нам для получения дерева никаких ребер удалять не пришлось, то есть исходный граф G изначально представлял собой дерево. □

Следствие 10.6. *Всякий связный граф, построенный на n вершинах, имеет как минимум $(n-1)$ ребро.*

Доказательство этого утверждения абсолютно аналогично доказательству теоремы 10.5. Действительно, рассмотрим произвольный связный граф. Если такой граф не является деревом, то у него имеется хотя бы один цикл. В этом цикле мы всегда можем удалить одно ребро, и

граф при этом останется связным. Будем продолжать этот процесс до тех пор, пока в графе не останется ни одного цикла. В результате получим простой связный граф без циклов, то есть дерево. Количество ребер у дерева равно $(n-1)$. Следовательно, у исходного графа имеется как минимум $(n-1)$ ребро. □

10.1.4. Следствие 10.6 по-другому можно переформулировать так: у всякого связного графа существует остовное дерево. Получить это дерево можно, либо удаляя лишние ребра, либо (что более рационально) используя так называемый алгоритм поиска в глубину.

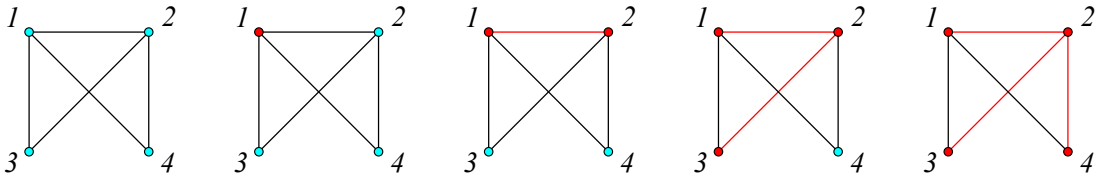


Рис. 48: Алгоритм поиска в глубину

В простейшем своем исполнении этот алгоритм выглядит так. Пусть у нас имеется связный граф G. Будем считать, что все его вершины и ребра окрашены в синий цвет. Выберем произвольную вершину x этого графа и окрасим ее в красный цвет. Затем начнем перебирать все инцидентные x ребра. В случае, если ребро e соединяет x с еще не окрашенной вершиной y, окрашиваем ребро $e = \{x, y\}$ в красный цвет и рекурсивно повторяем процесс, запуская алгоритм поиска в глубину с вершины y. Если для текущей вершины z все смежные с ней вершины оказываются окрашенными в красный цвет, то алгоритм поиска в глубину для нее заканчивается и мы переходим на более высокий уровень рекурсии.

На рис.48 приведено пошаговое выполнение алгоритма на примере графа G, построенного на четырех вершинах. Как видно, в результате работы алгоритма поиска в глубину все вершины перекрашиваются в красный цвет. Красным же цветом оказываются помечены и ребра, причем красные вершины и ребра представляют собой остовное дереве $T(G)$ графа G. Действительно, по окончании работы алгоритма мы получаем окрашенный в красный цвет связный подграф, количество ребер в котором на единицу меньше количества вершин, а мы только что доказали, что такой подграф обязательно является деревом.

В чуть более общем виде несвязного графа G псевдокод алгоритма поиска в глубину можно записать так:

```
Процедура DFS(x), x - вершина графа G:
  t := t + 1;
  pre[x] := time;
  color[x] := 1;
  for ( y : (x,y) принадлежит E(G) )
    begin
      if ( color[y] == 0 )
        begin
          parent[y] := x;
          DFS(y);
        end
    end
```

```
  t := t + 1;
  post[x] := time;
  color[x] := 2;

Основная программа:
 // Инициализация начальных значений
  for ( x : x - вершина графа G )
    begin
      color[x] := 0;
      parent[x] := -1;
      pre[x] := 0;
      post[x] := 0;
    end
  t := 0;
 // Запуск основного алгоритма
  for ( x : x - вершина графа G )
    begin
      if ( color[x] == 0 )
        begin
          DFS(x);
        end
    end
```

В этом варианте алгоритма вместо двух цветов вершин мы используем три — цвет 0 (голубой цвет на рис.49) для еще необнаруженной вершины, цвет 1 (желтый цвет на рис.49) для обнаруженной, но не до конца обработанной вершины (то есть для вершины, у которой есть смежные с ней вершины, окрашенные в синий цвет), а также цвет 2 (красный цвет на рисунке) для обнаруженной и обработанной вершины. Кроме того, мы вводим в алгоритм параметр t — время, а также массивы `pre[]` и `post[]`, отмечающие время захода и выхода в конкретную вершину x. Так, время `pre[2]` захода в вершину 2 для алгоритма, проиллюстрированного на рис.49, равно двум, а время `post[2]` выхода равно семи. В массиве `parent[]` у нас по результатам работы алгоритма для каждой вершины x, кроме начальной вершины для каждой компоненты связности графа G, хранится вершина, являющаяся предшественником вершины x при обходе графа G в глубину.

Алгоритм поиска в глубину очень часто используется в разнообразных приложениях, связанных с теорией графов. Например, с помощью этого алгоритма можно найти все компоненты связности неориентированного графа, осуществить топологическую сортировку вершин ориентированного ациклического графа, найти мосты и/или точки сочленения в связном графе и так далее.

10.1.5. Отметим еще две важные характеризации дерева. Первая из них описывает ребра дерева.

Утверждение 10.7. *Связный граф T является деревом тогда и только тогда, когда любое его ребро является мостом.*

Доказательство практически очевидно. Действительно, по определению, в дереве T циклы отсутствуют. Следовательно, ни одно из его ребер циклу не принадлежит. Согласно основной характеризации моста это и означает, что любое ребро графа T является мостом. Обратно,

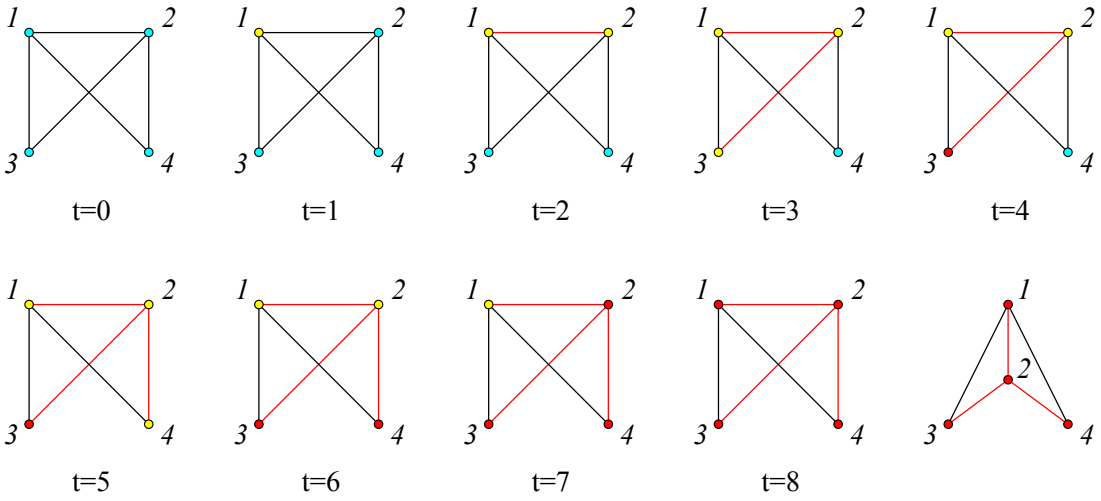


Рис. 49: Алгоритм поиска в глубину

если любое ребро связного графа является мостом, то оно не принадлежит никакому циклу. Поэтому данный граф обязан быть деревом. □

Вторая характеризация связана с понятием связанности в графе. Мы знаем, что в любом связном графе любые две вершины связаны по крайней мере одним путем. Дерево — это связный граф, в котором для любых двух вершин такой путь единственен.

Утверждение 10.8. *Граф T является деревом тогда и только тогда, когда для любых двух его вершин существует единственный простой путь, соединяющий эти вершины.*

Доказательство. Пусть в T для любой пары вершин существует единственный соединяющий их простой путь. Тогда T является связным графом. Если бы в таком графе существовал цикл C, то между любыми двумя вершинами этого цикла существовало бы два различных пути, соединяющие эти вершины, а это невозможно.

Обратно, пусть T является деревом, и пусть в нем существуют два различных простых пути P и Q, соединяющих какие-то две вершины x и y. Рассмотрим симметрическую разность $P\Delta Q$ этих путей. Подграф $P\Delta Q$ состоит из одного или нескольких циклов, а это противоречит тому, что T является деревом. □

Следствие 10.9. *Пусть T есть остовное дерево связного графа G, и пусть e есть ребро, не принадлежащее T. Тогда граф $T + e$ содержит единственный цикл, причем этот цикл проходит через ребро e.*

Доказательство. Заметим, прежде всего, что любой цикл в графе $T + e$ обязан содержать ребро $e = \{x, y\}$ — в противном случае мы бы получили цикл и в дереве T, чего быть не может. Кроме того, C есть цикл в $T + e$ тогда и только тогда, когда $C - e$ есть xy-путь в исходном дереве T. Но в дереве T xy-путь единственен. Следовательно, $T + e$ содержит единственный цикл, проходящий через ребро e. □

10.1.6. Рассмотрим теперь простой связный граф G, в котором имеется хотя бы одна пара отличных друг от друга остовных деревьев T и T'.

Утверждение 10.10. *Рассмотрим два различных остовных дерева T и T' простого связного графа G. Пусть e есть ребро остовного дерева T, не принадлежащее T'. Тогда в остовном дереве T' найдется ребро e', не принадлежащее остовному дереву T, такое, что граф $T - e + e'$ является остовным деревом графа G.*

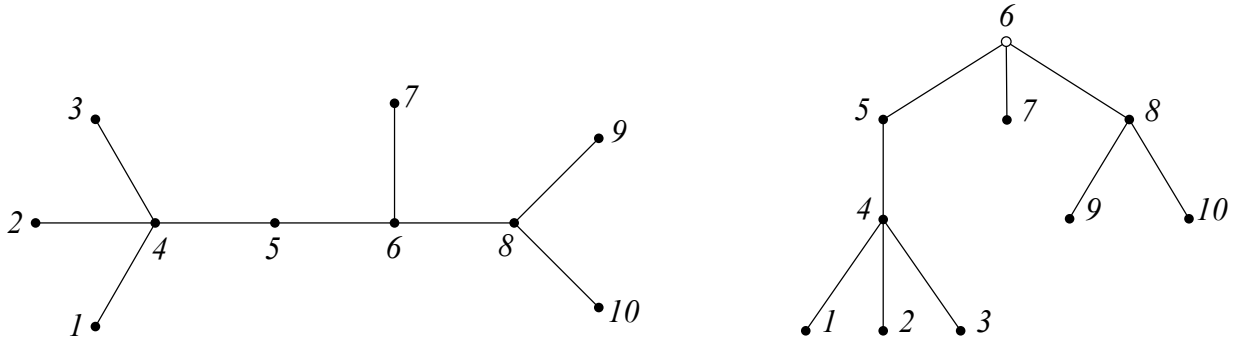
Решение. Удаление ребра $e = \{x, y\}$ разваливает T на две связные компоненты. Обозначим через V_1 и V_2 множества вершин этих двух компонент, такие, что $x \in V_1$, $y \in V_2$. В дереве T' существует единственный путь, соединяющий вершины x и y. Такой путь обязан иметь ребро e' из V_1 в V_2. Так как e есть единственное ребро в дереве T между V_1 и V_2, то ребро e' не принадлежит дереву T. Граф $T - e + e'$ является связным, содержит $n-1$ ребро, а следовательно, является остовным деревом графа G.

Заметим, что так как в дереве T имеется единственный путь между вершинами x и y, то добавление к этому пути ребра e' приводит к появлению единственного цикла в графе $T + e'$. Удаление же из этого графа ребра e уничтожает цикл, превращая граф $T - e + e'$ в связный ациклический подграф — еще одно остовное дерево графа G. □

10.2. Иногда наряду с обычными деревьями и лесами рассматривают так называемые *корневые* деревья и леса.

Определение 10.11. *Корневым деревом* называется дерево с выделенной вершиной, называемой корнем. *Корневым лесом* называется лес, каждая компонента связности которого представляет собой корневое дерево.

10.2.1. Наглядно корневое дерево можно представлять себе так. Представим дерево в виде некоторого физического объекта, вершинами которого являются некоторые шарниры, а ребрами — соединяющие эти шарниры трубки. Подвешивая такое дерево за какую-то вершину x, мы как раз и получим корневое дерево с корнем в вершине x.



(a) Дерево на десяти вершинах (b) Корневое дерево с корнем в вершине 6

Рис. 50: Дерево и корневое дерево

В качестве примера подвесим показанное на рис.50,a дерево за вершину 6. В результате получим корневое дерево, показанное на рис.50,b.

10.2.2. В любом корневом дереве T_r с корнем в вершине r у нас имеется единственный путь из корня в произвольную вершину x. Длина этого пути определяет *уровень* вершины x в корневом дереве T_r. *Высотой* дерева называется длина наибольшего пути от корня r дерева до вершины x. Ясно, что в последнем случае вершина x обязана быть листом дерева T_r.

Так, изображенное на рис.50,b дерево T_r имеет высоту, равную трем. Уровень вершины 4 равен двум.

10.2.3. Иногда корневое дерево T_r удобно ориентировать от корня r дерева к его листьям (см.рис.51), получая ориентированный граф $\overrightarrow{T}_r$, который в английском языке носит название arborescence. В таком орграфе входящая степень любой вершины, отличной от корня, будет

равняться единице. Корень ориентированного дерева $\overrightarrow{T}_r$ будет в этом случае источником, то есть вершиной, входящая степень которой равна нулю, а любой лист будет стоком, то есть вершиной, исходящая степень которой равна нулю.

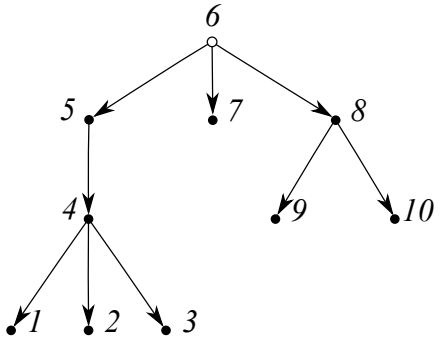


Рис. 51

В дереве $\overrightarrow{T}_r$ вершина x называется родителем (parent) вершины y, а вершина y — сыном (child) вершины x, если существует ориентированное ребро (x, y), ведущее из x в y. Так, на рис.51 вершина 5 является родителем для вершины 4, а вершина 4 — сыном для вершины 5.

Если существует ориентированный путь из вершины x в вершину z, то вершина x называется предком (ancestor) для вершины z, а вершина z — потомком (descendant) для вершины x. На рис.51 вершины 1, 2 и 3 являются потомками для вершины 5, а вершина 5 — их общим предком.

В случае, если наибольшая исходящая степень вершины в дереве равняется m, то такое дерево называется m-арным деревом. Так, дерево, показанное на рис.51, является 3-арным деревом. Часто в случае m-арных деревьев дополнительно подразумевается, что на множестве потомков произвольной вершины x задан некоторый порядок. Например, говоря про бинарные деревья, часто подразумевают, что это деревья, в которых каждая из вершин может иметь как максимум два различимых сына — левого и/или правого.

10.2.4. Выбор корня в дереве T задает на множестве $V(T)$ вершин этого дерева частичный порядок, превращая это множество в частично упорядоченное. Напомним определение такого множества.

Определение 10.12. *Частично упорядоченным множеством* называется множество P (конечное или счетное) с введенным на нем бинарным отношением $\preccurlyeq$ (так называемым *отношением частичного порядка*), удовлетворяющим следующим трем аксиомам:

1. Рефлексивность: для любого $x \in P$ справедливо $x \preccurlyeq x$.
2. Транзитивность: если $x \preccurlyeq y$ и $y \preccurlyeq z$, то $x \preccurlyeq z$.
3. Антисимметричность: если $x \preccurlyeq y$ и $y \preccurlyeq x$, то $x = y$.

В случае корневого дерева мы будем считать, что $x \preccurlyeq y$, если x лежит на пути, соединяющим корень r дерева с вершиной y. Иными словами, это означает, что в соответствующем T_r ориентированном дереве $\overrightarrow{T}_r$ вершина x является предком для вершины y.

Определение 10.13. *Цепью* в частично упорядоченном множестве P называется подмножество множества P, любые два элемента которого сравнимы между собой, а *антицепью* — подмножество, любые два элемента которого не сравнимы между собой.

В дереве любые вершины, лежащие на пути, соединяющем корневую вершину с одним из его листов, сравнимы между собой, то есть образуют *цепь* в рассматриваемом частично упорядоченном множестве. Примером антицепи является набор листьев дерева T.

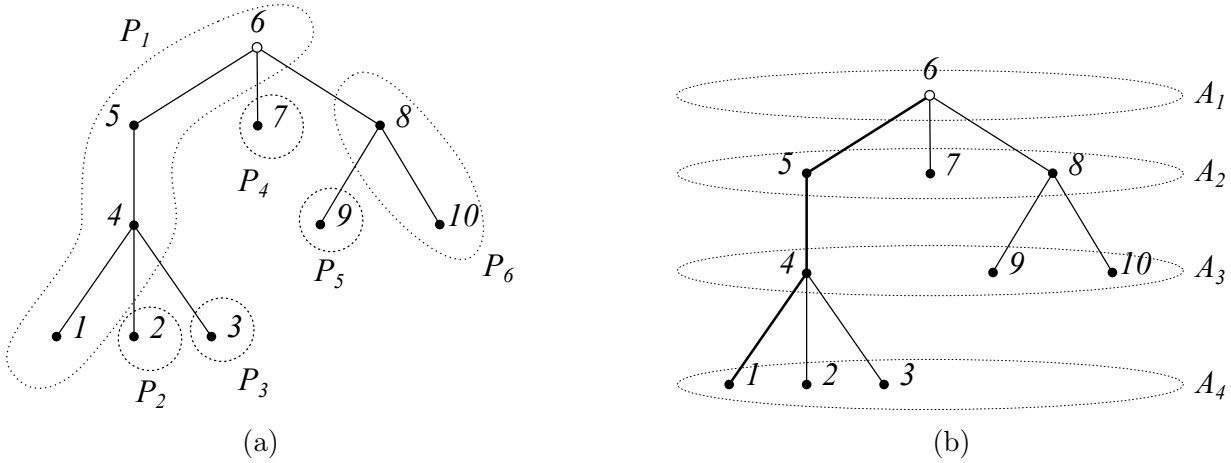


Рис. 52

В комбинаторике доказываются следующие две важные теоремы, описывающие разбиение множества P на цепи или антицепи.

Теорема 10.14 (Dilworth, 1950)**.** *В любом конечном частично упорядоченном множестве P найдется такое разбиение множества P на цепи P_i, $i = 1, \ldots, k$, количество k которых совпадает с количеством элементов в некоторой антицепи A.*

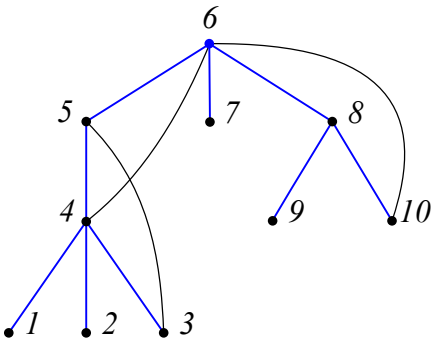
Теорема 10.15 (Mirsky, 1971)**.** *В любом конечном частично упорядоченном множестве P найдется такое разбиение множества P на антицепи A_j, $j = 1, \ldots, m$, количество m блоков в котором совпадает с длиной самой длинной цепи в P.*

В рассматриваемом случае множества вершин корневого дерева T данные утверждения имеют особенно наглядную интерпретацию (см. рис.52). Ясно, что подмножество листьев является наибольшей антицепью (см. рис.52,a). Кроме того, очевидно, как связать с каждым из листьев цепь P_i так, чтобы их объединение представляло собой разбиение множества $V(T)$ вершин графа T — нам нужно идти от листа вверх по дереву до тех пор, пока мы либо не встретим корень (и тогда его также нужно включить в цепь), либо пока мы не встретим вершину, принадлежащую уже построенной на предыдущих шагах цепи (и тогда эту вершину мы в цепь уже не включаем). При этом количество таких цепей как раз и совпадает с размером максимальной антицепи, в полном соответствии с теоремой Дилуорса. Наоборот, выбрав произвольный путь $(x_1, \ldots, x_m)$ максимальной длины m от корня до одного из листов дерева T, мы получим цепь максимальной длины (см. путь, помеченный жирными линиями на рис.52,b). При этом разбиение $V(G)$ на m антицепей A_i, $i = 1, \ldots, m$, строится следующим образом: мы в антицепь A_i включаем все вершины дерева T, лежащие на том же расстоянии от корня, что и вершина x_i.

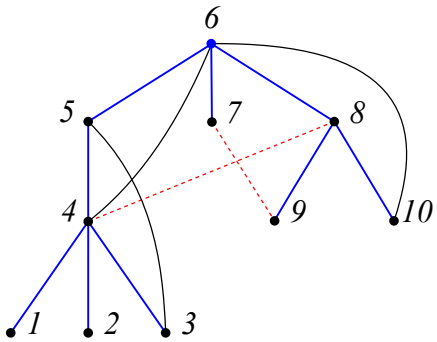
10.3. Давайте ненадолго вернемся к уже изученному нами алгоритму поиска в глубину в связном графе G.

10.3.1. Ранее мы показали, как с использованием данного алгоритма построить остовное дерево $T(G)$ графа G. Заметим сразу же, что в результате работы алгоритма мы, по сути, получаем не просто дерево, а корневое дерево — в качестве корня у нас выступает вершина, с которой мы

запускаем наш алгоритм (вершина 1 на рис.48). Такое корневое дерево $T^*(G)$ обладает одним очень полезным свойством, о котором следует сказать несколько слов. Оказывается, любое такое дерево является *нормальным.*



(a) Нормальное остовное дерево



(b) Остовное дерево, нормальным не являющееся

Рис. 53

Определение 10.16. Остовное дерево $T^*(G)$ в связном графе G называется нормальным, если концевые вершины x, y любого ребра $e = \{x, y\}$, не принадлежащего дереву $T^*(G)$, сравнимы относительно частичного порядка $\preccurlyeq$, введенного на множестве вершин корневого дерева $T^*(G)$. Иными словами, дерево $T^*(G)$ нормально, если один из концов x, y любого ребра, не принадлежащего этому дереву, лежит на единственном пути между вторым его концом и корнем дерева $T^*(G)$.

Рассмотрим в качестве примера остовное корневое дерево $T^*(G)$ для графа G, показанного на рис.53,a. В качестве корня дерева выбрана вершина 6, помеченная синим цветом на рисунке. Синим же цветом помечены ребра остовного дерева $T^*(G)$. Черные ребра — это ребра, не вошедшие в остовное дерево. Заметим, что в этом дереве имеется шесть листьев — вершины $1, 2, 3, 7, 9, 10$. Каждый из этих листьев соединен с корневой вершиной 6 некоторым простым путем. Все вершины, лежащие на любом из этих путей, сравнимы между собой относительно частичного порядка $\preccurlyeq$. Заметим теперь, что концевые вершины всех черных ребер лежат на каком-то из этих путей, соединяющих один из листов с корнем. Так, ребро $\{3, 5\}$ лежит на единственном пути $\{3, 4, 5, 6\}$, соединяющем лист 3 дерева $T^*(G)$ с корнем — вершиной 6, ребро $\{4, 6\}$ лежит, к примеру, на пути $\{1, 4, 5, 6\}$, и так далее. Как следствие, такое остовное дерево является нормальным.

Рассмотрим теперь граф, показанный на рис.53,b. В таком графе помеченное синим цветом дерево $T^*(G)$ по-прежнему является остовным, однако нормальным оно уже не является. Действительно, рассмотрим ребра $\{7, 9\}$ и $\{4, 8\}$, помеченные красным цветом на рисунке. В дереве $T^*(G)$ отсутствует путь из листа в корень, который содержал бы обе концевые вершины ребра $\{7, 9\}$. Тот же факт имеет место и для ребра $\{4, 8\}$.

Давайте теперь поймем, почему дерево $T^*(G)$ для графа, показанного на рис.53,b, невозможно построить с помощью алгоритма поиска в глубину. Предположим, что, выйдя из корня, мы пришли вначале в вершину 5, а затем — в вершину 4. После обхода смежных с 4 вершин $1, 2, 3$ мы обязаны пойти в еще одну смежную с 4 вершину — вершину 8. Следовательно, мы должны включить ребро $\{4, 8\}$ в строящееся остовное дерево. Наоборот, прийдя вначале в вершину 8, мы бы по тем же соображениям должны были бы зайти во все смежные с ней вершины, и в том числе в вершину 4. И тогда и в этом случае ребро $\{4, 8\}$ было бы ребром остовного дерева, строящегося поиском в глубину.

10.3.2. Отметим, однако, что построенное с помощью алгоритма поиска в глубину дерево $T^*(G)$ наряду с отмеченными выше достоинствами обладает и одним существенным недостатком. Именно, мы с вами знаем, что в любом дереве существует единственный путь между любыми двумя вершинами. В частности, в дереве $T^*(G)$ существует единственный путь между корневой вершиной и любой другой вершиной графа G. Длина этого пути совпадает с расстоянием между этими вершинами в дереве $T^*(G)$. Однако в исходном графе G расстояние между такими вершинами может быть меньше.

В качестве примера рассмотрим вершины 1 и 3 на рис.48. В построенном нами остовном дереве $T^*(G)$ расстояние между этими вершинами равно двум. При этом расстояние между этими же вершинами в исходном графе G меньше, чем в дереве $T^*(G)$ — в графе G это расстояние равно единице.

Итак, с помощью алгоритма поиска в глубину мы, помимо всего прочего, находим какие-то пути между корневой вершиной и оставшимися вершинами графа G, однако эти пути не обязательно оказываются кратчайшими с точки зрения путей в исходном графе G. В некоторых же приложениях нам необходимо получать такое остовное дерево $\widetilde{T}(G)$, в котором расстояние между корневой вершиной и оставшимися вершинами графа G совпадает с расстоянием между этими вершинами в исходном графе, то есть дерево, состоящее из кратчайших путей между такими вершинами. Алгоритм, позволяющий такого рода дерево получать, называется *алгоритмом поиска в ширину.*

При построении алгоритма нам понадобится дополнительная абстрактная структура данных, называемая очередью Q. В такой структуре данных элементы линейно упорядочены, то есть образуют линейный список, добавлять элементы мы можем по одному в конец этого списка, а удалять — также по одному, но с начала этого списка. Такого рода структуру данных иногда еще называют структурой данных вида First-In-First-Out (первым вошел — первым вышел: первый элемент, добавленный в очередь, первым же из нее выйдет).

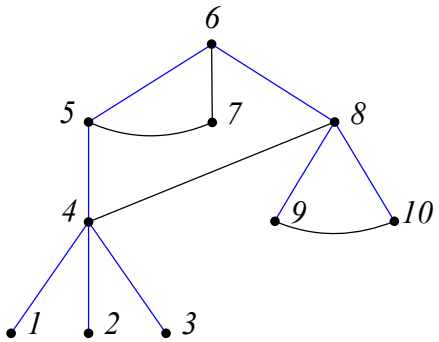


Рис. 54

Работа алгоритма начинается с того, что мы помещаем в очередь Q первую вершину графа — будущую корневую вершину строящегося остовного дерева $T(G)$ (вершина 6 на рис.54). Одновременно с этим действием мы будем считать, что эта вершина окрашивается в какой-то новый цвет, например, в красный. Теперь на каждом шаге алгоритма мы будем совершать следующие действия: вынимать вершину x из очереди Q, смотреть на все смежные с x вершины, и те из них, которые окрашены в черный цвет, помещать в очередь, одновременно окрашивая их в красный цвет. Алгоритм закончится тогда, когда в очереди Q никаких вершин не останется. Соответствующий описанному алгоритму псевдокод можно записать следующим образом:

```
Процедура BFS(x,G), x - стартовая вершина графа G:
```

```
// Инициализация начальных значений
for ( y : y - вершина графа G )
  begin
    color[y] := 0;
    parent[y] := -1;
    dist[y] := 0;
  end

// Помещаем в очередь корневую вершину
color[x] := 1;
Q.push(x);

// Запуск алгоритма поиска в ширину
while ( Q не пусто )
  begin
    y := Q.pop;
    for (z : (y,z) принадлежит E(G) )
      begin
        if ( color[z] == 0)
          begin
            color[z] := 1;
            Q.push(z);
            dist[z] := dist[y] + 1;
            parent[z] := y;
          end
      end
  end
```

В этом алгоритме мы дополнительно ввели массив `dist[]` кратчайших расстояний от заданной вершины до корня, а также массив `parent[]`, позволяющий для любой вершины определить единственного его родителя в строящемся остовном дереве.

Результат работы алгоритма для дерева, показанного на рис.54, можно свести в следующую таблицу:

Обрабатываемая вершина	Содержание очереди
	$[6\rangle$
6	$[8, 7, 5\rangle$
5	$[4, 8, 7\rangle$
7	$[4, 8\rangle$
8	$[10, 9, 4\rangle$
4	$[3, 2, 1, 10, 9\rangle$
9	$[3, 2, 1, 10\rangle$
10	$[3, 2, 1\rangle$
1	$[3, 2\rangle$
2	$[3\rangle$
3	$[\rangle$

Дадим в заключение физическую интерпретацию остовного дерева в графе, полученного алго-

ритма поиска в ширину. Если считать вершины графа шариками, а ребра — соединяющими эти шарики пружинками, то, подвесив граф за выбранный шарик, мы получим картинку подобную изображенной на рис.54. При этом все шарики (вершины графа) распределятся по слоям — на нулевом слое будет корневая вершина, на следующем — вершины, смежные с корневой, и так далее.

11 Перечисление деревьев. Формула Cayley

11.1. Перейдем теперь к задаче перечисления всех (помеченных) деревьев, построенных на n вершинах.

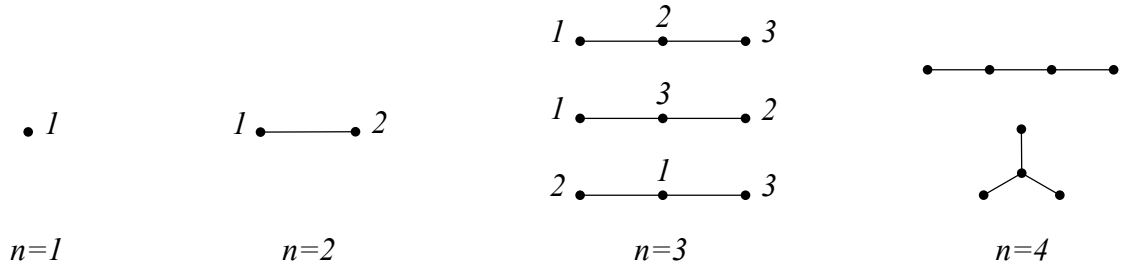


Рис. 55: Неизоморфные друг другу деревья для $n \leqslant 4$

11.1.1. Несложно убедиться, что существует по одному дереву, построенному на одной и на двух вершинах, а также три дерева, построенные на трех вершинах (см.рис.55). Все эти три дерева отличаются друг от друга только расположением меток вершин. В случае $n = 4$ таких деревьев существует уже 16 штук — 12 деревьев, отвечающих линейной цепочке вершин, и 4 дерева, построенных на "треножнике". Задача этого пункта — доказать следующий общий результат, известный как формула Кэли.

Теорема 11.1 (Формула Кэли)**.** *Количество a_n различных помеченных деревьев на n вершинах равно n^{n-2}.*

11.1.2. К настоящему моменту известно около шестнадцати различных способов доказательства этого результата (см., например, [**?**]). Метод, связанный с производящими функциями, мы рассмотрим в курсе комбинаторики. Здесь же мы приведем доказательство этой формулы, впервые предложенное Хайнсом Прюфером (Heinz Prüfer) в 1918 году. Это доказательство является, пожалуй, наиболее простым и понятным, и входит практически в любой учебник по комбинаторике и теории графов. В основе доказательства лежит идея построения по данному дереву некоторой строки длины $n-2$ над алфавитом из n чисел, по которой можно однозначно восстановить исходное дерево.

11.1.3. Рассмотрим дерево T, вершины которого помечены элементами множества $[n]$, $n \geqslant 2$, и покажем вначале, как сопоставить такому дереву числовую последовательность вида

$$P(T) = (y_1, y_2, \ldots, y_{n-2}), \qquad y_i \in [n],$$

называемую кодом Прюфера (или последовательностью Прюфера). Для этого на первом шаге выберем среди всех листьев дерева T вершину x_1 с минимальным номером, удалим эту вершину вместе с инцидентным ей ребром $e_1 = \{x_1, y_1\}$, а затем запишем смежную с удаленным листом

вершину y_1 в качестве первого символа в нашем коде. Затем возьмем дерево T_1, полученное из T удалением вершины x_1, и выберем среди его листьев вершину x_2 с минимальным номером. Удалим этот лист вместе с инцидентным ему ребром $\{x_2, y_2\}$, а в код Прюфера запишем инцидентную x_2 вершину y_2. Будем повторять этот процесс до тех пор, пока у нас не останется дерево, состоящее ровно из двух вершин и соединяющего их ребра.

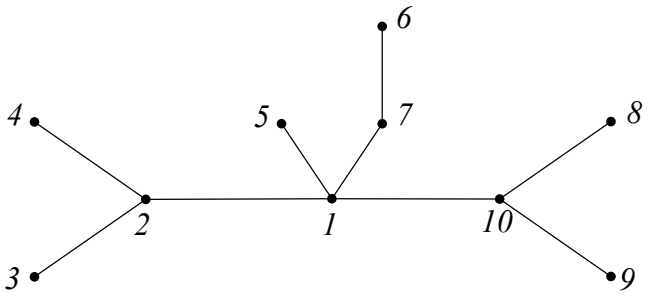


Рис. 56: Дерево T на десяти вершинах

Продемонстрируем этот алгоритм на дереве T, показанном на рис.56. Среди всех его листьев минимальный номер имеет вершина $x_1 = 3$. Удалим эту вершину 3 вместе с инцидентным ей ребром $e_1 = \{3, 2\}$, а в последовательность Прюфера добавим смежную с ней вершину y_1, то есть вершину 2. Среди листьев дерева T_1, полученного из T удалением вершины 3, минимальный номер имеет вершина $x_2 = 4$. Ее удаление добавляет нам второй член $y_2 = 2$ последовательности Прюфера. Продолжая этот процесс, на заключительном, 8-м шаге мы получаем дерево, построенное на вершинах 10 и 9, а также код Прюфера

$$P(T) = (2, 2, 1, 1, 7, 1, 10, 10).$$

Нам осталось доказать, что описанное выше отображение, сопоставляющее любому помеченному дереву T числовую последовательность $P(T)$, взаимно-однозначно. Для этого нам достаточно показать, что по любому коду Прюфера $P(T)$ мы сможем однозначно восстановить исходное дерево T.

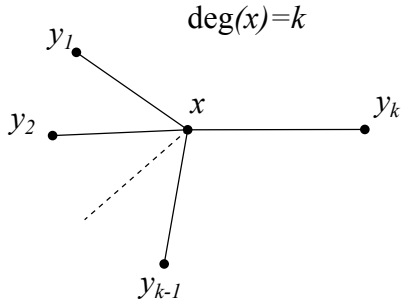


Рис. 57

11.1.4. Заметим, прежде всего, что листья исходного дерева T в этом коде вовсе не встречаются. Действительно, удаляя лист, мы записываем в последовательность Прюфера не номер листа, а номер смежной с ним вершины. Рассмотрим теперь более общий случай вершины x, степень $\deg(x) = k$ которой строго больше единицы (рис.57). В процессе построения кода Прюфера мы постепенно удаляем смежные с x вершины y_i, $i = 1, \ldots, k$, до тех пор, пока степень вершины x не станет равной единице. На этом шаге сама вершина x стала листом, и, если нам придется удалить вершину x, то мы уже в последовательность Прюфера включим не вершину

x, а смежную с ней вершину y_k. Таким образом, любая вершина x встречается в коде Прюфера $P(T)$ ровно $\deg_T(x) - 1$ раз.

11.1.5. Давайте теперь подумаем, что в идеале нам требуется, чтобы восстановить дерево на n вершинах. Если бы мы в процессе удаления вершины записывали бы не инцидентную ей вершину, а пару вершин — смежную и удаляемую, то мы легко бы по такому набору пар вершин восстановили исходное дерево. Действительно, рассмотрим пару последовательностей

$$\begin{array}{cccccccc} 2 & 2 & 1 & 1 & 7 & 1 & 10 & 10 \\ 3 & 4 & 2 & 5 & 6 & 7 & 1 & 8 \end{array}$$

для показанного на рис.56 дерева T. Первая из них — это последовательность Прюфера $P(T)$. Вторая последовательность — это список удаляемых на каждом шаге листьев. Зная эти пары чисел, мы знаем все ребра исходного дерева T кроме, разве что, последнего ребра. Однако и его мы легко восстановим. Действительно, последним у нас всегда остается ребро, соединяющее вершину, имеющую наибольший номер n, с некоторой вершиной z. Вершина z — это единственная вершина, отличная от вершины с максимальным номером, не встречающаяся во второй последовательности — последовательности удаляемых вершин. Так, в случае дерева, показанного на рис.56, у нас в конце остается ребро, соединяющее вершину 10 с вершиной $z = 9$, не встречающейся во второй строчке таблицы и отличной от 10.

Итак, все, что нам не хватает — это восстановить по верхней последовательности $P(T)$ — коду Прюфера — соответствующую ей нижнюю последовательность вершин, удаляемых на соответствующем шаге алгоритма:

$$\begin{array}{rl} P(T) = & (y_1, y_2, \ldots, y_{n-3}, y_{n-2}) \\ & (x_1, x_2, \ldots, x_{n-3}, x_{n-2}) \quad + \quad \{x_{n-1}, x_n \equiv n\}. \end{array}$$

Так как любую вершину в процессе построения кода Прюфера $P(T)$ мы удаляем лишь однажды, то все числа в этой последовательности $\{x_i\}$ различны. Кроме того, в эту последовательность не входит вершина $x_n \equiv n$ с максимальным номером, а также вершина x_{n-1}, которая остается соединенной с n ребром на последнем шаге построения последовательности Прюфера.

11.1.6. Начнем строить последовательность $\{x_i\}$ с восстановления числа x_1, то есть с восстановления первой удаленной нами вершины. Напомним, что на первом шаге мы брали листья дерева, выбирали из них вершину с минимальным номером и ее удаляли. С точки зрения кода Прюфера листья — это все числа, не вошедшие в последовательность $P(T) = \{y_i\}$. Минимальное из этих чисел и будет нужным нам числом x_1. Так, для дерева, показанного на рис.56, в код Прюфера $P(T)$ не входят числа

$$\{3, 4, 5, 6, 8, 9\}.$$

Число 3 является минимальным среди этих чисел. Следовательно, именно вершину под номером 3 мы и удалили из дерева T на первом шаге построения кода Прюфера $P(T)$ для дерева T. В общем же случае мы рассматриваем множество чисел вида

$$X = \{1, 2, \ldots, n-1, n\},$$

исключаем из него числа, входящие в последовательность $P(T)$, выбираем из оставшихся минимальное — число x_1, и записываем его под числом y_1:

$$\begin{array}{l} (y_1, y_2, \ldots, y_{n-3}, y_{n-2}) \\ (x_1, \end{array}$$

Тем самым мы находим первое ребро — ребро $\{x_1, y_1\}$ — восстанавливаемого нами дерева.

11.1.7. Теперь нам нужно восстановить число x_2, то есть вершину, удаленную на втором шаге построения кода Прюфера $P(T)$. На этом шаге мы рассматривали поддерево T_1, полученное из T удалением листа x_1, и в этом поддереве снова выбирали лист с минимальным номером. С точки зрения кода Прюфера поддереву T_1 отвечает числовая последовательность, полученная из $P(T)$ удалением первого члена y_1:

$$P(T_1) = (y_2, y_3, \ldots, y_{n-3}, y_{n-2}).$$

Действительно, удаляя в дереве T вершину x_1, мы удаляем и соединяющее ее с вершиной y_1 ребро, а следовательно, уменьшаем степень y_1 на единицу. Но тогда мы должны уменьшить на единицу и вхождение y_1 в код Прюфера. Этого мы и достигаем, удаляя из последовательности $P(T)$ число y_1, стоящее в этой последовательности на первом месте.

А теперь для восстановления x_2 мы должны провести те же рассуждения, что и на предыдущем шаге. Именно, у нас имеется множество $X \setminus \{x_1\}$ оставшихся после первого шага чисел. Исключим из него числа, входящие в код Прюфера $P(T_1)$ поддерева T_1, а затем выберем среди оставшихся чисел минимальное. Выбранное число x_2 и будет соответствовать вершине, которую мы удалили на втором шаге при построении кода Прюфера исходного дерева T. Записывая его под числом y_2 в коде Прюфера, мы восстанавливаем второе ребро дерева T:

$$\begin{matrix}(y_1, y_2, \ldots, y_{n-3}, y_{n-2})\\ (x_1, x_2,\end{matrix}$$

Так, в рассматриваемом примере у нас после первого шага осталось множество чисел

$$X_1 := X \setminus \{x_1\} = \{1, 2, 4, 5, 6, 7, 8, 9, 10\}.$$

Исключим из них числа, входящие в обрезанный слева на единицу код Прюфера

$$P(T_1) = (2, 1, 1, 7, 1, 10, 10).$$

Среди оставшихся в множестве X_1 чисел минимальным является число 4. Следовательно, удаленным на втором шаге оказалось ребро $\{x_2, y_2\} = (4, 2)$ дерева T.

11.1.8. Теперь мы можем уже описать и общий шаг алгоритма восстановления дерева T по коду Прюфера $P(T)$. Предположим, что мы уже восстановили ребра $\{x_1, y_1\}, \ldots, \{x_{k-1}, y_{k-1}\}$ дерева T. Рассмотрим множество чисел

$$X_{k-1} := X \setminus \{x_1, x_2, \ldots, x_{k-1}\}.$$

Исключим из этого множества все числа, встречающиеся в коде Прюфера $P(T_{k-1})$, полученном из исходного кода Прюфера $P(T)$ удалением стоящих на первых $(k-1)$ позициях чисел $y_1, \ldots, y_{k-1}$. Выбирая из оставшихся в множестве X_{k-1} чисел минимальное число x_k и записывая его под числом y_k, мы тем самым восстанавливаем ребро $\{x_k, y_k\}$ дерева T, отвечающего последовательности Прюфера $P(T)$:

$$\begin{matrix}(y_1, y_2, \ldots, y_k, \ldots, y_{n-3}, y_{n-2})\\ (x_1, x_2, \ldots, x_k,\end{matrix}$$

11.1.9. Продолжая этот процесс, мы восстановим все числа x_k, $k = 1, \ldots, (n-2)$, то есть $n-2$ ребра $\{x_i, y_i\}$ дерева T, $i = 1, \ldots, n$. Оставшееся ребро $\{x_{n-1}, n\}$ отвечает двум не вошедшим в

набор $\{x_i\}$ числам x_{n-1} и n. Последовательно присоединяя к ребру $\{x_{n-1}, n\}$ ребра $\{x_{n-2}, y_{n-2}\}$, $\{x_{n-3}, y_{n-3}\}$, ..., мы построим в конечном итоге дерево T, отвечающее коду Прюфера $P(T)$.

В рассматриваемом примере мы по коду Прюфера

$$P(T) = (2, 2, 1, 1, 7, 1, 10, 10)$$

восстанавливаем числа

$$(x_1, \ldots, x_8) = (3, 4, 2, 5, 6, 7, 1, 8),$$

а следовательно, 8 ребер дерева T. Добавляя эти ребра к ребру $\{9, 10\}$, мы в результате получаем дерево T, показанное на рис.56.

12 Эйлеровы циклы

12.1. Обратимся теперь к самой первой по времени содержательной задаче теории графов — задаче о кенигсбергских мостах (смотри рис.58,a), которая была предложена жителями города Кенигсберга (ныне — Калининграда) для решения Леонарду Эйлеру в тридцатых годах восемнадцатого века.

12.1.1. Вот как описывал постановку задачи сам Эйлер: "Некогда мне была предложена задача об острове, расположенном в городе Кенигсберге и окруженном рекой, через которую перекинуто семь мостов. Спрашивается, может ли кто-нибудь обойти их, переходя только однажды через каждый мост. И тут же мне было сообщено, что никто до сих пор не мог это проделать, но никто и не доказал, что это невозможно. Вопрос этот, хотя и банальный, показался мне, однако, достойным внимания тем, что для его решения недостаточны ни геометрия, ни алгебра, ни комбинаторное искусство...". Эйлер не только решил эту задачу, но и установил необходимое условие, позволяющее определить, можно ли обойти любой город, имеющий мосты, так, чтобы пройти по всем мостам, не проходя дважды ни по одному из них.

12.1.2. Для решения задачи о кенигсберсгских мостах вслед за Эйлером нам следует, прежде всего, формализовать эту задачу. Именно, построим упрощенную схему города, заменяя части города точками — вершинами графа, а мосты — дугами, то есть ребрами этого графа (смотри рис.58,a). В результате мы придем к графу, изображенному на рис.58,b.

(a) Кёнигсберг (b) Граф

Рис. 58: К задаче о кёнигсбергских мостах

Теперь настало время дать несколько дополнительных определений.

Определение 12.1. *Эйлеровым путем* в произвольном (не обязательно простом) графе G называется путь, который проходит через *каждое* ребро графа ровно один раз. Эйлеров путь, начинающийся и заканчивающийся в одной и той же вершине, называется *эйлеровым циклом* C.

Определение 12.2. Любой граф G, в котором существует эйлеров цикл C, называется *эйлеровым графом.* Граф, в котором существует эйлеров путь, называется *полуэйлеровым.*

Итак, нам нужно определить, является ли граф, изображенный на рис.58,b, эйлеровым. Эйлер ответил на этот вопрос отрицательно, доказав следующее необходимое условие существования эйлерова цикла в графе.

Теорема 12.3 (Необходимое условие существования эйлерова цикла в графе)**.** *Для существования в графе эйлерова цикла необходимо, чтобы он имел как максимум одну нетривиальную (то есть отличную от одиночных вершин) компоненту связности, и чтобы все вершины этого графа имели четную степень.*

Доказательство этого факта довольно несложно. Предположим, что в графе G существует эйлеров цикл C. При движении вдоль этого цикла мы посещаем каждое ребро в графе лишь однажды. Следовательно, войдя в какую-то из вершин по одному ребру, мы должны выйти из этой же вершины по какому-то другому ребру. При этом количество входов в любую вершину должно совпадать с количеством выходов. Удовлетворить этим требованиям мы можем лишь тогда, когда степень любой вершины является четной. Кроме того, если два ребра принадлежат одному и тому же циклу C, то они, очевидно, лежат в одной и той же связной компоненте графа G. Как следствие, G может содержать как максимум одну нетривиальную связную компоненту. □

В графе, представленном на рис.58,b, имеются вершины нечетных степеней. Следовательно, эйлерова цикла в нем не существует.

12.1.3. Эйлер оставил без доказательства достаточность сформулированного им условия. Первое полное доказательство теоремы об эйлеровом цикле было дано немецким математиком Карлом Хиерхолцером лишь в 1873 году.

Теорема 12.4 (Достаточное условие существования эйлерова цикла в графе)**.** *Для того, чтобы граф имел эйлеров цикл, достаточно, чтобы он имел как максимум одну нетривиальную компоненту, и чтобы любая его вершина имела четную степень.*

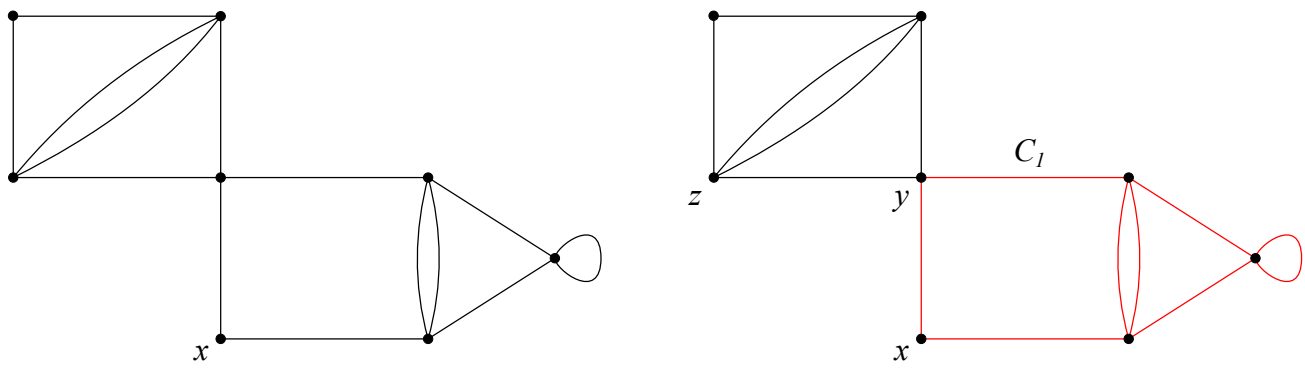


Рис. 59

Доказательство. Выберем в нетривиальной компоненте связности графа G, все вершины которого имеют четную степень (рис. 59,a), произвольную вершину $x \in V(G)$. Будем совершать обход этого графа, проходя по каждому ребру лишь один раз, до тех пор, пока мы не сможем двигаться дальше, не нарушая это условие (иными словами, будем пытаться строить наибольший по включению путь в графе). Так как любая вершина в графе имеет четную степень, то

войдя в любую вершину графа, отличную от x, по одному из ребер, мы всегда сможем из нее выйти по какому-то другому ребру. Единственным исключением в этом смысле является сама вершина x: как только мы вернемся в нее, обойдя по разу каждое из инцидентных x ребер, то мы уже не сможем из нее выйти. Итак, процесс обхода неизбежно закончится в вершине x. Иными словами, мы доказали, что любой наибольший по включению путь в графе, все степени вершин которого четны, представляет собой замкнутый путь.

Обозначим полученный в процессе такого обхода графа G замкнутый путь через C_1 (см. рис. 59(b), на котором входящие в C_1 ребра помечены красным цветом). Если он совпал со всем графом, то все доказано — граф G является эйлеровым. В противном случае у нас в графе остались какие-то ребра, через которые мы еще не прошли (ребро e на рис.59,b). Так как у нас имеется лишь одна нетривиальная компонента связности, то и e, и C_1 ей принадлежат. Следовательно, у нас имеется кратчайший путь, соединяющий e с вершиной x, а значит, и ребро e', инцидентное одной из вершин C_1 (вершина y на рис.59,b), но самому замкнутому пути C_1 не принадлежащее (см. рис.59,b).

Введем тогда подграф $G - E(C_1)$, образованный ребрами, не вошедшими в C_1, и повторим для него описанную выше процедуру обхода, начинающуюся с вершины y. Полученный в результате такого обхода замкнутый путь C_2 можно объединить с C_1 в единый замкнутый путь $C_1 \cup C_2$. Действительно, стартуя с точки $x \in C_1$, мы можем остановиться в точке $y \in C_1$, обойти все ребра из C_2, а затем продолжить обход по оставшейся части C_1. Если теперь $C_1 \cup C_2 = G$, то все доказано. Если же нет, то нам следует продолжить описанный выше алгоритм до тех пор, пока полученное на k-м шаге объединение замкнутых путей $C_1 \cup \ldots \cup C_k$ не совпадет со всем графом G. □

Замечание 12.5. Проведенное доказательство достаточности достаточно легко превращается в алгоритм поиска эйлерова цикла в произвольном графе G, работающий за полиномиальное по количеству ребер в графе время.

12.1.4. Обобщением эйлерова графа является понятие так называемого *четного* графа.

Определение 12.6. Граф G называется четным, если любая его вершина имеет четную степень.

Итак, в случае связного графа G мы выше доказали следующее утверждение.

Теорема 12.7. *Связный граф G является эйлеровым тогда и только тогда, когда он четный.*

Из этой теоремы немедленно вытекает и следующее

Следствие 12.8. *Связный граф G имеет эйлеров путь, начинающийся в вершине $x \in V(G)$ и заканчивающийся в некоторой другой вершине $y \in V(G)$, тогда и только тогда, когда степени вершин x и y нечетные, а степени всех остальных вершин являются четными.*

Доказательство. Действительно, добавим к графу G еще одно дополнительное ребро, соединяющее точки x и y. Полученный в результате граф имеет эйлеров цикл тогда и только тогда, когда исходный граф G имеет эйлеров путь, соединяющий точки x и y. Это обстоятельство и доказывает следствие 12.8. □

12.1.5. Описанные выше результаты достаточно легко обобщаются на случай эйлеровых циклов в ориентированных графах.

Определение 12.9. Эйлеровым циклом в орграфе D называется замкнутый путь, содержащий все ребра орграфа D. Орграф D называется эйлеровым, если в нем имеется хотя бы один эйлеров цикл.

Теорема 12.10. *Орграф D является эйлеровым тогда и только тогда, когда*

$$\mathrm{outdeg}(x) = \mathrm{indeg}(x) \qquad \text{для любой вершины } x \in V(D),$$

а соответствующий D underlying graph G имеет не более одной нетривиальной компоненты связности, то есть компоненты связности, отличной от изолированной вершины.

Доказательство. практически аналогично доказательству теоремы Эйлера для неориентированных графов. В частности, совершенно аналогично показывается, что любой наибольший по включению ориентированный путь замкнут. Далее, мы вновь выбираем произвольную вершину $x \in V(D)$ и пытаемся построить из нее наибольший по включению замкнутый путь C_1. Если он обходит все ребра D, то все доказано. Иначе осталось некоторое ребро $e \notin C_1$. Так как и x, и e принадлежат соответствующей D нетривиальной компоненте underlying graph G, то имеется путь кратчайшей длины между e и x, а значит, ребро e', инцидентное вершине $y \in C_1$ и не принадлежащее C_1. Это ребро может как входить в y, так и выходить из нее. Рассматривая тогда в орграфе $D - E(C_1)$ наибольший по включению путь, исходящий из y, мы некоторый замкнутый путь C_2. Если объединение $C_1 \cup C_2$ не покрывает все ребра орграфа D, то продолжаем описанный процесс до тех пор, пока на k-м шаге не построим замкнутый путь, обходящий все ребра орграфа. □

12.2. Доказанная в предыдущей главе теорема Татта о количестве остовных деревьев в орграфе D позволяет подсчитать количество эйлеровых циклов в произвольном ориентированном эйлеровом графе D. Этот результат известен как the BEST theorem. Свое название теорема получила по первым буквам фамилий математиков, ее доказавшим: в 1941 году Cedric Smith и William Tutte подсчитали количество эйлеровых циклов в графе, исходящая степень вершин которого равна двум, а в 1951 году Nicholaas de Bruijn и Tatyana van Aardenne-Ehrenfest обобщили эту формулу на случай произвольного значения исходящих степеней вершин орграфа.

Теорема 12.11 (de Bruijn, van Ardenne-Ehrenfest, Smith, and Tutte)**.** *Количество $e(D)$ эйлеровых циклов в орграфе D рассчитывается по формуле*

$$e(D) = t^-(D, x) \cdot \prod_{y \in D} (\mathrm{outdeg}(y) - 1)!,$$

где x — произвольная вершина орграфа D, $t^-(D, x)$ — количество корневых остовных деревьев, все ребра которых направлены к корню в вершине x.

Следствие 12.12. *Количество $t^-(D, x)$ корневых остовных деревьев в* эйлеровом *орграфе D не зависит от выбора корня, то есть вершины x.*

12.2.1. Зафиксируем произвольную вершину x в орграфе D (вершина 1 на рис.60), а также некоторое ребро $e = (x, y)$, являющееся стартовым ребром для любого эйлерова цикла W в графе (ребро $e_1 = (1, 3)$ на рис.60). Покажем вначале, как по эйлерову циклу построить остовное дерево с корнем в вершине x.

Мы знаем, что эйлеров цикл проходит по каждому ребру орграфа D ровно один раз, задавая на множестве $E(D)$ ребер некоторый линейный порядок (см. ребра $e_1, \ldots, e_{12}$ на рис.60). Кроме

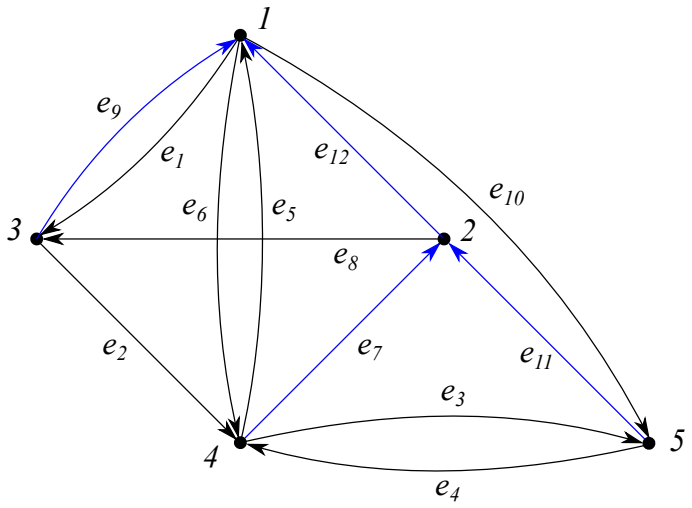


Рис. 60

того, он хотя бы один раз проходит через каждую вершину орграфа D. Выберем для каждой вершины $y \neq x$ исходящее из y ребро с наибольшим номером (см. ребра, помеченные синим цветом на рис.60). Покажем, что порожденный этими ребрами подграф T представляет собой корневое остовное дерево с корнем в вершине 1.

Действительно, мы построили какой-то остовный подграф, состоящий из $(n-1)$-го ребра. Покажем, что в таком подграфе циклы отсутствуют. Действительно, если бы мы, выйдя из какой-то вершины z по ребру e_i, вошли бы в нее же по какому-то ребру e_j, то мы должны были бы получить, что $j > i$. Но мы обходим вершины орграфа D по эйлеровому циклу, начинающемуся в вершине 1. Это означает, что, войдя в какую-то вершину $z \neq x$ по ребру с номером j, мы выйти из нее должны по ребру с номером $j+1$. Но тогда ребро e_i — это не ребро с наибольшим номером, выходящее из вершины z. Получили противоречие. Таким образом, синие ребра в орграфе D циклов не образуют, покрывают все вершины орграфа, а количество этих ребер равняется $n-1$. Следовательно, эти ребра порождают остовное дерево в орграфе D с корнем в вершине 1.

12.2.2. Рассмотрим теперь какое-то остовное дерево T, все ребра которого направлены к корню в вершине x. Нам нужно понять, сколько различных эйлеровых циклов мы сможем построить из одного и того же остовного дерева T. Для этого давайте пронумеруем (с повторениями) все ребра орграфа D согласно следующему принципу. Зафиксированному в самом начале ребру (x, y), с которого, как мы договорились, будут начинаться все эйлеровы циклы в орграфе D, мы присвоим номер 1 (ребро e_1 на рис.60). Остальным $\mathrm{outdeg}(x) - 1$ ребрам назначим $(\mathrm{outdeg}(x) - 1)!$ количеством способов произвольные номера из множества $\{2, \ldots, \mathrm{outdeg}(x)\}$. Для любой другой вершины $z \neq x$ орграфа D мы присвоим номер, равный $\mathrm{outdeg}(z)$, исходящему из вершины z ребру, принадлежащему остовному дереву. Остальным ребрам мы вновь $(\mathrm{outdeg}(z) - 1)!$ количеством способов назначим произвольные номера из множества $\{1, \ldots, \mathrm{outdeg}(z) - 1\}$. Всего, таким образом, мы получаем

$$d := \prod_{z \in D} (\mathrm{outdeg}(z) - 1)!$$

способов пронумеровать ребра орграфа D.

Убедимся теперь, что любая такая нумерация ребер задает нам один из d ориентированных эйлеровых циклов в орграфе D. Для этого при заданной нумерации ребер определим маршрут

в орграфе D следующим образом. Начнем маршрут с ребра (x, y), помеченного номером 1. При заходе в любую вершину z орграфа покинем эту вершину по еще не использованному ребру с наименьшим номером. Так как при входе в любую вершину $z \neq x$ у нас количество исходящих из z ребер на единицу больше количества входящих, то мы всегда из z сможем перейти в какую-то другую вершину. Остановиться же мы сможем только в вершине x. При этом все входящие в x ребра к этому моменту будут уже использованы. По построению это означает, что использованными оказались все ребра каждой из вершин, находящихся на расстоянии 1 от корня в остовном дереве. Далее по индукции можно показать, что использованными оказались вообще все ребра орграфа D.

12.2.3. Итак, мы для любого остовного дерева построили d различных эйлеровых циклов в D. Кроме того, различные остовные деревья порождают нам различные множества эйлеровых циклов. Таким образом, $e(D)$ действительно равно $t^-(D, x) \cdot d$. Теорема доказана.

12.3. Как мы уже упоминали, эйлеровы графы встречаются в самых разнообразных практических задачах. В качестве очень красивого и полезного примера остановимся на одной из таких задач — задаче о последовательностях де Брейна.

12.3.1. Формальная постановка задачи такова: найти наименьшую циклическую последовательность (циклическое слово) над алфавитом из n букв, содержащую все возможные подстроки длины k (так называемые k-меры).

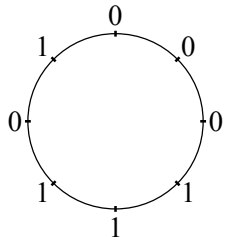


Рис. 61: Циклическая последовательность $B(2, 3)$

Пример 12.13. Рассмотрим циклические последовательности над алфавитом из двух букв — чисел 0 и 1, содержащие все возможные подпоследовательности длины три (тримеры). Таковых, как мы знаем, существует $2^3 = 8$ штук — восемь первых чисел в двоичной системе исчисления:

$$000, \quad 001, \quad 010, \quad 011, \quad 100, \quad 101, \quad 110, \quad 111.$$

Расставляя их по кругу, мы, конечно же, получаем циклическую последовательность, состоящую из $k \cdot n^k = 3 \cdot 2^3 = 24$ символов и содержащую все возможные тримеры. Однако такое слово минимальным не будет — несложно видеть, что изображенная на рис.61 циклическая последовательность длины $2^3 = 8$ содержит все тримеры, причем каждый из них она содержит по одному разу. Понятно, что циклическую последовательность длины меньшей, чем 2^3, построить невозможно — она не сможет содержать все 8 тримеров. Поэтому показанная на рис.61 циклическая последовательность минимальна.

Проведенные в примере 12.13 рассуждения позволяют предположить, что и в общем случае для произвольных n и k минимальная циклическая последовательность над алфавитом из n букв, содержащая все n^k возможных k-меров, также имеет длину n^k. В этой связи возникают сразу три вопроса — как доказать это предположение, как понять, сколько таких минимальных строк существует, ну и наконец, как их всех найти.

На все эти вопросы дал ответ Николас де Брейн в своей работе [**?**] 1946 года. Во-первых, он доказал, что действительно для произвольных n и k существуют циклические последовательности длины n^k над алфавитом из n букв, содержащие все возможные k-меры — в его честь такие циклические последовательности называют теперь *последовательностями де Брейна* $B(n,k)$ порядка k. Во-вторых, он подсчитал количество таких последовательностей. Наконец, он указал конструктивный алгоритм построения этих последовательностей. И сделал он это, построив для заданных n и k некоторый орграф специального вида.

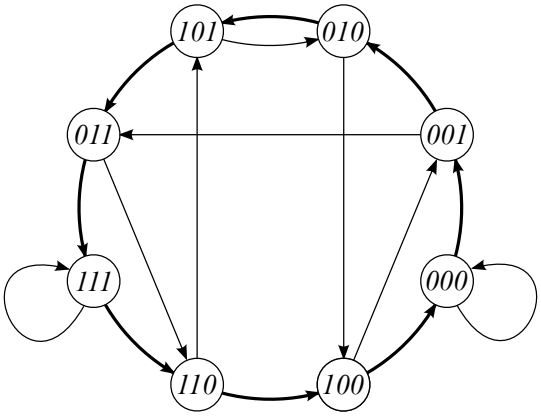


Рис. 62: Построение циклической последовательности с помощью гамильтонова цикла в графе

12.3.2. Итак, попытаемся, вслед за де Брейном, сопоставить множеству всевозможных k-меров над алфавитом из n букв некоторый орграф D, обход которого даст нам какую-то циклическую последовательность $B(n,k)$ длины n^k. Как правило, первое, что приходит в голову при анализе этой задачи, это взять в качестве вершин будущего графа все k-меры, и соединить любые две вершины направленным ребром в том случае, если $(k-1)$-мер, отвечающий суффиксу первой вершины, совпадает с $(k-1)$-мером, соответствующим префиксу второй.

Так, на рис.62 показан орграф, построенный с помощью этого алгоритма для разобранного выше примера 12.13. В этом орграфе, к примеру, вершина 011 соединена с вершиной 110 ребром потому, что суффикс первой из них — 2-мер 11 — совпадает с префиксом второй.

Заметим теперь, что обход этого орграфа по простому циклу, содержащему все вершины (помеченный жирными стрелками на рис.62 цикл), позволяет построить одну из искомых нами циклических последовательностей $B(2,3)$, а именно, последовательность 11101000. Тот же факт справедлив и в общем случае. Иными словами, при таком подходе задача поиска циклической последовательности $B(n,k)$ сводится к построению гамильтонова цикла в орграфе D.

12.3.3. На первый взгляд кажется, что мы решили поставленную задачу — действительно, нам удалось формализовать задачу поиска последовательностей де Брейна на языке теории графов. Однако у такого решения имеется множество недостатков. Во-первых, априори совсем не очевидно, что гамильтонов цикл у любого такого графа существует. Во-вторых, даже если он и существует, то непонятно, как его там искать — как мы знаем, задача построения гамильтонова цикла весьма нетривиальна. Наконец, если такие циклы и существуют, то не ясно, как подсчитать их количество.

Основное достижение де Брейна состояло в том, что он предложил другой, далеко не столь очевидный, подход к формализации данной задачи. Как показал де Брейн, эту задачу можно изящно переформулировать, сведя ее к к сравнительно легко решаемой задаче построения эйлерова цикла в орграфе специального вида — графе де Брейна.

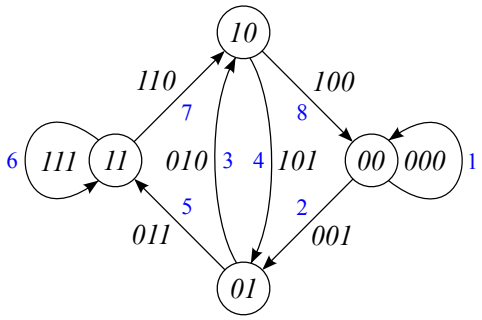


Рис. 63: Граф де Брейна для $n = 2$, $k = 3$

Алгоритм построения графа де Брейна следующий: возьмем в качестве вершин орграфа вместо k-меров все возможные $(k-1)$-меры (их, очевидно, $n^{(k-1)}$ штук) и свяжем любые две из них ориентированным ребром в случае, если существует k-мер, префиксом которого является первая вершина, а суффиксом — вторая.

На рис.63 показан граф де Брейна, построенный для данных из примера 12.13. Так как существуют тримеры 010 и 011, то вершина 01 соединена в этом графе с вершинами 10 и 11. Двум входящим в эту вершину ребрам отвечают тримеры 101 и 001. Так как к любому 2-меру можно двумя способами слева приписать единицу или ноль, и справа приписать единицу и ноль, то аналогичный факт справедлив и для любой другой вершины — в любую вершину входят ровно $n = 2$ ребра и выходят из этой вершины также ровно $n = 2$ ребра. Как следствие, в таком орграфе обязательно существует эйлеров цикл. Обход графа по эйлерову циклу 000, 001, 010, 101, 011, 111, 110, 100 позволяет восстановить искомую циклическую последовательность $B(2,3) = 00010111$.

Аналогичные рассуждения проходят и в общем случае. Следовательно, любой граф де Брейна является эйлеровым, а любой эйлеров цикл в нем отвечает некоторой последовательности де Брейна $B(n,k)$.

Итак, построив для любых n и k граф де Брейна, мы доказали существование соответствующей последовательности де Брейна. Далее, используя алгоритмы поиска эйлеровых циклов в орграфе, мы можем находить такие последовательности. Осталось понять, сколько всего таковых последовательностей существует. Де Брейн ответил и на этот вопрос, доказав, что их количество равно $(n!)^{n^{k-1}}/n^k$. Два различных доказательства этого результата — комбинаторное и алгебраическое — можно найти, например, в учебнике [**?**].

12.3.4. Как это часто бывает в прикладной математике, задача, очень похожая на рассмотренную выше, возникла относительно недавно еще в одной, достаточно молодой области прикладной математики — в биоинформатике [**?**]. Одной из наиболее актуальных задач в этой науке является задача ассемблирования (сборки) геномов из так называемых ридов (reads) — относительно коротких (содержащих порядка 100 символов — нуклеотидов) строк над 4-буквенным алфавитом {A,C,G,T}, получаемых в результате секвенирования (разделения) генома (а точнее, очень большого количества одинаковых геномов). Первые методы сборки генома из таких ридов как раз и базировались на построении графа, вершинам которого сопоставлялись риды, а ребрам — перекрытия между этими ридами фиксированной длины. При этом исходный геном восстанавливался с помощью построения гамильтонова цикла в подобном графе.

Рассмотрим в качестве простейшего примера очень короткий циклический геном, показанный на рис.64,а. Предположим, что в результате секвенирования мы получили из него пять ри-

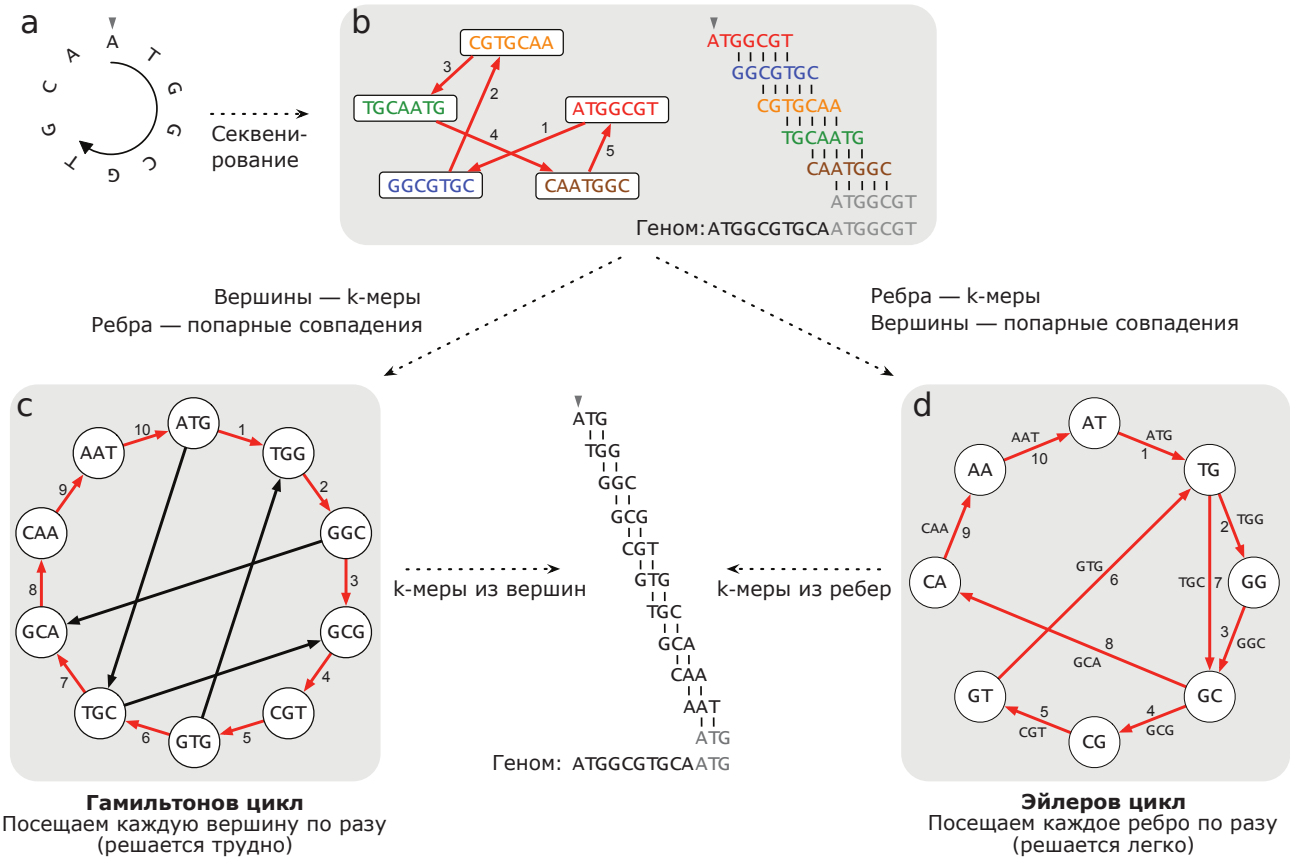


Рис. 64: Сборка простейшего генома

дов CGTGCAA, ATGGCGT, CAATGGC, GGCGTGC и TGCAATG длины 7. Соответствующий этой последовательности ридов граф показан на рис.64,b. Каждому из пяти ридов поставлена в соответствие одна из вершин этого графа. Две вершины соединяются ребром в случае, если ширина перекрытия соответствующих ридов составляет пять нуклеотидов (см. рис.64,b). Проход по гамильтонову циклу

$$\text{ATGGCGT} \to \text{GGCGTGC} \to \text{CGTGCAA} \to \text{TGCAATG} \to \text{CAATGGC} \to \text{ATGGCGT}$$

позволяет путем объединения первых двух нуклеотидов в каждом риде восстановить исходный геном ATGGCGTGCA.

Более современные методы сборки геномов обычно работают со строками определенной длины k (которые как раз и называются k-мерами), значительно более короткими, нежели исходные риды. Например, типичный 100-нуклеотидный рид разбивается вначале на 55-меры, длина перекрывающихся участков которых равна сорока шести. В нашем модельном примере каждый 7-нуклеотидный рид разбивается на пять 3-меров, перекрывающихся между собой по двум нуклеотидам (см. рис.64,c,d). Даже для этого примера найти соответствующий исходному геному гамильтонов цикл нелегко. В реальной же ситуации из одного генома в процессе секвенирования получают миллионы (10^6) ридов, триллионы (10^{12}) k-меров, то есть графы с огромным количеством вершин. Задача поиска гамильтонова цикла в таком графе практически нерешаема.

Павел Певзнер в 1989 году предложил для таких случаев использовать подход де Брейна. Переход от задачи поиска гамильтонова цикла в графе (рис.64,c) к задаче поиска эйлерова цикла (рис.64,d) существенным образом ускорил процесс ассемблирования геномов и стал общеприня-

тым в большинстве современных ассемблеров, предназначенных для сборки генома из коротких ридов.

13 Гамильтоновы циклы

13.1. Во многих практических задачах наряду с эйлеровыми циклами часто встречаются и так называемые *гамильтоновы циклы* — простые циклы, проходящие через каждую вершину графа.

13.1.1. Пожалуй, наиболее известная из таких задач — это так называемая *задача о коммивояжере*. В этой задаче торговец должен обойти все города из некоторого списка, заходя в каждый город только один раз, и вернуться в исходный город, с которого он начал свое путешествие. Обычно при этом указывается некоторый критерий оптимальности маршрута (кратчайший, самый дешевый и прочее). В случае, когда дополнительные критерии не указаны, задача сводится к поиску гамильтонова цикла в графе, вершинами которого являются города, а ребрами — соединяющие их дороги.

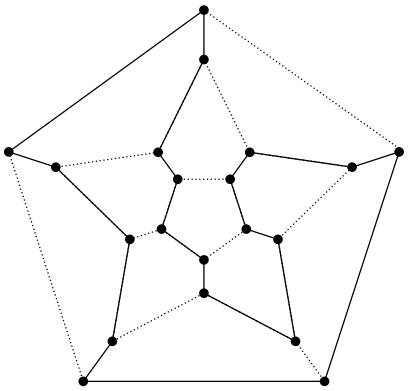


Рис. 65: Гамильтонов цикл в додекаэдре

Еще одна задача, собственно, и дала имя гамильтонову циклу — она была описана в письме Гамильтона своему другу в форме математической игры на додекаэдре (рис.65). В этой игре один из игроков должен был вставить палочки в любые пять последовательно идущих вершин додэкаэдра, а второй — продолжить этот путь на все оставшиеся вершины. Иными словами, игроки должны были найти простой цикл, проходящий через все вершины додекаэдра. Одно из возможных решений показано на рис.65 (сплошные линии на рисунке).

13.1.2. Сразу заметим, что наличие петель и мультиребер на существование гамильтонова цикла в графе G никак не влияет. Поэтому далее мы будем полагать, что любые рассматриваемые в этом параграфе графы являются простыми.

13.1.3. Как и в случае эйлерова цикла, первый вопрос, который возникает при анализе подобного рода задач, связан с существованием гамильтонова цикла в заданном графе. Очевидными необходимыми условиями существования гамильтонова цикла в графе G является связность этого графа, а также отсутствие в нем вершин степени $\deg(x) = 1$, то есть листьев. Чуть менее тривиальным является следующее необходимое условие существования гамильтонова цикла в графе.

Утверждение 13.1. *Пусть в графе G имеется гамильтонов цикл. Тогда количество $k := c(G-S)$ компонент связности $U_1, \ldots, U_k$, получающихся в результате удаления вершин некоторого непустого подмножества $S \subset V(G)$ графа G, не превосходит количества удаленных вершин:*

$$c(G-S) \leqslant |S|. \tag{42}$$

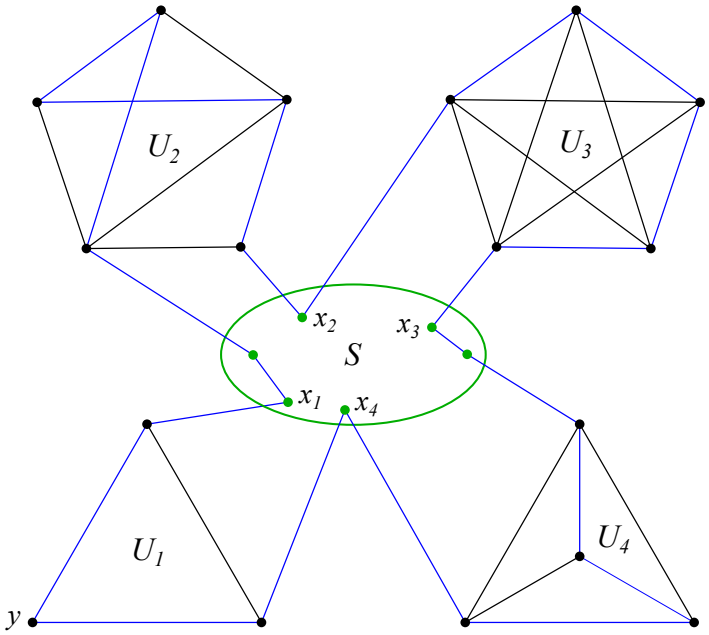


Рис. 66

Доказательство. Начнем обход графа G по гамильтонову циклу C с произвольной вершины y компоненты U_1 (рис.66). Выйти из этой компоненты U_1 в другие компоненты U_i, $i \neq 1$, мы не можем, поэтому, выходя из U_1, мы должны прийти в какую-то вершину $x_1 \in S$. Аналогично, выходя из оставшихся компонент U_i, мы должны прийти в какие-то вершины $x_i \in S$. Так как мы обходим граф G по гамильтонову циклу C, то все вершины $x_i \in S$ должны быть различными. Как следствие, количество вершин в множестве S должно быть больше или равно k. □

13.1.4. Основная проблема со всеми известными на сегодняшний момент необходимыми условиями существования гамильтонова цикла в графе состоит в том, что ни одно из них не является одновременно и достаточным. Конечно же, существуют графы, для которых вопрос о существовании гамильтонова цикла очевиден. Так, в любом циклическом графе C_n, построенном на $n > 2$ вершинах, существует ровно один гамильтонов цикл. В графе K_2 гамильтонов цикл не существует (хотя существует *гамильтонов путь* — простой путь, проходящий через все вершины графа). В случае полного графа K_n, $n > 2$, имеется $(n-1)!/2$ гамильтоновых циклов (смотри упражнение **??**).

В общем же случае ответ на вопрос, существует ли в данном графе гамильтонов цикл, совершенно нетривиален. В частности, на настоящий момент нет никаких простых критериев существования гамильтонова цикла, подобных тому, что мы сформулировали выше для, казалось бы, очень похожего понятия эйлерова цикла. Более того, в 1972 году Ричард Карп доказал, что задача определения того, существует ли в произвольном графе гамильтонов цикл, является NP-полной задачей.

Несмотря на это печальное обстоятельство, в теории графов все же имеется целый ряд достаточных условий существования гамильтонова пути или цикла в графе. Так, интуитивно

понятно, что чем больше у простого связного графа, построенного на n вершинах, ребер, тем больше вероятность того, что в нем существует гамильтонов цикл. Это видно хотя бы из того, что в полном графе K_n, количество ребер в котором максимально, гамильтонов цикл гарантированно существует. Ниже мы сформулируем несколько результатов, которые формализуют это интуитивное наблюдение.

13.1.5. Предположим вначале, что мы смогли каким-то образом построить в простом графе G гамильтонов путь $P = x_1, \ldots, x_n$, $n > 2$, то есть простой путь, проходящий через каждую из вершин графа G. Возникает вопрос, когда мы этот путь можем достроить до гамильтонова цикла C. Ответ очевиден в том случае, когда концы построенного пути — вершины x_1 и x_n — оказываются смежными. В этом случае мы всегда можем достроить путь P до гамильтонова цикла C, добавив к нему ребро $\{x_1, x_n\}$. Случай несмежных вершин x_1 и x_n является уже не столь очевидным.

Лемма 13.2. *Пусть в простом графе G имеется гамильтонов путь $P = x_1, \ldots, x_n$, $n > 2$, соединяющий пару несмежных вершин x_1 и x_n. Достаточным условием существования гамильтонова цикла в таком графе является выполнение следующего неравенства:*

$$\deg(x_1) + \deg(x_n) \geqslant n. \tag{43}$$

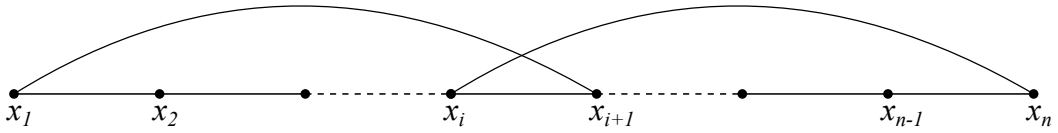


Рис. 67: Построение гамильтонова цикла в графе

Доказательство. Основная идея доказательства этого утверждения довольно проста. Пусть в графе G существует гамильтонов путь

$$P = (x_1, x_2, \ldots, x_n),$$

соединяющий несмежные между собой вершины x_1 и x_n. Нам нужно показать, что в G обязательно существует принадлежаще этому пути ребро $e = \{x_i, x_{i+1}\}$, такое, что вершина x_{i+1} смежна с x_1, а вершина x_i смежна с x_n (рис.67). Тогда мы всегда сможем заменить путь P на гамильтонов цикл

$$C = (x_1, x_2, \ldots, x_i, x_n, x_{n-1}, \ldots, x_{i+1}, x_1).$$

Докажем, что такое ребро e обязательно найдется. Обозначим через l степень вершины x_1. Покажем, что среди *любых* l отличных от x_n вершин графа G найдется хотя бы одна вершина, смежная с x_n. Действительно, предполагая противное, мы получаем неравенство $\deg(x_n) \leqslant n - 1 - l$ — помимо x_n в графе G имеется $n - 1$ вершина, и l из них гарантированно не смежны с x_n. Это, в свою очередь, противоречит неравенству (43).

Теперь возьмем l смежных с x_1 вершин графа G и отступим от этих вершин на одно ребро назад вдоль гамильтонова пути P. Получим набор из l вершин, одна из которых — какая-то вершина x_i — обязательно смежна с x_n. Тогда ребро $\{x_i, x_{i+1}\}$ и будет нужным нам ребром e. □

13.1.6. Заметим, что на самом деле мы доказали чуть более общий результат, а именно:

Следствие 13.3. *Пусть $P = x_1, \ldots, x_k$, $k > 2$, есть наибольший по включению простой путь в графе G. Тогда этот путь можно превратить в простой цикл C либо в случае, когда концы*

пути P — вершины x_1 и x_k — являются смежными, либо в случае, когда сумма степеней этих вершин больше или равна k:

$$\deg(x_1) + \deg(x_k) \geqslant k. \tag{44}$$

Доказательство. Действительно, в случае, когда вершины x_1 и x_k являются смежными, искомый цикл C получается добавлением к пути P ребра $\{x_1, x_k\}$.

Предположим, что вершины x_1 и x_k оказались несмежными. Заметим, что эти вершины соединены ребрами только с какими-то другими вершинами того же пути — в противном случае мы смогли бы продолжить путь P на какие-то другие вершины.

Рассмотрим подграф H, индуцируемый всеми вершинами пути P. Согласно сделанному выше замечанию, степени вершин x_1 и x_k в этом подграфе останутся прежними. Как следствие, неравенство (44) для этих вершин окажется верным и в подграфе H. Но в этом подграфе путь P является гамильтоновым. Следовательно, его можно превратить в гамильтонов цикл C способом, описанным при доказательстве леммы 13.2. □

13.1.7. Следующая теорема дает нам достаточные условия существования гамильтонова пути в графе.

Теорема 13.4 (Оре)**.** *Пусть G — простой граф, построенный на $n > 2$ вершинах. Если для любых двух несмежных вершин x, y графа G выполняется условие*

$$\deg(x) + \deg(y) \geqslant n - 1, \tag{45}$$

то граф G имеет гамильтонов путь.

Доказательство. Заметим, прежде всего, что любой граф G, удовлетворяющий условиям (45), является связным. Более того, длина кратчайшего пути, соединяющего произвольную пару несмежных вершин, равна двум. Действительно, рассмотрим произвольную пару несмежных вершин x и y. Условие (45) вместе с принципом Дирихле гарантирует нам, что среди оставшихся $(n-2)$-х вершин обязательно найдется хотя бы одна вершина z, смежная как с x, так и с y, то есть в графе существует путь $\{x, z, y\}$.

Теперь предположим, что у нас выполнено условие (45), а гамильтонов путь в графе G не существует. Это означает, что максимальный простой путь P в таком графе содержит $k < n$ вершин. Ясно также, что длина такого пути больше или равна двум — если в графе существует пара несмежных вершин, то она соединена путем длины два, а если все вершины графа смежны между собой, то граф — полный, и там длина максимального пути равна $n - 1$. Наконец, если концевые вершины пути P не смежны, то в силу (45) сумма их степеней больше или равна $n-1$, то есть больше или равна k. Все это дает нам основание воспользоваться следствием 13.3 и утверждать, что в графе G существует простой цикл длины k.

Так как $k < n$ и граф G связен, то в G обязана существовать вершина y, не входящая в цикл C и смежная хотя бы с одной из вершин $x \equiv x_1$ этого цикла $C = (x_1, \ldots, x_k)$. Но в таком случае мы всегда можем построить в графе G простой путь $y, x_1, x_2, \ldots, x_k$ длины, большей, чем k, что противоречит предположению о том, что P есть путь максимальной длины. Полученное противоречие доказывает теорему. □

13.1.8. Сформулируем теперь несколько очевидных следствий из теоремы Оре.

Следствие 13.5. *Пусть G — граф, построенный на $n > 2$ вершинах. Если для любой пары несмежных вершин выполняется условие*

$$\deg(x) + \deg(y) \geqslant n, \tag{46}$$

то в графе G имеется гамильтонов цикл.

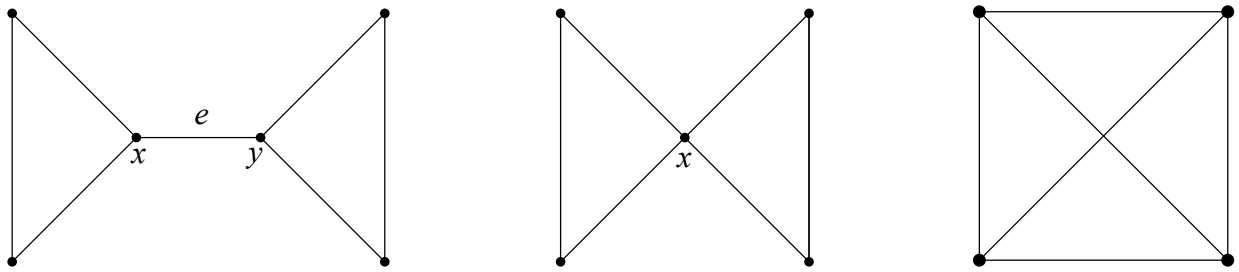
Доказательство. Действительно, в силу теоремы Оре, в таком графе обязан существовать гамильтонов путь $P = x_1, \ldots, x_n$. Согласно следствию 13.3, его всегда можно превратить в гамильтонов цикл. □

Следствие 13.6 (Дирак)**.** *Пусть G — простой граф на $n > 2$ вершинах. Если степень каждой из его вершин больше или равна $(n-1)/2$, то в графе существует гамильтонов путь, а если больше или равна $n/2$ — то в нем существует и гамильтонов цикл.*

Связность в графах

14 Вершинная и реберная связность графа

14.1. Любая коммуникационная сеть тем более надежна, чем большее количество вершин и/или ребер нужно удалить в графе (или в орграфе), моделирующем эту сеть, для того, чтобы эту сеть оборвать. Раздел теории графов, описывающий такого рода свойства надежности, называется теорией связности. К изучению этой теории мы и приступаем в данном параграфе.



(a) Граф G_1 (b) Граф G_2 (c) Граф G_3

Рис. 68

14.1.1. Рассмотрим в качестве примера три графа, изображенные на рис.68. Видно, что с точки зрения надежности все эти три графа достаточно сильно отличаются друг от друга. Показанный на рис.68,a граф G_1 наименее надежен — его можно превратить в несвязный удалением как одной из двух вершин x или y, так и удалением ребра $e = \{x, y\}$ (моста графа G_1). Граф G_2 (рис.68,b) с этой точки зрения выглядит получше — у него нет ни одного ребра, удалив которое, мы развалим G_2 на несколько связных компонент. Однако у него имеется вершина x (точка сочленения графа G_2), удаление которой приводит к потере связности графа. Наконец, граф G_3 с рассматриваемой точки зрения является наиболее надежным — у него нет ни мостов, ни точек сочленения, удаление которых приводит к потере связности графа.

Про граф G_1 говорят, что он реберно и вершинно односвязен, про граф G_2 — что он вершинно односвязен, но реберно двусвязен, а про граф G_3 — что он реберно и вершинно двусвязен. Наша задача состоит в том, чтобы дать формальное определение этих понятий.

14.1.2. Начнем мы с понятия реберной связности графа G. Напомним, что в прошлой главе мы дали понятие реберного разделяющего множества F, реберного разреза ∂X, а также минимального реберного разреза. Выберем среди всех минимальных реберных разрезов графа G тот, количество ребер в котором минимально. Количество ребер в таком разрезе и называется реберной связностью $\lambda(G)$. Иными словами, мы можем дать следующее определение.

Определение 14.1. *Реберной связностью* $\lambda(G)$ графа G называется минимальное количество ребер, которое нужно удалить в графе G для того, чтобы он стал несвязным.

В случае несвязного графа или графа K_1 реберная связность считается равной нулю.

14.1.3. Реберную связность можно определить и с помощью еще одного важного понятия — реберно k-связного графа.

Определение 14.2. Граф G, построенный на n вершинах, называется *реберно k-связным,* если он остается связным при удалении любых ребер в количестве, строго меньшем k.

Например, любой граф, в котором имеется мост, является реберно односвязным графом, но реберно двусвязным он уже не является. Любой цикл C_n является реберно односвязным и реберно двусвязным, однако реберно трехсвязным он уже не является. Наконец, полный граф K_n в случае $n > 1$ является реберно $(n-1)$-связным графом.

Ясно, что любой реберно k-связный граф является также реберно i-связным для любых i, $0 \leqslant i \leqslant k$.

Определение 14.3. Наибольшее значение k, при котором граф G все еще остается реберно k-связным, называется реберной связностью $\lambda(G)$ графа.

14.1.4. Перейдем теперь к определению вершинной связности графа G.

Определение 14.4. Подмножество $S \subset V(G)$ называется *вершинно* разделяющим множеством или *вершинным разрезом* графа, если при удалении вершин из множества S граф $G - S$ становится несвязным.

Казалось бы, по аналогии с реберной связностью мы можем определить вершинную связность как количество вершин в минимальном вершинно разделяющем множестве S. Однако такое определение не охватывает важный частный случай полного графа K_n. Действительно, у полного графа K_n, построенного на $n > 0$ вершинах, вершинно разделяющее множество отсутствует — удаление любого количества вершин $|S|$ в этом графе не нарушает его связности. Поэтому вершинную связность нам нужно определить чуть более аккуратно.

Определение 14.5. *Вершинной связностью* $\kappa(G)$ называется минимальное количество вершин $|S|$, $S \subset V(G)$, которое нужно удалить для того, чтобы граф $G - S$ стал несвязным или же содержал единственную вершину.

Как следствие, вершинная связность любого полного графа K_n равна $\kappa(K_n) = n - 1$. В частности, граф K_1, состоящий из единственной изолированной вершины, имеет связность, равную нулю. Ребро K_2 имеет связность, равную единице. Полный граф K_3 на трех вершинах имеет связность, равную двум, а дерево T_3 — связность $\kappa(T_3) = 1$. Первый нетривиальный простой граф с $\kappa = 2$ — это цикл $_n$.

14.1.5. Связность можно также определить, используя понятие (вершинно) k-связного графа.

Определение 14.6. Граф G называется (вершинно) k-связным, если он построен на $n \geqslant (k+1)$ вершине, и при удалении любых вершин графа G в количестве, меньшем, чем k, граф остается связным.

Определение 14.7. Вершинной связностью $\kappa(G)$ графа G называется максимально возможное k, для которого граф остается вершинно k-связным.

Пример 14.8. Граф K_4 с $\kappa(K_4) = 3$ является 1-связным, 2-связным и 3-связным, но не 4-связным графом, хотя бы потому, что он построен на четырех вершинах.

14.2. Наша следующая задача состоит в том, чтобы получить какие-то простейшие оценки на числа $\lambda(G)$ и $\kappa(G)$.

14.2.1. Обозначим через $\delta(G)$ минимальную из степеней вершин графа G. Достаточно очевидно, что реберная связность графа ограничена сверху значением $\delta(G)$. Действительно, удаление всех ребер, инцидентных вершине x с $\deg(x) = \lambda(G)$, превращает любой граф в несвязный.

Несколько менее тривиальным является следующее

Утверждение 14.9. *Для любого простого графа G справедливо неравенство*

$$\kappa(G) \leqslant \lambda(G).$$

Доказательство. Рассмотрим некоторый минимальный реберный разрез $[S, \bar{S}] = \{e_1, \ldots, e_k\}$, $k = \lambda(G)$, в графе G. Нам нужно показать, что в графе G найдется не более чем k вершин, удаление которых приведет либо к потере связности, либо к появлению графа K_1.

На первый взгляд кажется, что данное утверждение доказать несложно. Действительно, удалим для каждого ребра e_i из разреза $[S, \bar{S}]$ одну из двух инцидентных этому ребру вершин. Удаляя такую вершину, мы удаляем и все инцидентные ей ребра, в том числе и ребро e_i. Заметив теперь, что количество таких вершин не превосходит количества ребер в множестве $[S, \bar{S}]$, мы, казалось бы, доказываем требуемое неравенство.

Проблема, однако, состоит в том, что удаление таких вершин не всегда приводит к появлению несвязных компонент в графе. Действительно, рассмотрим, к примеру, дерево T_n, построенное на $n \geqslant 2$ вершинах. Понятно, что для него $\lambda(T_n) = \kappa(T_n) = 1$. Удаление любого ребра e приводит к появлению двух компонент связности в графе $T_n - e$. Однако, удаляя в этом графе произвольный лист x, мы получим односвязный граф $T_n - x = T_{n-1}$. Поэтому для доказательства неравенства $\kappa(G) \leqslant \lambda(G)$ нам нужно действовать чуть более осторожно.

Именно, в случае, если граф G — несвязен, либо если он состоит из одной вершины, то, по определению, $\kappa(G) = \lambda(G) = 0$. Если граф G — связен и имеет мост, то $\lambda(G) = 1$. Единице равняется в этом случае и вершинная связаность. Действительно, либо $G = K_2$, и тогда $\kappa(G) = = 1$, либо одна из двух инцидентных мосту вершин является точкой сочленения.

Наконец, предположим, что $\lambda(G) \geqslant 2$. Удаляя $\lambda(G) - 1$ ребер, мы получаем связный граф, имеющий мост $e = \{x, y\}$. Для любого из $\lambda(G) - 1$ ребер выберем инцидентную ему вершину, отличную от x и y. Если удаление таких вершин приведет к несвязному графу, то $\kappa(G) \leqslant \lambda(G) - - 1$. В противном случае у нас останется мост $\{x, y\}$. Удаление одной из вершин x, y приведет либо к потере связности, либо к тривиальному графу K_1. В обоих случаях мы получаем, что $\kappa(G) \leqslant \lambda(G)$. □

14.2.2. Таким образом, мы для произвольных графов доказали справедливость цепочки неравенств

$$\kappa(G) \leqslant \lambda(G) \leqslant \delta(G).$$

Простейший пример графа G, для которого эти неравенства являются строгими, показан на рис.69. Действительно, в таком графе $\kappa(G) = 1$, $\lambda(G) = 2$, а $\delta(G) = 3$. Для полного графа, с другой стороны, все эти числа одинаковы и равны $(n - 1)$.

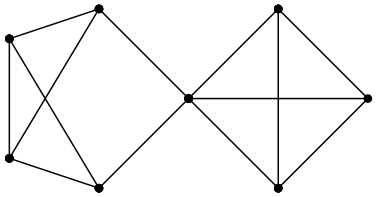


Рис. 69

15 Двусвязные графы

15.1. Вернемся к описанию вершинно односвязных графов. Все множество таких графов можно разбить на два блока, отнеся к первому блоку графы с $\kappa(G) = 1$, а ко второму блоку — все двусвязные графы. Мы знаем, что двусвязный граф — это граф, в котором отсутствуют точки сочленения. Исследуем вначале структуру графов с $\kappa(G) = 1$.

15.1.1. Напомним, что множество вершин любого графа G естественным образом разбивается на блоки — компоненты связности такого графа. Оказывается, что в случае $\kappa(G) = 1$ мы можем разбить на блоки множество $E(G)$ ребер графа G, введя очень полезное отношение похожести.

Определение 15.1. Два ребра называются *похожими,* если они либо совпадают (то есть любое ребро похоже на само себя), либо входят в один и тот же простой цикл.

Лемма 15.2. *Отношение похожести в связном графе G есть отношение эквивалентности.*

Доказательство. Рефлексивность и симметричность такого отношения очевидны, так что нам остается проверить транзитивность. Пусть ребра e_1, e_2 входят в простой цикл C_1, а ребра e_2, e_3 — в простой цикл C_2 графа G. Возьмем ребро e_1 и будем идти от него вдоль цикла C_1 в двух противоположных направлениях до тех пор, пока не дойдем до вершин x и y, $x \neq y$, принадлежащих циклу C_2 (смотри рисунок 70). Такие вершины обязательно найдутся хотя бы потому, что ребро e_2 входит в цикл C_2, так что две инцидентные этому ребру вершины обязательно этому циклу C_2 принадлежат. Заметим теперь, что вершины x и y разбивают цикл C_2 на два простых пути, соединяющих данные вершины. Один из этих простых путей содержит ребро e_3. Объединяя этот простой путь с куском цикла C_1, содержащим ребро e_1 и ограниченным вершинами x и y, получим простой цикл C, содержащий ребра e_1 и e_3. Тем самым мы доказали, что ребра e_1 и e_3 похожи. □

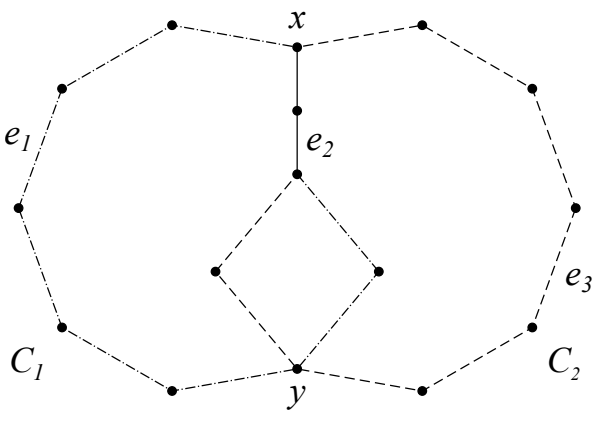


Рис. 70

15.1.2. Отношение похожести, как и любое отношение эквивалентности, разбивает множество $E(G)$ ребер на классы эквивалентности, называемые *блоками* связного графа G. Каждый такой блок представляет собой либо простое ребро K_2, либо максимальный по включению блок, состоящий из нескольких ребер, любые два из которых принадлежат некоторому простому циклу. Оказывается, что такая группа ребер есть не что иное, как двусвязный подграф графа G. Именно, справедлива

Теорема 15.3. *Пусть G есть связный граф, построенный на $n \geqslant 3$ вершинах. Следующие три утверждения равносильны:*

(1) граф G является двусвязным;
(2) любые два ребра этого графа принадлежат некоторому циклу C;
(3) для любых двух вершин графа G существует цикл C, проходящий через эти вершины.

Доказательство. Предположим вначале, что граф G является двусвязным. Заметим сразу же, что в силу неравенства $\lambda(G) \leqslant \delta(G)$ степень любой вершины в таком графе больше или равна двум. Рассмотрим теперь произвольную вершину $x \in V(G)$, а также два произвольных инцидентных этой вершине ребра $e_1 = \{x, y\}$ и $e_2 = \{x, z\}$. Так как граф $G - x$ является связным, то существует простой путь P, соединяющий вершины y и z. Объединяя путь P с ребрами e_1 и e_2, получаем простой цикл C в графе G.

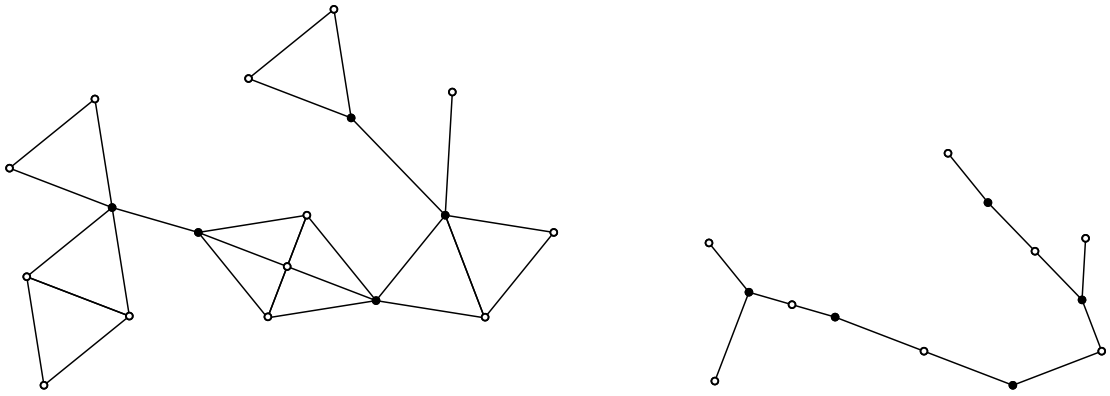
Таким образом, любые два ребра, инцидентные одной и той же вершине двусвязного графа, принадлежат одному и тому же циклу. Иными словами, эти два ребра являются похожими. Но отношение похожести транзитивно, поэтому из похожести любых двух соседних ребер следует похожесть любых двух ребер графа G.

Пусть теперь любые два ребра графа G принадлежат некоторому циклу C. Выберем любые две вершины x_1, x_2 нашего графа и рассмотрим два различных ребра e_1 и e_2, инцидентных этим вершинам. В силу неравенства $n \geqslant 3$ такие ребра обязательно найдутся. Цикл C, содержащий ребра e_1 и e_2, обязательно проходит через выбранные нами вершины x_1 и x_2.

Последнее, что осталось — это доказать, что если через любые две вершины графа G проходит цикл C, то граф является двусвязным, то есть не содержит точек сочленения. А это утверждение достаточно очевидно. Действительно, пусть такой граф содержит точку сочленения $x \in V(G)$. Обозначим через G_1 и G_2 две компоненты связности, полученные в результате удаления вершины x, а через $y_1 \in G_1$ и $y_2 \in G_2$ — пару вершин, смежных вершине x в исходном графе. Так как в графе $G - x$ нет пути, соединяющего вершины y_1 и y_2, то и в исходном графе G нет цикла, проходящего через эти вершины. Полученное противоречие доказывает нашу теорему. □

15.1.3. Итак, с помощью отношения похожести мы разбили ребра графа G с $\kappa(G) = 1$ на блоки (рис.71,a), каждый из которых представляет собой либо мост, либо максимальный по включению (то есть не содержащийся ни в каком большем подграфе) двусвязный подграф графа G. Эти блоки соединены между собой точками сочленения (черные вершины на рис.71,a).

Используя описанное выше разбиение, по графу G можно построить двудольный граф $B(G)$ (рис.71,b), описывающий блочную структуру исходного графа G. Одна доля графа $B(G)$ состоит из точек x_j сочленения графа G (черные вершины на рис.71,b), а вторая — из вершин b_i, каждая из которых отвечает некоторому блоку B_i исходного графа G (светлые вершины на рис.71,b). Вершины x_j и b_i соединяются в графе $B(G)$ ребром в случае, если в исходном графе вершина x_j принадлежала блоку B_i.



(a) (b)

Рис. 71

Понятно, что в построенном таким образом двудольном графе $B(G)$ циклов быть не может, то есть граф $B(G)$ является деревом. Это дерево называется деревом блоков и точек сочленения исходного графа G. Любая точка сочленения обязательно является внутренней вершиной дерева $B(G)$, тогда как вершины b_i, отвечающие блокам дерева G, могут быть как внутренними вершинами $B(G)$, так и его листьями. Блок, отвечающий листу дерева $B(G)$, называется крайним блоком графа G. Понятно, что блок графа G является крайним тогда и только тогда, когда он содержит только одну точку сочленения графа G.

15.2. Опишем алгоритм поиска точек сочленения в односвязном графе, линейный по количеству вершин и ребер в графе. Данный алгоритм был предложен в 1973 году Джоном Хопрофтом и Робертом Тарьяном и базируется на алгоритме обхода графа в глубину.

15.2.1. Пусть G есть простой односвязный граф. В результате алгоритма поиска в глубину мы получаем дерево $T(G)$ обхода графа с корнем в вершине x_0 — вершине, с которой мы начинаем обход графа (рис.72). Как мы знаем, в результате работы алгоритма все ребра графа разбиваются на два класса — ребра, принадлежащие дереву обхода (ребра, помеченные синим цветом на рис.72), а также ребра, дереву не принадлежащие (ребра, помеченные черным цветом на рис.72) — так называемые обратные ребра. При этом, в силу нормальности такого дерева, один из концов обратного ребра обязательно лежит на пути, соединяющем второй конец с корнем x_0 остовного дерева.

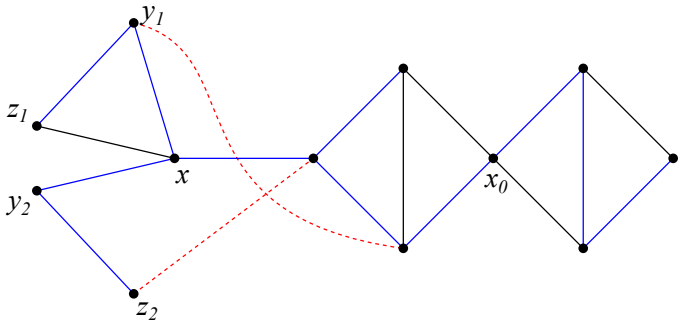


Рис. 72

Достаточно очевидно, что корень x_0 дерева $T(G)$ является точкой сочленения тогда и только тогда, когда он имеет два или более потомка в дереве обхода (см. вершину x_0 на рис.72). Действительно, после удаления x_0 смежные с x_0 вершины, являющиеся потомками x_0 в дереве

$T(G)$, оказываются в разных компонентах связности графа $G - x_0$. Для того, чтобы понять, в каком случае вершина $x \neq x_0$ является точкой сочленения, рассмотрим подмножество y_i непосредственных потомков (детей) вершины x в дереве $T(G)$ обхода графа G (см.рис.72). Предположим, что *хотя бы для одной* вершины y_i обратные ребра, исходящие из поддерева T_i с корнем в вершине y_i, либо вовсе отсутствуют (вершина y_2 на рис.72), либо оканчиваются в вершине x или в вершинах самого поддерева T_i (вершина y_1 на рис.72). В этом случае при удалении x граф G разваливается как минимум на две компоненты связности, то есть x является точкой сочленения. Если же в G существовали бы обратные ребра, исходящие, например, из точек z_2 и y_1 и входящие в предки вершины x (см. пунктирные ребра, помеченные красным цветом на рис.72), то вершина x точкой сочленения бы не являлась.

15.2.2. Описанные выше наблюдения можно положить в основу алгоритма поиска точек сочленения. Именно, будем обходить граф G в глубину, помечая вершины графа в порядке этого обхода натуральными числами $k(x)$ из диапазона от 1 до $n = |V(G)|$. Наряду с $k(x)$ введем для каждой вершины графа функцию $l(x)$ следующим образом. Для стартовой вершины x_0 с $k(x_0) = 1$ положим $l(x_0) = 1$. Для всех остальных вершин $l(x)$ определим следующим образом:

$$l(x) = \min \begin{cases} k(x), & \\ l(y_i), & \text{где } y_i \text{ — непосредственные потомки вершины } x \text{ в дереве } T(G), \\ k(z_j), & \text{где } z_j \text{ — предки вершины } x \text{ в } T(G), \text{ соединенные с } x \text{ обратным ребром в } G. \end{cases}$$

Функция $l(x)$ для вершины x может быть сосчитана только лишь после посещения всех вершин, являющихся потомками x в дереве $T(G)$.

Предположим теперь, что вершина $x \neq x_0$ является точкой сочленения. Это означает, что среди непосредственных потомков y_i вершины x в дереве $T(G)$ найдется хотя бы одна вершина y_i, для которой все исходящие из вершин поддерева T_{y_i} обратные ребра не поднимаются выше вершины x в дереве $T(G)$. Тогда, как видно из определения функции l, значения l на вершинах поддерева T_{y_i} ограничены снизу значением $k(x)$.

Наоборот, пусть вершина $x \neq x_0$ не является точкой сочленения. В таком случае для любого поддерева T_{y_i} найдется исходящее из него обратное ребро $\{v, z\}$, $v \in T_{y_i}$, приходящее в некоторую вершину z, являющуюся предком x в дереве $T(G)$. Наличие такого ребра $\{v, z\}$, в свою очередь, означает, что величина $l(v)$ оказывается меньше $k(x)$. Теперь остается заметить, что при движении по дереву T_{y_i} от вершины v к корню значения функции l могут разве что уменьшиться. Как следствие, в рассматриваемом случае мы имеем неравенство $l(y_i) < k(x)$ для любого непосредственного потомка y_i вершины x.

Таким образом, мы доказали, что вершина $x \neq x_0$ является точкой сочленения тогда и только тогда, когда хотя бы для одного из ее потомков y_i значение $l(y_i)$ оказывается большим или равным номеру $k(x)$ вершины x:

$$\textbf{if} \quad \exists\, y_i\colon \quad l(y_i) \geqslant k(x), \qquad \textbf{then} \quad x \neq x_0 \text{ is a cut point.}$$

Для корневой вершины x_0 достаточно подсчитать количество ее потомков в дереве $T(G)$. Если это количество больше единицы, то x_0 является точкой сочленения.

Результаты работы описанного выше алгоритма для графа G из рис.72 показаны на рис.73. Вершины этого графа помечены числами от 1 до 12 в порядке обхода этих вершин поиском в глубину. В скобках для каждой вершины i стоит соответствующее ей число $l(i)$. Красным цветом

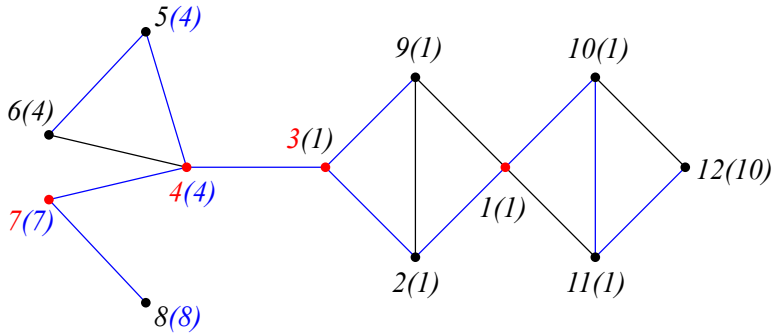


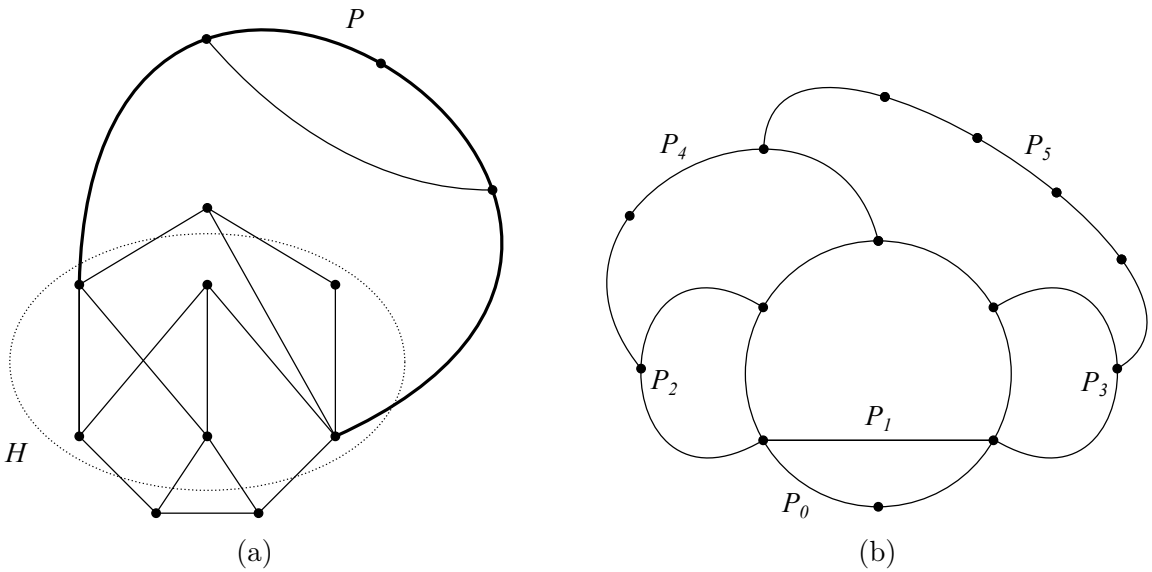
Рис. 73

на этом рисунке помечены точки сочленения графа G. Действительно, у вершины 7 имеется потомок — вершина 8, значение $l(8) = 8$ которой больше номера вершины 7. Аналогичный факт справедлив для вершин 4 и 3 — у них имеются непосредственные потомки y_i с $l(y_i)$, большими или равными номерам этих вершин. Наконец, у вершины 1 имеются два потомка в дереве $T(G)$, так что 1 есть точка сочленения G.

15.3. Ранее мы достаточно подробно описали структуру графов с $\kappa(G) = 1$. Постараемся теперь понять структуру вершинно двусвязного графа.

15.3.1. Оказывается, существует достаточно удобный конструктивный способ построения вершинно двусвязного графа — любой такой граф может быть построен из некоторого цикла $C = P_0$ последовательным добавлением к нему так называемых ручек P_i.

Определение 15.4. Пусть H есть некоторый подграф графа G. *Ручкой* подграфа H в графе G называется простой путь P, концы которого принадлежат H, а все внутренние вершины которого этому подграфу не принадлежат (смотри рис.74,a).



(a) (b)

Рис. 74

Определение 15.5. Разложением графа G на ручки называется последовательность $P_0, \ldots, P_k$ подграфов графа G, такая, что P_0 представляет собой цикл, P_i для любого $i = 1, \ldots, k$ представляет собой ручку для подграфа $G_{i-1} = P_0 \cup \ldots \cup P_{i-1}$ графа G, а G_k совпадает с исходным графом G (смотри рис.74,b).

Утверждение 15.6. *Граф G является двусвязным тогда и только тогда, когда он допускает разложение G на ручки, начинающееся с произвольного цикла в этом графе.*

Доказательство. Пусть граф G допускает разложение на ручки. Так как цикл P_0 является двусвязным графом, то нам достаточно показать, что добавление к графу G ручки P двусвязность графа $G \cup P$ не нарушит. Обозначим через x и y концевые вершины пути P. Заметим, что добавление ребра $\{x, y\}$ к графу G двусвязность не нарушит. Теперь заметим, что P представляет собой подразбиение ребра $\{x, y\}$ вершинами степени 2, а любое такое подразбиение также не нарушает двусвязность графа. Действительно, мы показали, что граф G двусвязен тогда и только тогда, когда любые два ребра принадлежат некоторому циклу, а последнее свойство, очевидно, в процессе подразбиения ребра $\{x, y\}$ не меняется.

Обратно, пусть G является двусвязным графом. Так как $2 = \kappa(G) \leqslant \delta(G)$, то в G обязательно найдется цикл C. Построим разложение на ручки графа G, начиная с цикла $C = P_0$. Для этого рассмотрим граф G_i, полученный добавлением i ручек. Если $G_i \neq G$, то в G найдется ребро $e \in G - E(G_i)$, а также ребро $e' \in E(G_i)$. Так как граф G двусвязный, то эти ребра принадлежат некоторому общему циклу. Обозначим через P ту часть этого цикла, которая содержит ребро e, а также ровно две вершины $x, y \in V(G_i)$. По отношению к G_i путь P является ручкой. Продолжая далее, мы и получим разложение G на ручки.

15.3.2. Перейдем теперь к описанию реберно двусвязных графов. Мы знаем, что любой реберно двусвязный граф — это связный граф, не содержащий мостов. Такое описание имеет один недостаток — оно показывает нам, чего в графе быть не должно для того, чтобы он был реберно двусвязным. Нам же часто нужно более конструктивное описание такого графа, похожее на то, что мы только что дали для вершинно двусвязного графа.

Заметим, прежде всего, что в силу неравенства $\kappa(G) \leqslant \lambda(G)$ любой вершинно двусвязный граф является одновременно и реберно двусвязным графом. Обратное, однако же, неверно. На рис.75 в качестве примера показан граф "бабочка", являющийся реберно двусвязным, но имеющим точку сочленения x. Иными словами, реберно двусвязных графов больше, нежели чем вершинно двусвязных графов. Как следствие, декомпозиция реберно двусвязного графа должна быть несколько более сложной по сравнению с декомпозицией вершинно двусвязного графа.

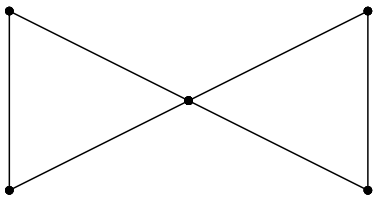


Рис. 75

Как видно из рис.75, реберно двусвязный подграф может содержать один или несколько циклов, имеющих с оставшейся частью графа только лишь одну общую точку. Такого рода цикл называется *замкнутой ручкой.* В упражнениях предлагается доказать, что любой реберно двусвязный граф можно разложить на ручки и замкнутые ручки, начиная с произвольного замкнутого цикла C в исходном графе.

16 k-связные графы. Теорема Менгера

16.1. В предыдущих параграфах мы довольно подробно изучили структуру односвязных и двусвязных графов. Можно этот процесс продолжать, пытаясь описывать 3-связные, 4-связные и т.д. графы. При этом с ростом k структура k-связных графов становится все сложнее. Оказывается, однако, что любой k-связный граф при $k \geqslant 3$ обладает целым рядом базовых свойств, характерных и для 2-связных графов. И основной результат такого рода — это альтернативное описание k-связного графа на языке путей, связывающих любые две вершины в таком графе.

16.1.1. Вернемся ненадолго к двусвязным графам. Теорема 15.3 утверждает, что в двусвязном графе G через любые две вершины x, y графа проходит хотя бы один простой цикл. Но любой такой цикл представляет собой, по сути, два простых пути, соединяющих x и y и не пересекающихся между собой ни в одной из своих внутренних вершин. Иными словами, мы можем переформулировать теорему 15.3 следующим образом.

Следствие 16.1 (Whitney, 1932)**.** *Граф G, построенный на трех или более вершинах, является двусвязным тогда и только тогда, когда для любой пары x, y его вершин найдутся два простых пути, соединяющих x и y и не имеющих общих внутренних вершин.*

Отмеченное свойство можно рассматривать и как альтернативное определение двусвязного графа, а именно, называть граф G двусвязным в случае, если для любой пары его вершин найдется по крайней мере два простых пути, соединяющих эти вершины и не имеющих общих внутренних вершин.

16.1.2. Оказывается, что аналогичный результа имеет место и для произвольного k-связного графа.

Теорема 16.2 (Уитни, 1932)**.** *Простой граф G является k-связным тогда и только тогда, когда между любыми двумя его вершинами x и y существует k путей, не имеющих общих внутренних вершин.*

Доказательство этой теоремы базируется, в свою очередь, на одном из наиболее фундаментальных результатов теории графов — теореме, доказанной австрийским математиком Карлом Менгером в 1927 году.

Определение 16.3. Пусть $\{x, y\}$ есть пара различных вершин графа G. Подмножество R множества $V(G) \setminus \{x, y\}$ называется вершинно разделяющим x и y множеством, если в графе $G - R$ пути между x и y отсутствуют.

Теорема 16.4 (Менгер, 1927)**.** *Пусть $x, y \in V(G)$ — две несмежные вершины графа G. Тогда количество $\kappa(x, y)$ вершин в наименьшем вершинном разделяющем x и y множестве совпадает с наибольшим количеством простых путей из x в y, не имеющих общих внутренних вершин.*

Замечание 16.5. Теорема Менгера характеризует, как говорят, локальную связность графа — она говорит о том, сколько вершин нужно удалить в графе G с тем, чтобы разорвать в этом графе вершины x и y. С этой точки зрения теорема Уитни имеет глобальный характер — она описывает связность графа в целом на языке путей, соединяющих в этом графе две произвольные вершины.

16.1.3. Давайте вначале докажем теорему Уитни в предположении, что теорема Менгера верна.

Заметим, прежде всего, что в одну сторону утверждение теоремы Уитни очевидно — если между любыми двумя вершинами простого графа существует k не имеющих общих внутренних вершин путей, то в таком графе имеется как минимум $k+1$ вершина, и в нем не существует разделяющего множества, содержащего менее, чем k вершин. Поэтому такой граф k-связен.

Теперь предположим, что граф G является k-связным. Если вершины x и y несмежны, то существование k путей между ними сразу следует из теоремы Менгера. Поэтому далее будем рассматривать случай, когда в графе G существует ребро $e = \{x, y\}$. Рассмотрим тогда граф $G - e$. Покажем, что связность $\kappa(G - e)$ такого графа уменьшается не более чем на единицу по сравнению со связностью $\kappa(G)$ исходного графа.

Действительно, так как любое вершинно разделяющее множество S графа G является вершинно разделяющим множеством графа $G - e$, то $\kappa(G - e) \leqslant \kappa(G)$. Строгое неравенство $\kappa(G - e) < \kappa(G)$ имеет место в случае, когда в $G - e$ найдется вершинно разделяющее множество S, мощность $|S| = \kappa(G - e)$ которого строго меньше $\kappa(G)$. В этом случае S не является вершинно разделяющим множеством в G, граф $G - S$ является связным, граф $G - S - e$ имеет две компоненты связности $G[X]$ и $G[Y]$, такие, что $x \in X$, $y \in Y$, и в графе $G - S$ ребро e является мостом. Если теперь $|X| > 1$, то $S \cup \{x\}$ является вершинно разделяющим множеством в G, так что $|S| + 1 = \kappa(G - e) + 1 \geqslant \kappa(G)$. Аналогичная ситуация имеет место в случае, когда $|Y| > 1$. Наконец, в случае $|X| = |Y| = 1$ имеем $|S| = \kappa(G - e) = n - 2$, $\kappa(G) \geqslant n - 1$, а значит $G = K_n$. Но и для этого случая $\kappa(G - e) = n - 2 = \kappa(G) - 1$, что и требовалось доказать.

Итак, связность $\kappa(G-e)$ графа $G-e$ больше или равна $\kappa(G)-1$. Как следствие, любое вершинно разделяющее x и y множество R содержит как минимум $\kappa(G)-1$ вершину. Тогда, на основании теоремы Менгера, между x и y имеется по меньшей мере $\kappa(G) - 1$ простых путей, не имеющих общих внутренних вершин. Вместе с ребром e они образуют k путей, соединяющих x и y в исходном графе G и не имеющих общих внутренних вершин. Теорема Уитни доказана.

16.1.4. Вернемся к теореме Менгера. Изначальное ее доказательство было довольно сложным и громоздким. За прошедшие годы было предпринято довольно большое количество попыток упростить доказательство получено несколько довольно удачных доказательств этого результата. Мы здесь изложим, пожалуй, наиболее короткое доказательство теоремы Менгера, найденное немецким математиком Франком Гёрингом в 2000 году. Это доказательство базируется на некоторых вспомогательных понятиях и фактах, к изложению которых мы сейчас и перейдем.

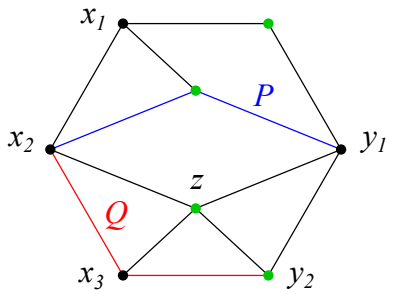


Рис. 76

Определение 16.6. Пусть $X, Y \subset V(G)$. Путем между X и Y называется любой простой путь P, начальная вершина которого принадлежит множеству X, конечная — множеству Y, а все внутренние вершины не принадлежат ни множеству X, ни множеству Y.

В качестве примера на рис.76 показаны подмножества $X = \{x_1, x_2, x_3\}$ и $Y = \{y_1, y_2\}$ вершин графа G, а также простой путь P, соединяющий X и Y (помечен синим цветом на рисунке). Помеченный красным цветом простой путь Q путем между X и Y не является — одна из внутренних вершин этого пути, а именно, вершина x_3, принадлежит X, а такая ситуация в определении исключается.

Замечание 16.7. Не исключен случай, когда $X \cap Y \neq \varnothing$. Поэтому данным определением допускаются и так называемые *тривиальные* пути, каждый из которых состоит из единственной вершины $x \in X \cap Y$.

Так, для графа, изображенного на рис.76, мы в качестве X и Y можем выбрать подмножества $X = \{x_1, x_2, x_3, z\}$ и $Y = \{z, y_1, y_2\}$. В этом случае путь $P = (z)$ является тривиальным путем, соединяющим два этих подмножества.

16.1.5. Следующее понятие, которое нам понадобится — это понятие вершинного отделяющего множества R.

Определение 16.8. Пусть X, Y и R есть некоторые подмножества множества $V(G)$ вершин. Говорят, что R *отделяет* множество X от множества Y, если любой путь из X в Y проходит через вершины множества R.

На рис.76 зеленым цветом помечены вершины подмножества R, отделяющего X от Y.

Замечание 16.9. Обратим внимание на отличие вершинного отделяющего X от Y множества от вершинного разделяющего граф G множества. При удалении вершинного разделяющего множества граф становится несвязным. От множества, которое отделяет X от Y, этого не требуется. Действительно, само множество X отделяет X от Y, а граф $G - X$ вполне может остаться связным.

16.1.6. Обозначим через k количество вершин в минимальном отделяющем X от Y множестве. Как мы уже отмечали ранее, X всегда отделяет себя от Y и наоборот. Поэтому $k \leqslant \min(|X|, |Y|)$. В случае, если между X и Y пути отсутствуют, $k = 0$. В остальных случаях $k > 0$.

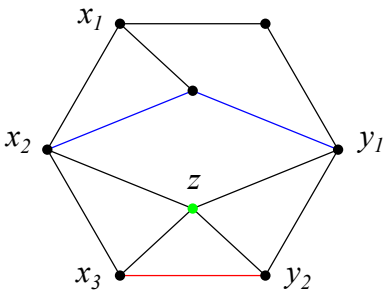


Рис. 77

В качестве примера рассмотрим граф G, показанный на рис.77. В этом графе в качестве отделяющего $X = \{x_1, x_2, x_3, z\}$ от $Y = \{z, y_1, y_2\}$ множества R можно выбрать подмножество Y. Возникает вопрос, является ли такое множество минимальным. Следующее утверждение позволяет получить ответ на этот вопрос.

Теорема 16.10. *Пусть X, Y — пара подмножеств множества $V(G)$. Тогда количество k вершин в минимальном отделяющем X от Y множестве R совпадает с максимальным количеством l попарно непересекающихся друг с другом путей из X в Y.*

Так, для множеств $X = \{x_1, x_2, x_3, z\}$ от $Y = \{z, y_1, y_2\}$, показанных на рис.77, имеются три попарно не пересекающиеся между собой пути, соединяющие вершины множества X с вершинами множества Y (пути, помеченные синим, красным и зеленым цветами на рисунке). Большее количество путей быть не может — это количество не может превосходить мощность множества Y. Следовательно, на основании теоремы 16.10, размер минимального отделяющего X от Y множества равен трем, так что Y является минимальным отделяющим X от Y множеством.

16.1.7. Приступим к доказательству теоремы 16.10. Понятно, что максимальное количество l попарно непересекающихся путей из X в Y не может превосходить количества k вершин в минимальном отделяющем X от Y множестве R — в противном случае, согласно принципу Дирихле, через какую-то вершину этого множества проходило бы два или более простых путей, чего быть не может. Теорема утверждает, что эти два числа на самом деле равны. Иными словами, нам нужно доказать, что если размер минимального вершинно отделяющего X от Y множества R равен k, то в графе G обязательно найдутся ровно k непересекающихся путей из X в Y.

Сразу заметим, что в двух тривиальных случаях $k = 0$ и $k = 1$ теорема верна. Действительно, случай $k = 0$ по определению означает, что пути между X и Y отсутствуют. В случае, когда минимальное отделяющее X от Y множество R состоит из одной вершины x, нам годится любой путь из X в Y, проходящий через x.

Доказательство теоремы в случае $k > 1$ проведем индукцией по количеству вершин и ребер в графе. В качестве базы индукции мы можем взять граф $\bar{K}_2$, состоящий из двух изолированных вершин, в котором $X = Y = \bar{K}_2$ (рис.78,a). Размер минимального отделяющего X от Y множества R равен в этом случае двум (в качестве такого множества необходимо взять множество $X = Y$). Максимальное же количество путей, соединяющих X и Y, также равно двум — в качестве таких путей выступают тривиальные пути, состоящие из одной вершины. Таким образом, для данного случая доказываемая нами теорема верна.

Возьмем теперь некоторый граф $G \neq \bar{K}_2$, в котором существуют множества вершин X и Y, такие, что размер минимального отделяющего X от Y множества R равен $k > 1$. Докажем наше утверждение для G, X и Y при условии, что для всех графов с меньшим количеством вершин и/или ребер теорема 16.10 верна.

Рассмотрим вначале случай, когда в графе G существует вершина $z \in X \cap Y$ (см.рис.77). В графе $G - z$ минимальное отделяющее $X \setminus \{z\}$ от $Y \setminus \{z\}$ множество $R \setminus \{z\}$ содержит $k - 1$ вершин. Тогда, по предположению индукции, в $G - z$ имеется $(k - 1)$ непересекающийся путь из $X \setminus \{z\}$ в $Y \setminus \{z\}$. Добавляя к этому набору тривиальный путь $\{z\}$, получаем набор из k непересекающихся путей в графе G.

Теперь разберем случай $X \cap Y = \varnothing$. Условие $k > 0$ означает, что X и Y соединены между собой хотя бы одним путем P. То, что $X \cap Y = \varnothing$, означает, что этот путь будет нетривиальным, то есть он будет содержать хотя бы одно ребро $e = \{x, y\}$. При этом, по определению пути между X и Y, вершина $x \notin Y$, а вершина $y \notin X$ даже в том случае, когда P состоит из единственного ребра e.

Предположим вначале, что при удалении ребра $e = \{x, y\}$ размер минимального отделяющего X от Y множества R' в графе $G - e$ не уменьшится по сравнению с размером разделяющего множества R в графе G (рис.78,b). Тогда, по индукционному предположению, в графе $G - e$ найдется k непересекающихся друг с другом путей из X в Y. Добавление же ребра e это количество путей уменьшить никак не сможет.

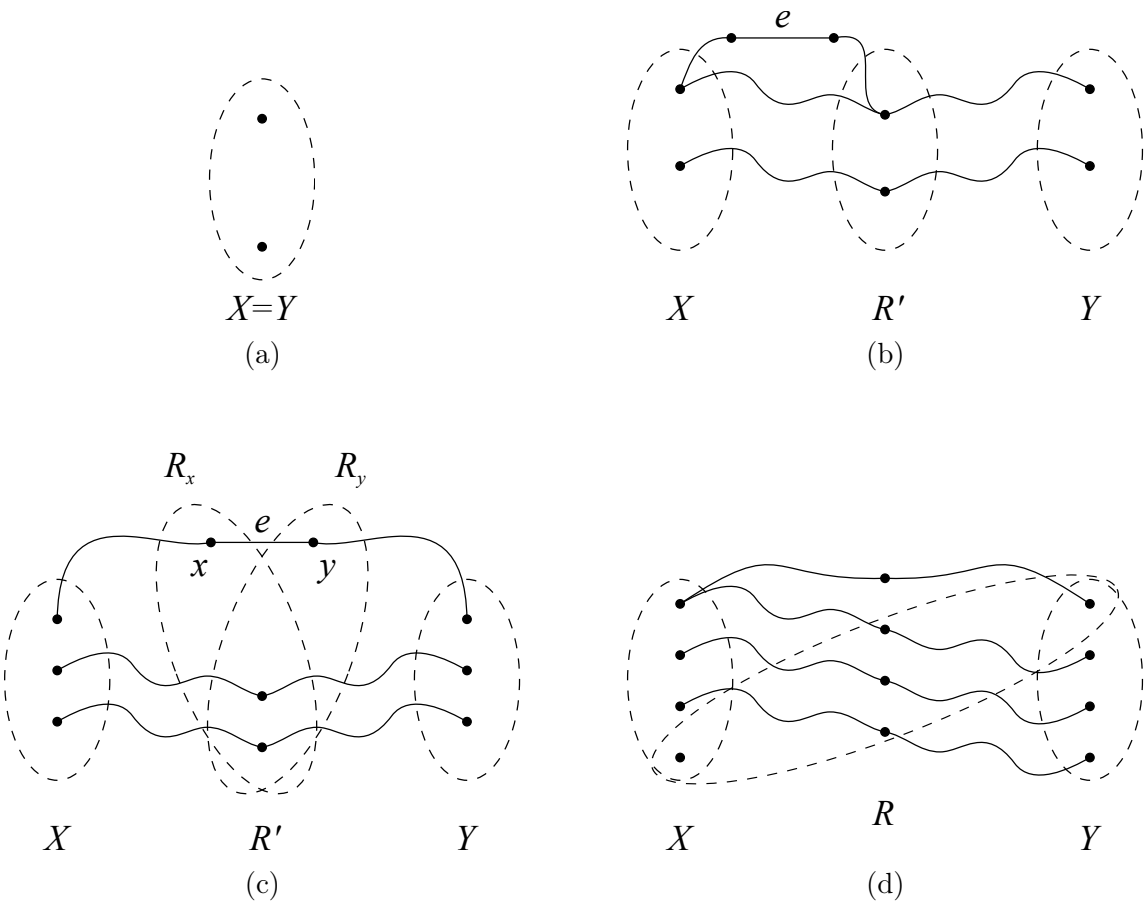


Рис. 78

Теперь предположим, что при удалении ребра $e = \{x, y\}$ размер отделяющего X от Y множества R' в графе $G - e$ уменьшился по сравнению с размером отделяющего X от Y множества R в графе G (рис.78,c) и стал равным $k - 1$. Это, в свою очередь, означает, что множества вершин $R_x = R' \cup \{x\}$ и $R_y = R' \cup \{y\}$ отделяют X от Y в графе G и содержат ровно k вершин. Кроме того, хотя бы одно из множеств R_x, R_y не совпадает ни с X, ни с Y. Действительно, в противном случае $R' = X \cap Y$, $|R'| = k - 1 \geqslant 1$, что противоречит условию $X \cap Y = \varnothing$. Иными словами, мы доказали, что в рассматриваемом случае в исходном графе G обязательно существует отделяющее X от Y множество R мощности k, не совпадающее ни с X, ни с Y (рис.78,d).

Обозначим через $\bar{X}$ и $\bar{Y}$ подмножества $X \setminus R$ и $Y \setminus R$ соответственно. Так как R отличен и от X, и от Y, то оба эти множества не пусты. Удалим в графе G все вершины множества $\bar{Y}$. Рассмотрим в получившемся графе G' множества X и R. Эти множества не совпадают, причем $|X| \geqslant k$, а $|R| = k$. Любое отделяющее X от R множество в графе G', если оно существует, содержит не менее, чем k вершин — в противном случае мы бы и в исходном графе G смогли отделить X от Y, удалив эти вершины. Но тогда, по индукционному предположению, в графе G' существует k непересекающихся путей из X в R. Аналогично доказывается, что в графе $G'' = G - \bar{X}$ существует k непересекающихся путей из R в Y. Так как $|R| = k$, то мы всегда можем состыковать эти пути с ранее построенными путями из X в R и получить искомый набор путей, соединяющих X и Y в исходном графе G. □

16.1.8. Покажем теперь, как, используя теорему 16.10, доказать теорему Менгера. Пусть x, y — две несмежные вершины графа G, удовлетворяющие условиям теоремы Менгера. Рассмотрим

множество X вершин, смежных с x, и множество Y вершин, смежных с y. Любое отделяющее X от Y множество разделяет вершины x и y и наоборот. Следовательно, размер минимального отделяющего X от Y множества совпадает с $\kappa(x, y)$. Но тогда, согласно теореме 16.10, существует k попарно непересекающихся путей из X в Y. Продолжая эти пути до точек x и y, мы тем самым доказываем справедливость теоремы Менгера.

17 Теорема Форда-Фалкерсона

17.1. Перейдем теперь к реберно k-связным графам. Как и следовало ожидать, и для таких графов имеет место аналоги как вершинной теоремы Менгера, так и вершинной теоремы Уитни.

Теорема 17.1 (Реберная теорема Менгера)**.** *Максимальное количество $\lambda'(x, y)$ реберно непересекающихся простых путей, соединяющих две различные вершины x и y связного графа G, совпадает с размером $\kappa'(x, y)$ минимального реберно разделяющего вершины x и y множества $R \subset E(G)$.*

Теорема 17.2 (Реберная теорема Уитни)**.** *Граф G является реберно k-связным тогда и только тогда, когда любые две вершины x, y этого графа связаны между собой по меньшей мере k попарно реберно не пересекающимися между собой путями.*

Интересно заметить, что эти утверждения были доказаны лишь в 1956 году как следствия более общего результата — теоремы Форда-Фалкерсона о максимальном потоке в сети. В силу чрезвычайной важности теоремы Форда-Фалкерсона нам также будет полезно вначале сформулировать и доказать этот более общий результат, а затем из него как частный случай получить реберную теорему Менгера (см. упражнение **??**) и реберную теорему Уитни.

17.1.1. Для того, чтобы сформулировать теорему Форда-Фалкерсона, нам необходимо ввести несколько дополнительных понятий. Начнем с понятия сети.

Сеть можно представлять себе как математическую модель некоторой транспортной системы, доставляющей какой-то продукт (людей, нефть, воду, электричество) из одного места в другое. Если каналы такой системы представить ориентированными ребрами, а промежуточные станции — вершинами, то мы получим некоторый простой слабо связный орграф в качестве математической модели такой транспортной системы (рис.79). Однако такого простого математического объекта нам для полного описания транспортной сети явно не достаточно.

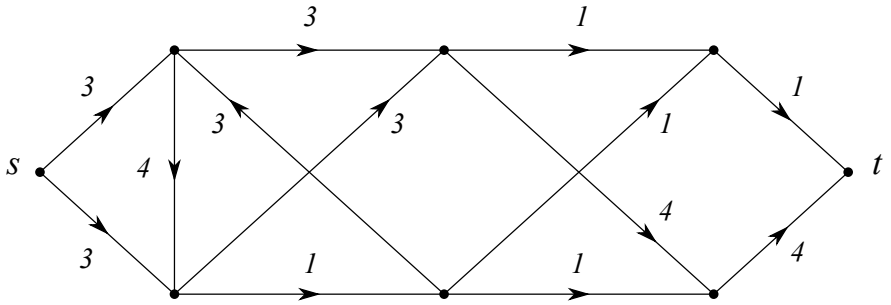


Рис. 79: Сеть с источником s и стоком t

Во-первых, нам для любого канала рассматриваемой транспортной системы нужно указать пропускную способность этого канала как максимальное количество транспортируемого продукта,

которое можно пропустить через данный канал в единицу времени (см. числа над ребрами на рис.79). Как следствие, любому ориентированному ребру (x, y) мы должны приписать некоторое неотрицательное вещественное число $c(x, y)$, называемое пропускной способностью этого ребра $e = (x, y)$.

Во-вторых, в простейшем случае можно считать, что система сконструирована для передачи продукта из единственного места добычи в единственное же место потребления данного продукта. На языке теории графов это означает, что в орграфе D имеются две выделенные вершины — источник s и сток t (рис.79). Степень входа вершины s равна нулю, то есть в источник ничего не втекает. Напротив, у вершины t равна нулю степень выхода — из этой вершины ничего вытекать не должно.

Теперь мы можем дать формальное определение сети.

Определение 17.3. Пусть среди вершин простого слабо связного орграфа D выделены две вершины s и t, называемые источником и стоком, такие, что $\operatorname{indeg}(s) = \operatorname{outdeg}(t) = 0$. Пусть, кроме того, на множестве $E \subset V \times V$ ребер орграфа D определена вещественная неотрицательная функция $c(x, y)$. Тогда орграф $D = (V, E, s, t, c)$ называется *сетью,* а функция $c(x, y)$ — *пропускной способностью* сети D.

17.1.2. Предположим, что мы через нашу транспортную систему пропускаем какое-то количество Q продукта в единицу времени из места добычи этого продукта в место его потребления. Это означает, что какая-то часть этого продукта транспортируется через каждый отдельно взятый канал нашей системы. Нам при этом важно, во-первых, следить за тем, чтобы количество продукта, проходящего через каждый отдельно взятый канал, не превосходило максимальной пропускной способности этого канала. Во-вторых, нам нужно следить за тем, чтобы количество прибывающего на каждую промежуточную станцию продукта равнялось количеству исходящего из данной станции продукта, то есть чтобы на каждой станции не нарушался закон сохранения вещества.

Для того, чтобы описать все вышесказанное на языке теории графов, нам нужно ввести понятие потока в сети.

Определение 17.4. *Потоком в сети* D из вершины s в вершину t называется неотрицательная вещественная функция, определенная на множестве $E(D)$ ребер орграфа D и удовлетворяющая следующим условиям:

1. для любых $(x, y) \in E(D)$ функция $f(x, y) \geqslant 0$ и $f(x, y) \leqslant c(x, y)$;
2. для любой вершины $x \in V(D)$, отличной от s и t, справедливо равенство

$$f^+(x) := \sum_{y\colon (x,y)\in E(D)} f(x, y) = \sum_{z\colon (z,x)\in E(D)} f(z, x) =: f^-(x),$$

описывающее закон сохранения вещества в сети: количество $f^+(x)$ вещества, вытекающее из вершины x, равняется количеству $f^-(x)$ вещества, втекающего в x.

Число

$$\operatorname{val}(f) := \sum_{x\colon (s,x)\in E(D)} f(s, x) = f^+(s),$$

характеризующее количество исходящего из источника s в единицу времени продукта, называется при этом *величиной* данного потока. В силу закона сохранения потока, эта же величина характеризует количество входящего в сток t продукта ($f^+(s) = f^-(t)$).

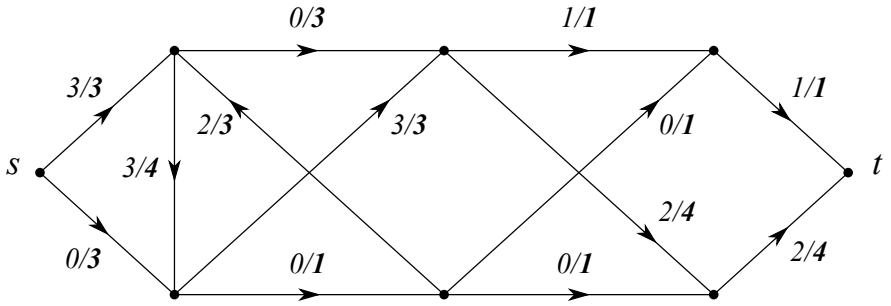


Рис. 80: Поток в сети

В показанной в качестве примера на рис.80 сети нормальным шрифтом помечен поток через каждое ребро, а жирным шрифтом — величина пропускной способности этого ребра.

17.1.3. Основной задачей, связанной с потоками в сетях, является задача максимизации потока $\mathrm{val}(f)$ в заданной сети D. Можно показать, что эта задача представляет собой задачу линейного программирования. Как и всякая задача линейного программирования, такая задача имеет и двойственную к ней задачу линейного программирования, а именно, задачу о нахождении так называемого *минимального разреза* в сети.

Определение 17.5. Пусть (S,T) есть разбиение множества V вершин орграфа D на два блока S и T, таких, что вершина $s \in S$, а вершина $t \in T$. Тогда подмножество ребер

$$R(S,T) := \{e = (x,y) \in E(D) \mid x \in S,\ y \in T\}$$

называется (S,T)-*разрезом* сети D, а величина

$$\mathrm{cap}(S,T) := \sum_{(x,y)\in R(S,T)} c(x,y)$$

называется *пропускной способностью* данного разреза.

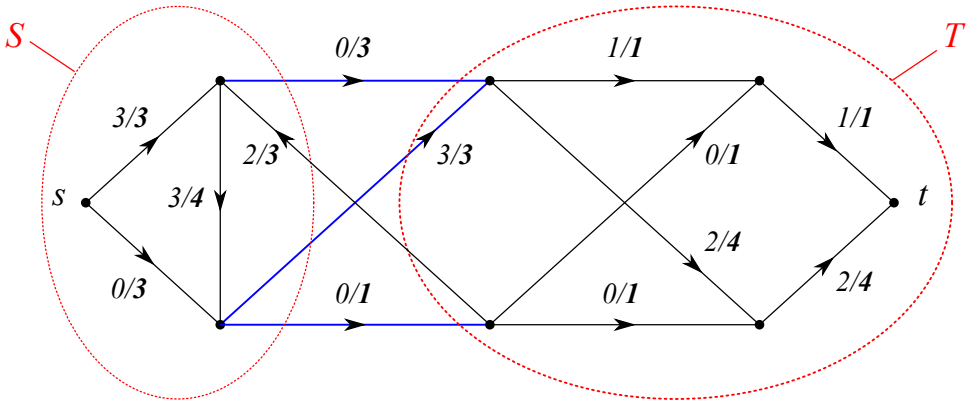


Рис. 81: (S,T)-разрез в сети

Заметим, что (S,T)-разрезу принадлежат не все ребра между вершинами из подмножеств S и T, а лишь те, которые выходят из S и входят в T. Ребра, исходящие из T и входящие в S, такому разрезу не принадлежат.

В качестве примера рассмотрим (S,T)-разрез, показанный на рис.81 (синие ребра на рисунке), пропускная способность которого равна 7. Исходящее из T и входящее в S ребро с пропускной способностью 3 этому разрезу не принадлежит.

Нас будут интересовать разрезы, пропускная способность которых минимальна. Любой такой разрез мы и будем называть *минимальным разрезом.*

17.1.4. Зафиксируем некоторый поток в сети D. Более-менее очевидно, что величина Q этого потока не может превосходить пропускной способности *любого* разреза сети D. Именно, справедлива

Лемма 17.6 (Слабая двойственность)**.** *Для любого потока f в сети D и произвольного (S,T)-разреза в этой сети имеет место неравенство*

$$\operatorname{val}(f) \leqslant \operatorname{cap}(S,T). \tag{47}$$

Доказательство. Зафиксируем какое-то подмножество U множества $V(D)$ сети D. Обозначим через $f^+(U)$ суммарный поток на ребрах, исходящих из U, а через $f^-(U)$ — суммарный поток на ребрах, входящих в U. С физической точки зрения $f^+(U)$ представляет собой количество вещества, вытекающего из U, а $f^-(U)$ — количество вещества, втекающего в U. Покажем, что разность между количеством вытекающего из U вещества и количеством втекающего в него вещества, характеризующая, по сути, поток вещества через границу U, равна

$$f^+(U) - f^-(U) = \sum_{u\in U} \left[f^+(u) - f^-(u)\right].$$

Посмотрим для этого, какой вклад дает поток $f(e)$ через некоторое ребро e в каждую из частей этого равенства. Если e соединяет две внутренние вершины подмножества U, то в левую часть $f(e)$ вклада не дает, а в правой части этот вклад сокращается. В случае, если e соединяет две внешние по отношению к U вершины, то вклад от $f(e)$ отсутствует в обоих частях рассматриваемого равенства. Если e представляет собой ребро, исходящее из U, то $f(e)$ дает одинаковый положительный вклад в каждую из частей нашего равенства, а если e есть ребро, входящее в U, то $f(e)$ дает одинаковый отрицательный вклад в обе части равенства.

Теперь рассмотрим некоторый разрез (S,T) в сети. Для потока, вытекающего из S, мы на основании доказанного равенства и законов сохранения ($f^+(x) = f^-(x)$ для любого $x \neq s,t$) можем записать, что

$$f^+(S) - f^-(S) = f^+(s) = \operatorname{val}(f).$$

Но тогда мы и получаем, что

$$\operatorname{val}(f) = f^+(S) - f^-(S) \leqslant f^+(S) \leqslant \sum_{(x,y)\in E(D)\colon x\in S,\, y\in T} c(x,y) = \operatorname{cap}(S,T).$$

Следствие 17.7. *Пусть f есть поток в сети D, а (S,T) — разрез в этой сети, такие, что* $\operatorname{val}(f) = \operatorname{cap}(S,T)$*. Тогда f представляет собой максимальный поток в сети, а (S,T) — минимальный разрез в этой сети.*

17.1.5. В теории экстремальных задач слабая двойственность (47) используется обычно для того, чтобы получить некоторую верхнюю границу на значения целевой функции (в данном случае, на величину $\operatorname{val}(f)$). Кроме того, как мы уже заметили выше, слабая двойственность может быть использована при доказательстве оптимальности того или иного решения — если в задаче можно найти величину потока в сети, равную пропускной способности некоторого разреза, то в таком случае найденная величина $\operatorname{val}(f)$ потока гарантированно является оптимальной.

В общем случае вполне возможна ситуация, при которой максимальное значение целевой функции в прямой задаче оказывается строго меньше значения целевой функции в обратной задаче.

Однако для рассматриваемого класса задач такая ситуация невозможна — именно, мы сейчас покажем, что если величина $\mathrm{val}(f)$ потока достигает своего максимального значения, то это значение обязательно совпадает с минимальной пропускной способностью некоторого разреза. В теории экстремальных задач подобного рода результат называется обычно строгой двойственностью.

Теорема 17.8 (Сильная двойственность; L. R. Ford, D. R. Fulkerson, 1956)**.** *Во всякой сети величина любого максимального потока равна пропускной способности любого минимального разреза.*

Доказательство. Предположим, что в сети D существует максимальный поток f. Мы покажем, как в этом случае построить в этой сети разрез (S, T), пропускная способность которого совпадает с величиной максимального потока. На основании следствия 17.7 это, в свою очередь, означает, что такой разрез является минимальным.

Для этого рассмотрим вместо D неориентированный граф G, полученный из исходного орграфа заменой всех ориентированных ребер на неориентированные. Разобъем множество $V(G)$ вершин этого графа на два блока S и T следующим образом. К блоку S отнесем вершину s, а также любую вершину $x \in V(G)$, для которой существует простой путь $(s = x_0, x_1, \ldots, x_k = x)$ из s в x, такой, что любому ребру $\{x_i, x_{i+1}\} \in E(G)$ данного пути соответствует

(1) либо ориентированное ребро (x_i, x_{i+1}) графа D, значение $f(x_i, x_{i+1})$ на котором строго меньше $c(x_i, x_{i+1})$ (в этом случае говорят, что ребро $\{x_i, x_{i+1}\} \in E(G)$ соответствует *ненасыщенному* ребру $(x_i, x_{i+1}) \in E(D)$),
(2) либо ориентированное ребро (x_{i+1}, x_i) графа D, направленное навстречу построенному пути, значение $f(x_{i+1}, x_i)$ на котором строго больше нуля.

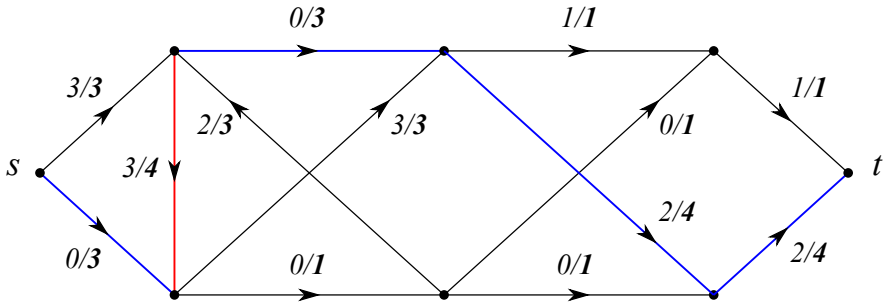


Рис. 82: Поток и дополняющий путь

Покажем теперь, что множество $T = V \setminus S$ не пусто и, в частности, обязательно содержит вершину t. Действительно, предположим, что это не так. Тогда в графе G существует простой путь из s в t, все ребра которого удовлетворяют описанным выше свойствам (1)-(2) — так называемый дополняющий (или увеличивающий, или ненасыщенный) путь (см.рис.82, на котором показан дополняющий путь в сети из вершины s в вершину t; синие ребра отвечают первому типу ребра, при котором $f(e_i) < c(e_i)$, а красные — второму типу ребра, при котором оно направлено навстречу пути и при этом $f(e_j) > 0$). Эти свойства означают, что существует положительное число ε, которое, во-первых, не превосходит ни одного числа $c(x_i, x_{i+1}) - f(x_i, x_{i+1})$, необходимого для насыщения любого ребра первого типа, а во-вторых, не превышает ни одной величины потока $f(x_{i+1}, x_i)$ через любое из ребер второго типа ($\varepsilon = 2$ для увеличивающего пути на рис.82). Добавим эту величину ε к потоку $f(x_i, x_{i+1})$ через каждое ребро первого типа и вычитем ее из потока $f(x_{i+1}, x_i)$ через каждое ребро второго типа. В результате мы получим

новый поток, величина которого превосходит величину исходного потока на значение $\varepsilon > 0$. Однако это невозможно — по предположению, величина исходного потока была максимальной. Следовательно, вершина t обязательно принадлежит множеству T, то есть это множество T не пусто.

Для завершения доказательства остается понять, что построенное разбиение множества вершин на два блока S и T является (S,T)-разрезом, пропускная способность которого совпадает с величиной максимального потока. Но это действительно так: по построению множества S, любое ребро, идущее из S в T, является насыщенным, а поток на любом ребре, идущем из T в S, равен нулю. Следовательно, поток через (S,T) в точности равен пропускной способности разреза (S,T). □

Замечание 17.9. Доказанная теорема ничего не говорит о существовании максимального потока в сети.

17.1.6. Идеи, положенные в основу доказательства теоремы Форда-Фалкерсона, можно использовать для построения простейшего алгоритма нахождения максимального потока в сети [**?**, **?**] — так называемого алгоритма Форда-Фалкерсона. Именно, положим на первом шаге алгоритма $f(x,y) = 0$ для всех $(x,y) \in E(D)$. Рассмотрим далее произвольный простой путь из s в t и увеличим поток в сети на описанную в доказательстве теоремы величину ε. На следующем шаге найдем произвольный дополняющий путь из s в t в нашей сети и увеличим поток на величину ε, сосчитанную для данного пути. Будем продолжать процесс до тех пор, пока в сети не останется дополняющих путей. В случае, если этот процесс завершится, результирующий поток будет максимальным.

Основным недостатком этого алгоритма является тот факт, что время его работы существенно зависит от выбора дополняющего пути на каждом шаге алгоритма. При неудачном выборе этих путей данный алгоритм может вообще никогда не завершиться (смотри упражнение **??**). Можно показать, однако, что в случае целочисленных или рациональных значений пропускных способностей $c(x,y)$ мы всегда получим максимальный поток за конечное число шагов алгоритма (упражнение **??**).

Конструктивным доказательством существования максимального потока в сети в случае вещественных значений пропускных способностей является модификация метода Форда-Фалкерсона, известная как алгоритм Эдмонса-Карпа. В этой модификации на каждом шаге с помощью алгоритма поиска в ширину находится *кратчайший* по количеству ребер дополняющий путь. Затем поток в сети, как и в основном алгоритме, увеличивается на величину ε. Можно доказать (см., например, [**?**]), что время выполнения данной модификации алгоритма составляет $O(|V| \cdot |E|^2)$.

Паросочетания в графах

18 Понятие паросочетания. Теорема Бержа. Независимые множества и покрытия графа

18.1. Понятие паросочетания является одним из наиболее важных понятий теории графов. Оно встречается в огромном количестве прикладных задач [**?**].

18.1.1. Начнем, как всегда, с определения данного понятия.

Определение 18.1. Паросочетанием M в произвольном графе G называется любой набор ребер, не имеющих общих концевых вершин.

Замечание 18.2. Из определения видно, что при изучении паросочетаний нас будут интересовать только лишь простые графы. Действительно, любая петля в мультиграфе соединяет вершину саму с собой, и поэтому по определению не может входить ни в какое из паросочетаний. Далее, если одно из ребер мультиграфа, соединяющего пару вершин графа, входит в паросочетание M, то остальные ребра, соединяющие ту же пару вершин, в паросочетание M по определению войти не могут. Поэтому и мультиребра в графе также рассматривать смысла не имеет.

Пример 18.3. Любое одиночное ребро e в (простом) графе $G \neq \bar{K}_n$ является простейшим примером паросочетания.

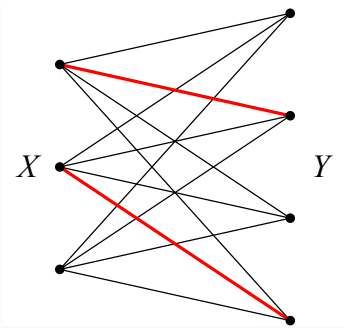


Рис. 83

Пример 18.4. На рис.83 показан чуть менее тривиальный пример паросочетания M в полном двудольном графе $K_{3,4}$ (ребра, помеченные красным цветом на рисунке). Это паросочетание состоит из двух ребер и соединяет (или, как говорят, покрывает) четыре вершины этого графа.

Определение 18.5. Говорят, что вершина $x \in V(G)$ покрыта паросочетанием M, если она является концом одного из ребер e, входящего в паросочетание M.

18.1.2. На практике нам, как правило, хочется покрыть паросочетанием как можно большее количество вершин в графе G. Например, мы можем захотеть для выполнения какого-то задания разбить всех студентов в группе на пары так, чтобы студенты внутри каждой пары имели опыт работы друг с другом. В этом примере все множество студентов можно рассматривать

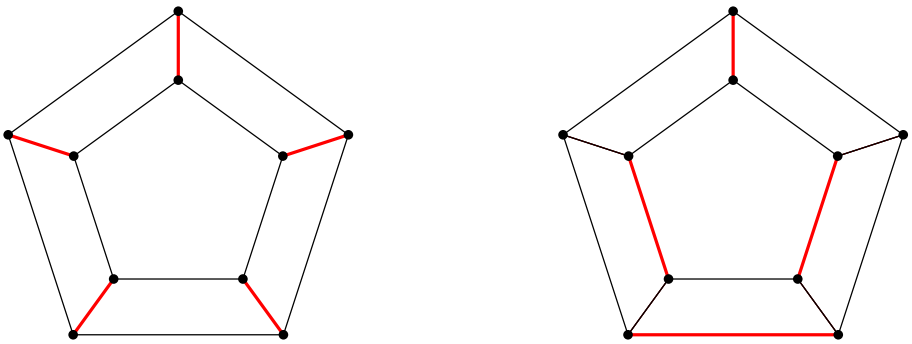
как множество $V(G)$ вершин графа G. Ребро между двумя вершинами проводится в том случае, если соответствующие этим вершинам студенты имели до этого момента опыт совместной работы. Понятно, что если граф $G \neq \bar{K}_n$, то хотя бы одну такую пару студентов мы как-то сможем найти. Однако нам хотелось бы разбить на пары как можно большее количество студентов, в идеале — всех. На языке теории графов такое идеальное для нас разбиение носит название совершенного паросочетания.

Определение 18.6. Паросочетание M называется *совершенным,* если оно покрывает все вершины графа G.

В качестве примера рассмотрим граф G, показанный на рис.84,a. Красным цветом на этом рисунке помечены ребра, образующие паросочетание в представленном графе. Так как эти ребра покрывают все десять вершин графа G, то такое паросочетание является совершенным.

18.1.3. Далеко не все графы имеют совершенное паросочетание. Так, очевидно, что любое паросочетание M покрывает четное количество вершин в графе G. Поэтому любой граф, построенный на нечетном количестве вершин, совершенного паросочетания не имеет. Для графов, у которых совершенные паросочетания отсутствуют, полезно ввести понятие максимального паросочетания.

Определение 18.7. *Максимальным паросочетанием* (maximum matching) в графе G называется паросочетание M, размер которого (то есть количество входящих в него ребер) является наибольшим среди всех паросочетаний в графе G. Количество ребер $|M|$ в таком паросочетании M обозначается обычно через $\alpha'(G)$.



(a) совершенное (b) наибольшее по включению

Рис. 84: Паросочетания

18.1.4. Казалось бы, для построения максимального паросочетания в графе можно использовать жадный алгоритм, добавляя случайным образом к M ребра графа до тех пор, пока это возможно. Полученное паросочетание, однако, максимальным может и не оказаться.

В качестве примера на рис.84,b показано паросочетание M, полученное в результате работы некоторого жадного алгоритма. Оно является *наибольшим по включению* (maximal matching) в том смысле, что добавление к нему любого другого ребра из множества $E(G)$ уже невозможно. Однако количество ребер в M строго меньше $\alpha'(G) = 5$. Максимальное (и одновременно совершенное) паросочетание для такого графа показано на рис.84,a.

18.2. Итак, мы убедились на приведенном выше примере, что жадный алгоритм дает нам не максимальное, а наибольшее по включению паросочетание. Такое паросочетание иногда может

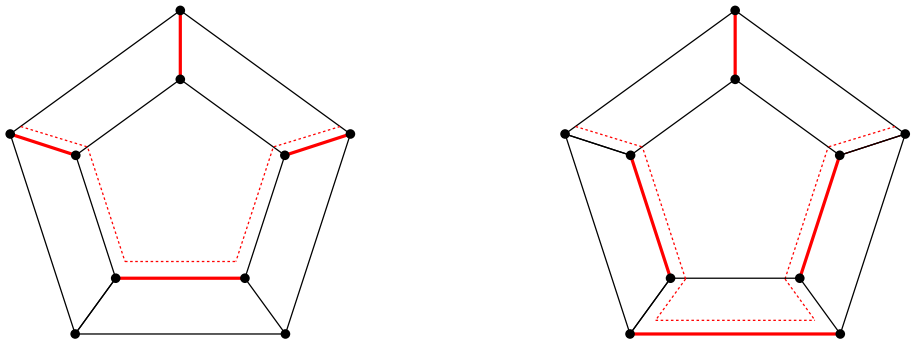
оказаться максимальным, а иногда может таковым и не быть. В принципе, эти соображения могли бы нам позволить построить алгоритм поиска максимального паросочетания в графе, если бы в дополнении к жадному алгоритму у нас был бы некоторый критерий проверки того или иного графа на максимальность.

18.2.1. Действительно, если бы такой критерий существовал, то мы могли бы для произвольного графа G запускать жадный алгоритм, получать с его помощью наибольшее по включению паросочетание, а затем с помощью этого критерия проверять полученное паросочетание на максимальность. В случае, если паросочетание оказывается максимальным, мы останавливаемся. В противном случае мы вновь запускаем жадный алгоритм.

18.2.2. Итак, для реализации описанного выше подхода нам необходим критерий максимальности паросочетания. Для того, чтобы этот критерий сформулировать, нам понадобятся следующие полезные понятия.

Определение 18.8. Пусть M есть некоторое паросочетание в графе G. Произвольный путь P в графе G, в котором котором чередуются ребра, входящие в M, и ребра, в M не входящие, называется *M-чередующимся* (рис.85,a).

Определение 18.9. M-чередующийся путь, обе концевые вершины которого не покрыты паросочетанием M, называется *M-дополняющим* путем (рис.85,b).



(a) M-чередующийся путь (b) M-дополняющий путь

Рис. 85

Посмотрим теперь повнимательнее на рис.85,b. Заметим, что в M-дополняющем пути P (помеченном пунктирной ломаной на рис.85,b) количество ребер, покрытых паросочетанием M, на единицу меньше количества ребер в P, этим паросочетанием не покрытых. Исключим теперь из паросочетания M ребра, входящие в P, и добавим к M ребра этого пути, изначально паросочетанием не покрытые (процесс, известный в англоязычной литературе как matching augmentation). В результате мы получим некоторое новое паросочетание M', содержащее на единицу большее количество ребер по сравнению с исходным паросочетанием M. Одновременно с этим у нас увеличится количество вершин, покрытых паросочетанием M'. Иными словами, мы показали, что наличие в графе G для заданного паросочетания M-дополняющего пути является признаком того, что паросочетание M максимальным не является.

18.2.3. В 1957 году французский математик Claude Berge доказал, что условие отсутствия для заданного паросочетания M в графе M-дополняющего пути является не только необходимым, но и достаточным условием максимальности паросочетания M. Именно, справедлива

Теорема 18.10 (Berge, 1957)**.** *Паросочетание M в графе G является максимальным тогда и только тогда, когда в таком графе M-дополняющие пути отсутствуют.*

Доказательство. Необходимость этого условия мы уже проверили — мы доказали, что если в графе существует M-дополняющий путь, то паросочетание M максимальным не является. Нам осталось доказать достаточность этого условия. Именно, предположим, что M не является максимальным паросочетанием в графе G. Покажем, что тогда в графе G обязательно найдется M-дополняющий путь.

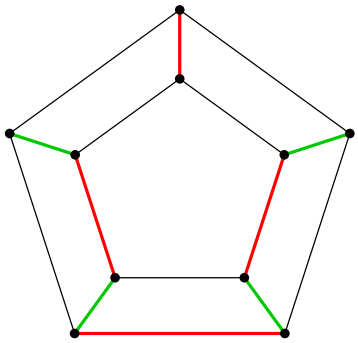


Рис. 86

Обозначим через M' максимальное паросочетание в G (зеленые ребра на рис.86). Так как паросочетание M максимальным не является, то $|M'| > |M|$. Рассмотрим симметрическую разность $F := M \Delta M'$ этих двух паросочетаний. Так как любая вершина $x \in V(G)$ может быть инцидентна максимум одному ребру из M и максимум одному ребру из M', то степень любой такой вершины в подграфе H, индуцированном подмножеством ребер F, не превосходит двух. Следовательно, любая компонента связности подграфа H представляет собой путь или цикл, в котором чередуются ребра из M и из M'.

В случае цикла чередование ребер из M и M', в частности, означает, что любой цикл в H имеет четную длину, а значит, количество ребер из M и M' в любом из циклов одинаково. Следовательно, условие $|M'| > |M|$ означает, что в графе H обязан существовать путь, начальное и конечное ребра которого принадлежат M'. Начальная и конечная вершины такого пути покрыты M', а значит, не покрыты M. Поэтому данный путь представляет собой M-дополняющий путь в G. □

18.3. Наряду с паросочетанием, в теории графов имеется еще целый ряд тесно связанных с ним понятий, к изучению которых мы и перейдем во второй части данного параграфа.

18.3.1. Начнем с определения *вершинного покрытия* графа.

Определение 18.11. Вершинным покрытием графа G называется набор вершин, покрывающих все ребра данного графа, то есть, более строго, подмножество K множества $V(G)$ вершин графа, такое, что любое ребро $e \in E(G)$ инцидентно по крайней мере одной вершине этого подмножества.

Тривиальным примером вершинного покрытия является все множество $V(G)$ вершин графа G. На рис.87,a приведен чуть менее тривиальный пример вершинного покрытия — вершины, помеченные красным цветом на рисунке, покрывают все ребра нашего графа. Нас же, как правило, будут интересовать вершинные покрытия наименьшего размера — для графа G, показанного на рисунке, таковым является пара вершин, помеченных зеленым цветом на рис.87,b.

Определение 18.12. Покрытие K графа G называется *минимальным,* если любое другое покрытие K' имеет размер $|K'|$, больший или равный $|K|$. Количество вершин в минимальном вершинном покрытии обозначается через $\beta(G)$.

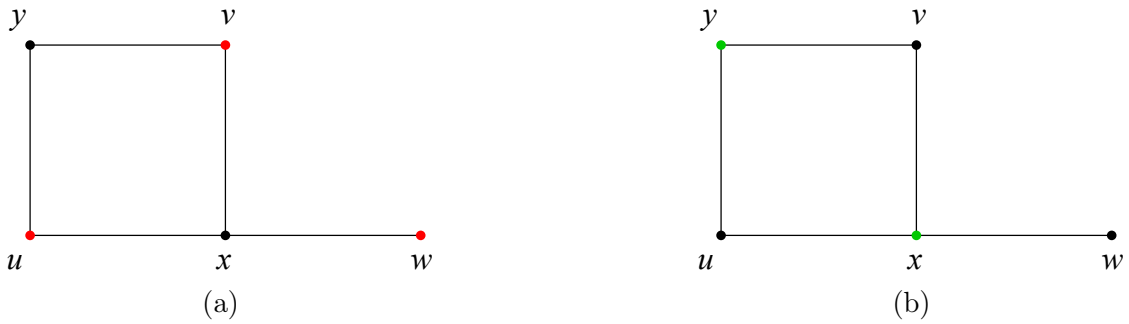


Рис. 87

Замечание 18.13. Задача поиска минимального вершинного покрытия имеет довольно наглядную практическую интерпретацию. Предположим, что у нас имеется музей, упрощенный план которого описывается графом G, показанным на рис.87. При этом ребра графа G на этом плане описывают галереи музея, а вершины — проходы между галереями. Дирекция музея хочет так расставить смотрителей в музее, чтобы ими просматривались все галереи музея. Самое простое решение — поставить по смотрителю в каждом проходе. Однако оно, конечно же, оптимальным не является. Решив для этого графа задачу о поиске минимального вершинного покрытия, мы определим минимально необходимое количество $\beta(G)$ смотрителей музея, а также места, в которых эти смотрители должны находиться.

18.3.2. Вернемся к задаче поиска максимального паросочетания в графе G. Оказывается, что с точки зрения теории экстремальных задач задача поиска минимального вершинного покрытия графа G оказывается двойственной к задаче поиска максимального паросочетания в G. Именно, имеет место

Лемма 18.14 (Слабая двойственность)**.** *В любом графе G размер $|M|$ произвольного паросочетания в G не превосходит количества $|K|$ вершин в произвольном вершинном покрытии графа G. В частности, для любого графа G справедливо неравенство*

$$\alpha'(G) \leqslant \beta(G). \tag{48}$$

Доказательство. Действительно, никакая вершина в графе G не может быть инцидентна двум или более ребрам одного и того же паросочетания M. Следовательно, даже для того, чтобы покрыть все ребра паросочетания M (не говоря уже о всех ребрах графа), нам понадобится по меньшей мере $|M|$ вершин. □

18.3.3. Как обычно, слабая двойственность оказывается крайне полезной для доказательства оптимальности решений двойственных экстремальных задач — если мы в графе G нашли какое-то паросочетание M, размер $|M|$ которого совпал с количеством $|K|$ вершин в некотором вершинном покрытии графа, то отсюда и из леммы сразу следует, что M является максимальным паросочетанием в G, а K — минимальным вершинным покрытием G.

В качестве простейшего примера на рис.88,а показан цикл C_4, в котором имеется паросочетание M размером два (красные ребра), а также вершинное покрытие K размером два (зеленые вершины на рисунке). Следовательно, для такого графа соответствующее паросочетание M является максимальным, а вершинное покрытие K — минимальным, причем $\beta(C_4) = |K| = 2 = |M| = \alpha'(C_4)$.

В общем случае произвольного графа G равенство $\beta(G) = \alpha'(G)$ выполняться, конечно же, не обязано. В качестве характерного примера на рис.88 показан граф C_5, для которого $\beta(C_5) = 3$,

(a) (b)

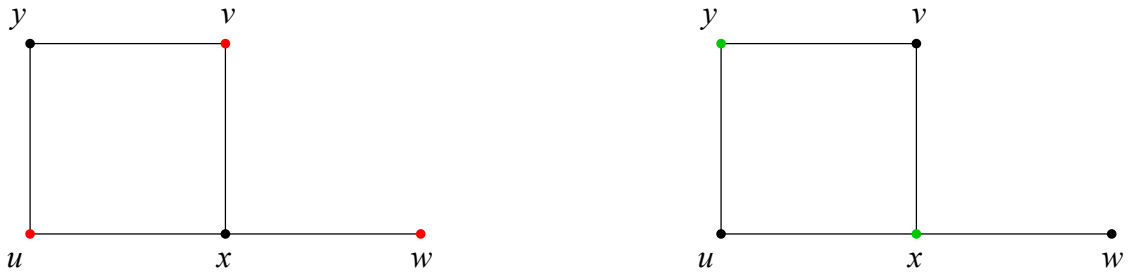
Рис. 88

а $\alpha(C_5) = 2$. В следующем параграфе мы покажем, что в частном случае двудольных графов равенство $\beta(G) = \alpha'(G)$ выполнено всегда, то есть в этом случае имеет место так называемая сильная двойственность.

18.3.4. Напомним, что под паросочетанием мы понимаем любой набор ребер в графе, не имеющих общих концевых вершин. Иногда паросочетание в графе еще называется *реберно независимым множеством.* Последнее название обычно используется как реберная аналогия еще одного важного понятия — *вершинно независимого множества.*

Определение 18.15. *Вершинно независимым* (или просто *независимым*) *множеством* в графе G называется любой набор S попарно несмежных между собой вершин.

Довольно очевидно, что, выбирая произвольную вершину x в нетривиальном графе G, мы получаем некоторое независимое множество $S = \{x\}$. Нам, однако, часто хочется понять, каков может быть максимальный размер независимого множества S.



(a) (b)

Рис. 89

Рассмотрим, к примеру, граф G, изображенный на рис.89. Любая вершина такого графа является независимым множеством в G. Независимое же множество S максимального размера состоит из попарно несмежных между собой вершин u, v и w (вершины, помеченные красным цветом на рис.89,а).

Определение 18.16. Вершинно независимое множество S называется *максимальным,* если количество $|S'|$ вершин в любом другом вершинно независимом множестве S' меньше или равно $|S|$. Количество вершин в максимальном вершинно независимом множестве графа G иногда называется *числом независимости* (independence number) и обозначается через $\alpha(G)$.

Таким образом, для графа G, показанного на рис.89,а, число независимости $\alpha(G) = 3$.

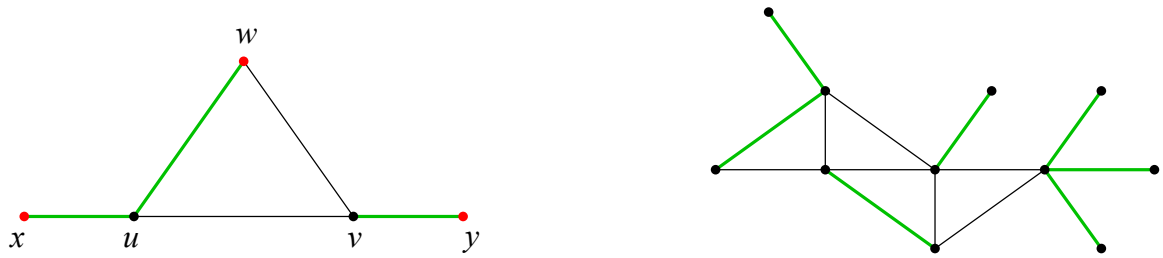
18.3.5. Для графа, показанного на рис.89, достаточно легко убедиться в том, что множество $S = \{u, v, w\}$ действительно является максимальным вершинно независимым множеством. В случае более сложно устроенных графов часто очень непросто понять, действительно ли то или иное подмножество вершин представляет собой вершинно независимое множество максимального размера. Данную ситуацию, однако, часто спасает тот факт, что, как и в случае максимального паросочетания в графе G, для рассматриваемой задачи имеется двойственная к ней экстремальная задача — задача о поиске так называемого минимального реберного покрытия графа.

Определение 18.17. *Реберным покрытием* графа G называется набор L ребер, покрывающий *все* вершины этого графа. Иными словами, любая вершина графа инцидентна одному из ребер, входящих в подмножество $L \subseteq E(G)$.

Как видно из определения, реберное покрытие в графе существует лишь тогда, когда в G отсутствуют изолированные вершины. Другими словами, реберное покрытие существует в случае, когда минимальная степень $\delta(G)$ вершин в графе G строго больше нуля.

Понятно, что все множество $E(G)$ ребер в графе G с $\delta(G) > 0$ является реберным покрытием графа G. Нас, как обычно, будет интересовать минимальный набор ребер, покрывающий все вершины графа.

Определение 18.18. Реберное покрытие L графа G называется *минимальным,* если размер любого другого реберного покрытия L' графа G больше или равен $|L|$. Размер минимального реберного покрытия графа G обозначается обычно через $\beta'(G)$.



(a) (b)

Рис. 90

Рассмотрим в качестве примера граф, показанный на рис.90,a. Минимальное реберное покрытие такого графа состоит из трех ребер, помеченных зеленым цветом на рисунке. Действительно, ребра $\{x, u\}$ и $\{v, y\}$ нам необходимы, чтобы покрыть два листа нашего графа — вершины x и y. При этом остается непокрытой вершина w, которую можно покрыть, к примеру, ребром $\{u, w\}$. Таким образом, для этого графа $\beta'(G) = 3$.

На рис.90,b показан чуть более сложный граф G. Утверждается, что размер $\beta'(G)$ минимального реберного покрытия такого графа равен семи. Действительно, пять ребер нужны нам для того, чтобы покрыть пять листьев графа G. Минимальное количество ребер, покрывающих оставшиеся три вершины, равно двум.

18.3.6. Вернемся к вершинно независимым множествам и их связи с реберными покрытиями графа.

Лемма 18.19 (Слабая двойственность)**.** *В любом графе G с $\delta(G) > 0$ размер $|S|$ произвольного вершинно независимого множества S не превосходит величины $|L|$ произвольного реберного покрытия L графа G. В частности,*

$$\alpha(G) \leqslant \beta'(G). \tag{49}$$

Доказательство. Действительно, никакое ребро не может покрывать две вершины, принадлежащие вершинно независимому множеству. Как следствие, любое ребро из L покрывает либо одну, либо ноль вершин из S, так что $|L| \geqslant |S|$. □

Используя свойство слабой двойственности, достаточно просто убедиться в том, что найденные в разобранных выше примерах подмножества действительно оптимальны. Так, для графа G, показанного на рис.90,а, мы предъявили вершинно независимое множество $S = \{x, y, w\}$ (красные вершины на рисунке), а также реберное покрытие $L = \{\{x, u\}, \{u, w\}, \{v, y\}\}$ (зеленые ребра на рисунке). Так как их размеры совпадают, то первое действительно является максимальным вершинно независимым множеством, а второе — минимальным реберным покрытием в G.

(a) (b)

Рис. 91

18.3.7. Заметим, что любому вершинно независимому множеству в графе G отвечает некоторая клика в дополнении $\bar{G}$ к графу G (см.рис.91). Как следствие, задача поиска максимального независимого множества эквивалентна так называемой *задаче о клике* — задаче нахождения в заданном графе клики наибольшего размера. Количество вершин в клике максимального размера обозначается обычно через $\omega(G)$. Мы показали, таким образом, что

$$\omega(G) = \alpha(\bar{G}).$$

Задача о клике довольно встречается на практике — например, она возникает в социологии при изучении в некотором сообществе подгруппы людей, каждый человек в которой знает любого другого члена этой подгруппы. Не менее часто встречается на практике и исходная задача — задача о поиске максимального вершинно независимого множества. Все это делает крайне актуальным поиск эффективного алгоритма, позволяющего подобного рода задачи решать. К сожалению, ситуация с эффективными алгоритмами для этих задач складывается даже хуже, чем для задач поиска максимальных паросочетаний в графах.

Заметим, прежде всего, что как и в случае максимального паросочетания, жадный алгоритм в данных задачах также не применим — несложно убедиться, что вместо максимального такой алгоритм дает нам так называемое наибольшее по включению независимое множество.

Определение 18.20. Вершинно независимое множество S называется *наибольшим по включению,* если при добавлении к нему любой другой вершины x графа G множество $S \cup \{x\}$ перестает быть независимым.

В качестве примера выберем для показанного на рис.89 графа G в качестве начальной вершины жадного алгоритма вершину x (рис.89,b). Удаляя смежные с ней вершины u, v и w, мы в результате работы алгоритма получаем независимое наибольшее по включению множество $K = \{x, y\}$ вершин, не являющееся максимальным. Несложная оптимизация этого алгоритма, заключающаяся в выборе на каждом шаге вершины с минимально возможной степенью, также не гарантирует получения максимального независимого множества.

В задаче поиска максимального паросочетания ситуацию спасает критерий Бержа существования в графе максимального паросочетания — мы далее покажем, как с его помощью построить довольно эффективные алгоритмы ее решения. Проблема с вершинно независимыми множествами состоит в том, что для них подобного рода простые критерии оптимальности отсутствуют. Следствием этого является тот факт, что с алгоритмической точки зрения задача поиска максимального вершинно независимого множества простых решений не имеет — в семидесятые годы прошлого века было показано, что задача поиска максимального независимого множества (independent set problem) NP-трудна.

18.4. Мы поняли, как связаны между собой паросочетания (то есть реберно независимые множества) и вершинные покрытия, а также вершинно независимые множества и реберные покрытия. Разберемся теперь с тем, как связаны между собой вершинное покрытие и вершинно независимое множество, а также паросочетание и реберное покрытие.

18.4.1. Начнем с вершинного независимого множества и вершинного покрытия. В качестве примера рассмотрим граф G, показанный на рис.89. Его максимальное вершинное независимое множество состоит из подмножества вершин $S = \{u, v, w\}$ (рис.89,a), а минимальное вершинное покрытие — из подмножества $K = \{x, y\}$ (рис.89,b), являющегося дополнением к S в $V(G)$. Оказывается, что этот факт имеет место и в общем случае, причем для любых, а не только оптимальных вершинных независимых множеств и вершинных покрытий. Именно, справедлива следующая

Теорема 18.21. *Подмножество S множества $V(G)$ вершин графа G является независимым тогда и только тогда, когда подмножество $K = V(G) \setminus S$ образует вершинное покрытие графа G.*

Доказательство. По определению, S является независимым в случае, когда никакие две вершины из S не соединены ребром. Иными словами, если один из двух концов некоторого ребра $e \in E(G)$ входит в вершину подмножества S, то второй конец этого ребра обязан входить в вершину, принадлежащую $K = V(G) \setminus S$. Если же ребро e не является инцидентным ни одной вершине подмножества S, то оно тем более обязано быть инцидентным каким-то вершинам множества K. Иными словами, любое ребро графа G обязано быть инцидентным хотя бы одной вершине множества K. Но это и означает, что K образует вершинное покрытие графа G.

Обратно, предположим, что у нас имеется какое-то вершинное покрытие K. Это означает, в частности, что в графе не существует ребра, оба конца которых не покрыты вершинами из K. Следовательно, вершины из $V(G) \setminus K$ покрывают не более одной вершины каждого ребра, то есть образуют вершинно независимое множество. □

Следствие 18.22. *Для любого графа G справедливо равенство*

$$\alpha(G) + \beta(G) = n = |V(G)|. \tag{50}$$

Доказательство. Пусть S есть максимальное независимое множество графа G, $|S| = \alpha(G)$. По теореме (18.21), $V(G) \setminus S$ образует какое-то (может, и не минимальное) вершинное покрытие K

графа G. Как следствие,

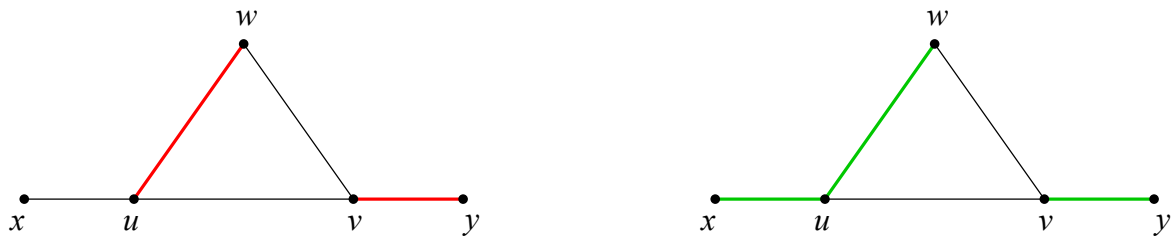
$$n = \alpha(G) + |K| \geqslant \alpha(G) + \beta(G).$$

Аналогично, пусть K есть минимальное вершинное покрытие графа G, $|K| = \beta(G)$. По теореме (18.21), $V(G) \setminus K$ представляет собой какое-то (может, и не максимальное) независимое множество S графа G. Поэтому

$$n = \beta(G) + |S| \leqslant \beta(G) + \alpha(G).$$

Из полученных неравеств следует, что $\alpha(G) + \beta(G) = n$. □

18.4.2. Давайте теперь попытаемся разобраться с реберными покрытиями и паросочетаниями. В случае вершинного покрытия и вершинно независимого множества мы доказали, что дополнение к любому вершинному покрытию обязано являться вершинно независимым множеством и наоборот. Несложно убедиться, что в случае ребер аналогичный факт места не имеет.



(a) (b)

Рис. 92

В качестве примера рассмотрим граф G, показанный на рис.92. Набор ребер $M = \{\{u, w\}, \{v, y\}\}$ образует паросочетание, то есть представляет собой реберно независимое множество (красные ребра на рис.92,a), однако его дополнение — набор ребер $E(G)\setminus M = \{\{x, y\}, \{u, v\}, \{v, w\}\}$ — реберным покрытием G не является (вершина y остается непокрытой этим набором). Обратно, набор ребер $L = \{\{x, u\}, \{u, w\}, \{v, y\}\}$ образует реберное покрытие графа G (зеленые ребра на рис.92,b), однако дополнение к нему — набор ребер $E(G) \setminus L = \{\{u, v\}, \{v, w\}\}$ — паросочетания в графе G не образует.

Несмотря на это, числа $\alpha'(G)$ и $\beta'(G)$ связаны между собой — оказывается, сумма этих чисел равна количеству $|V(G)|$ *вершин* графа G.

Теорема 18.23 (Gallai, 1959)**.** *Для любого графа G с $\delta(G) > 0$ справедливо равенство*

$$\alpha'(G) + \beta'(G) = n, \tag{51}$$

где $n = |V(G)|$ — количество вершин в графе G.

18.4.3. Прежде чем доказывать теорему Галлаи, исследуем структуру остовного подграфа $H = (V(G), L)$, индуцированного ребрами минимального реберного покрытия L (рис.90).

Как видно из рис.90,b, подграф H состоит из нескольких связных компонент, каждая из которых представляет собой некоторое дерево. Оказывается, что это дерево имеет специальный вид, а именно, представляет собой *звезду* — дерево, в котором лишь одна вершина может иметь степень, большую единицы (рис.93,a). Докажем все эти утверждения строго.

(a) (b)

Рис. 93

Лемма 18.24. *Остовный подграф H, порожденный набором ребер, входящих в минимальное реберное покрытие L графа G, представляет собой объединение связных компонент, каждая из которых представляет собой звезду.*

Доказательство. Прежде всего, заметим, что в H циклов быть не должно — в противном случае мы бы могли всегда разорвать цикл, убрав одно из ребер $e \in C$, не нарушив при этом покрытие вершин $V(C)$ оставшимися ребрами. Следовательно, H представляет собой остовный лес графа G.

Теперь предположим, что в остовном подграфе H у нас нашлось ребро e, обе инцидентные вершины которого имеют степени, большие единицы (см.рис.93,b). В этом случае мы всегда такое ребро можем удалить, не нарушив покрытия ребрами остовного подграфа $H - e$ всех вершин графа G. Продолжая описанный процесс до тех пор, пока у нас не останется подобных ребер, мы получим лес, каждая связная компонента которого представляет собой звезду. □

18.4.4. Перейдем к доказательству теоремы Галлаи. Идея этого доказательства состоит в следующем. Мы вначале возьмем максимальное паросочетание M размера $\alpha'(G)$ и построим по нему некоторое реберное покрытие L, количество ребер в котором окажется равным $n - \alpha'(G)$. Так как L — какое-то, не обязательно минимальное реберное покрытие, то

$$n - \alpha'(G) \geqslant \beta'(G) \qquad \Longrightarrow \qquad n \geqslant \alpha'(G) + \beta'(G).$$

Затем мы возьмем минимальное реберное покрытие L, $|L| = \beta'(G)$, и построим по нему паросочетание M размера $|M| = n - \beta'(G)$. Так как M — произвольное паросочетание, то

$$n - \beta'(G) \leqslant \alpha'(G) \qquad \Longrightarrow \qquad n \leqslant \alpha'(G) + \beta'(G).$$

Как следствие, n окажется равным $\alpha'(G) + \beta'(G)$.

Приступим теперь к реализации этого плана. Пусть M — какое-то максимальное паросочетание в графе G, $|M| = \alpha'(G)$ (красные ребра на рис.92,a). Так как каждое ребро паросочетания покрывает две различные вершины графа G, то общее количество вершин, покрытых паросочетанием M, равно $2|M| = 2\alpha'(G)$. Обозначим через U подмножество множества $V(G)$ вершин графа G, не покрытых этим паросочетанием (вершина x на рис.92,a). Размер этого подмножества равен, очевидно, $n - 2\alpha'(G)$, где $n = |V(G)|$. Кроме того, любые две вершины x, y подмножества U несмежны между собой — в противном случае мы бы могли добавить к M ребро $\{x, y\}$ и получить паросочетание $M' = M \cup \{x, y\}$ большего размера, что невозможно.

Так как $\delta(G) > 0$, то из каждой вершины x подмножества U исходит хотя бы одно ребро в одну из вершин u подмножества $V(G) \setminus U$ (ребро $\{x, u\}$ на рис.92,a). Выберем тогда по одному ребру, исходящему из каждой вершины подмножества U, и сформируем из них подмножество F множества $E(G)$ ребер графа G. Размер подмножества F совпадает с размером подмножества U и равен $n - 2\alpha'(G)$. По построению, $M \cap F = \varnothing$.

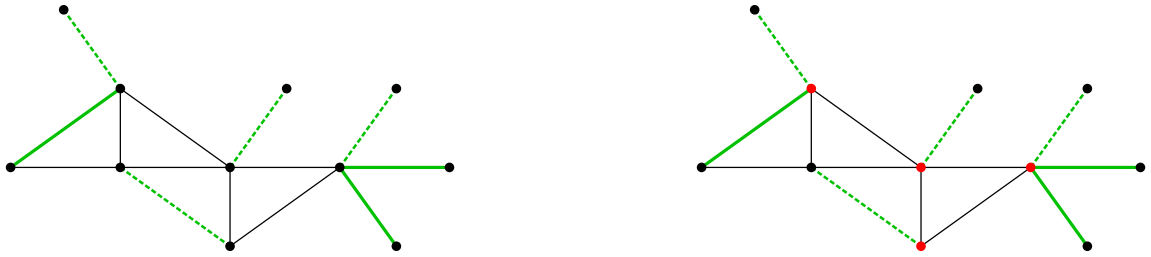
Заметим теперь, что подмножество $M \cup F$ множества $E(G)$ ребер графа G покрывает все вершины нашего графа. Подсчитаем количество ребер, входящих в подмножество $M \cup F$:

$$|M \cup F| = |M| + |F| = \alpha'(G) + n - 2\alpha'(G) = n - \alpha'(G).$$

Размер подмножества $M \cup F$, как и размер любого реберного покрытия, больше или равен размеру $\beta'(G)$ *минимального* реберного покрытия L графа G, поэтому

$$n - \alpha'(G) \geqslant \beta'(G) \qquad \Longrightarrow \qquad n \geqslant \alpha'(G) + \beta'(G).$$

Для завершения доказательства теоремы достаточно доказать обратное неравенство.



(a) (b)

Рис. 94

Для этого рассмотрим некоторое минимальное реберное покрытие L графа G, $|L| = \beta'(G)$ (зеленые ребра на рис.94,a). Мы знаем, что любая компонента связности остовного подграфа H, построенного на ребрах подмножества L, представляет собой звезду. Давайте теперь выберем в каждой из этих звезд по одному ребру (зеленые пунктирные ребра на рис.94,a). Эти ребра образуют какое-то паросочетание M в исходном графе G, причем размер $|M|$ этого паросочетания совпадает с количеством центров звезд (красные вершины на рис.94,b).

Теперь заметим, что количество концевых вершин каждой звезды в точности совпадает с общим количеством ребер в звезде. Следовательно, количество вершин, не являющихся центрами звезд (черные вершины на рис.94,b), равно общему количеству $|L| = \beta'(G)$ ребер в минимальном реберном покрытии графа G (зеленые ребра на рис.94,b). Общее же количество $n = |V(G)|$ вершин в графе G, таким образом, равно

$$n = |M| + |L| = |M| + \beta'(G).$$

Но M — это какое-то паросочетание в графе G, поэтому его размер меньше или равен размеру $\alpha'(G)$ максимального паросочетания в этом графе. Следовательно,

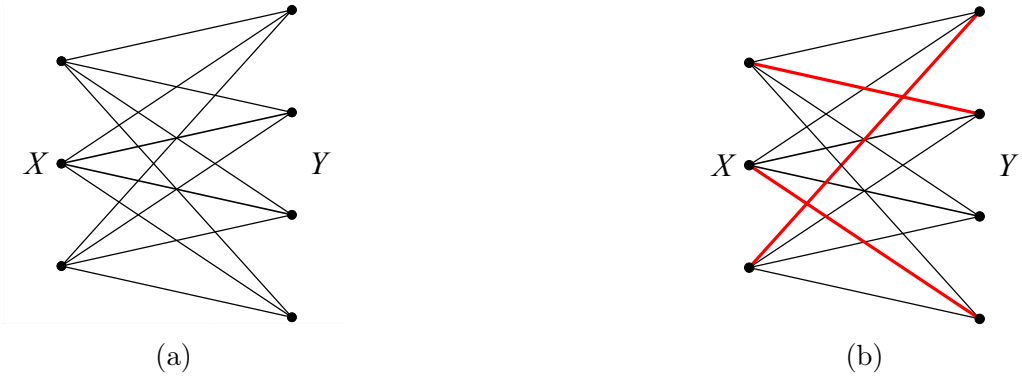
$$n = |M| + \beta'(G) \leqslant \alpha'(G) + \beta'(G).$$

Сравнивая это неравенство с доказанным выше неравенством $n \geqslant \alpha'(G) + \beta'(G)$, мы и получаем требуемый результат. □

19 Паросочетания в двудольных графах. Алгоритм Куна поиска максимального паросочетания в двудольном графе

19.1. Как правило, бóльшая часть задач, встречающихся на практике и сводящихся к поиску паросочетания в графе, связанна с поиском максимального паросочетания в двудольном графе.

Приведем характерный пример такого рода задач.



(a) (b)

Рис. 95

Пусть в группе имеется m студентов, которых необходимо распределить по n компаниям на летнюю практику. Для моделирования этой задачи распределения мы можем ввести двудольный граф G, блоки X и Y которого содержат m и n вершин соответственно. Любые две вершины $x \in X$ и $y \in Y$ этого графа соединяются ребром в случае, если квалификация студента удовлетворяет данную компанию, а сама компания, в свою очередь, устраивает данного студента. На рис.95,a в качестве примера изображен полный двудольный граф $K_{3,4}$, моделирующий некоторую идеальную ситуацию, при которой у нас любой студент готов пройти практику в любой компании, а любая компания, в свою очередь, готова принять на практику любого студента. Если же у нас какой-то студент не готов пойти в ту или иную компанию, или в случае, когда компания не готова того или иного студента на практику пригласить, мы получим некоторый подграф данного полного двудольного графа.

Предположим теперь, что в каждую компанию мы можем устроить только лишь одного студента. Последнее условие означает, что задача распределения студентов по летним практикам сводится к задаче поиска максимального паросочетания в построенном двудольном графе (см.рис.95,b, на котором красными ребрами помечено некоторое решение данной задачи).

19.2. Наша следующая задача — установить критерий максимальности паросочетания M в двудольном графе $G[X, Y]$. На основании леммы 18.14 нам достаточно предъявить для этого некоторое вершинное покрытие K, размер которого совпадает с количеством ребер в M. Как мы уже отмечали в первом параграфе, в общем случае это удается сделать не всегда. Оказывается, однако, что в двудольном графе такое вершинное покрытие обязательно найдется. Гарантирует этот факт нам следующая

Теорема 19.1 (Сильная двойственность; Кёниг, 1931; Эгервари, 1931)**.** *В любом двудольном графе G количество $\alpha'(G)$ ребер в максимальном паросочетании равно количеству $\beta(G)$ вершин в минимальном вершинном покрытии G:*

$$\alpha'(G) = \beta(G). \tag{52}$$

Доказательство легче всего провести, используя вершинную теорему Менгера. Именно, добавим к вершинам графа $G[X, Y]$ две дополнительные вершины x и y, и соединим с x все вершины блока X, а с y — все вершины блока Y (рис.96). Заметим, что в полученном графе G' любое подмножество S множества $V(G')$ вершин разделяет вершины x и y тогда и только тогда, когда это же подмножество покрывает все ребра исходного графа $G[X, Y]$ (зеленые вершины на рисунке). Кроме того, любое множество путей в графе G', соединяющих вершины x и y и не

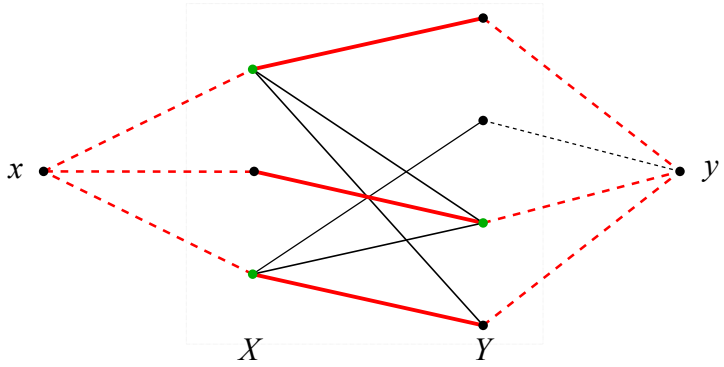
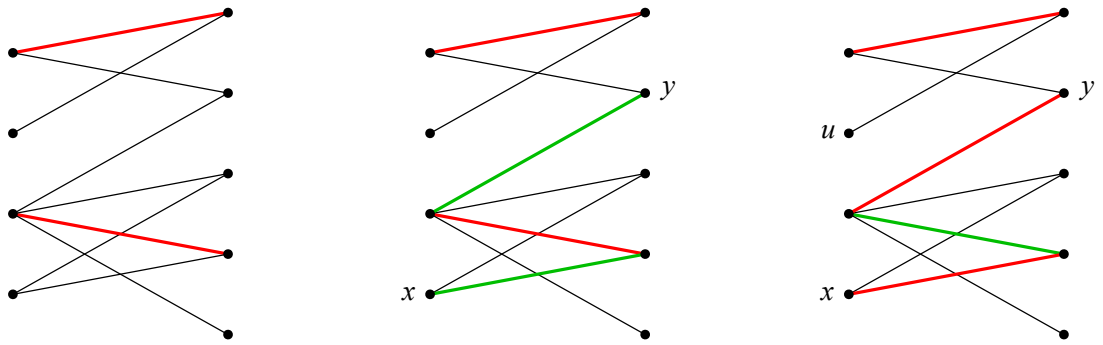


Рис. 96

имеющих общих внутренних вершин, образует некоторое паросочетание в графе $G[X, Y]$ (красные ребра на рисунке). Верно и обратное: любое паросочетание M в G может быть продолжено до путей, соединяющих в графе G' вершины x и y, и не имеющих общих внутренних вершин. Теперь для доказательства равенства $\alpha'(G) = \beta(G)$ достаточно сослаться на вершинную теорему Менгера. □

19.3. Итак, в отличие от графов общего вида, в случае двудольных графов у нас вместо условия слабой оптимальности имеет место условие сильной оптимальности (52). Это условие, в свою очередь, служит теоретической основой существования эффективных алгоритмов поиска максимального паросочетания в графе $G[X, Y]$. Мы опишем простейший из них — так называемый *алгоритма Куна.*



(a) (b) (c)

Рис. 97

19.3.1. В качестве первого шага описываемого алгоритма построим в графе $G[X, Y]$, $|X| \leqslant |Y|$, произвольное паросочетание M (см.рис.97,a). Для этого мы можем, например, перебрать все вершины $x_i \in X$ и добавить к M ребро $\{x_i, y_j\}$ в случае, если вершина y_j еще не покрыта паросочетанием M (смотри ребра, помеченные красным цветом на рис.97,a). Если при этом все вершины x_i оказались покрытыми паросочетанием M, то мы на этом работу алгоритма прекращаем.

В противном случае мы выбираем произвольную вершину $x \in X$, еще не покрытую паросочетанием M, и запускаем из x поиск в глубину, строя из этой вершины M-чередующиеся пути (см.рис.97,b). В случае, если хотя бы один из строящихся путей P в какой-то момент окажется M-дополняющим (путь $P = (x, y)$ на рис.97,b), мы останавливаем поиск в глубину, меняем ребра этого пути, покрытые M, на непокрытые M (то есть вместо M рассматриваем паросочетание

$M' = M\Delta E(P)$), и добавляем их в строящееся паросочетание (рис.97,c). В противном случае переходим к следующей еще не покрытой паросочетанием M вершине $x' \in X$.

19.3.2. Наша следующая задача — убедиться в корректности описанного выше алгоритма, а именно, доказать, что по окончании его работы мы действительно получим максимальное паросочетание M в двудольном графе $G[X, Y]$. На основании условия сильной оптимальности (52) в случае максимального паросочетания M в графе $G[X, Y]$ обязано существовать вершинное покрытие K, размер которого совпадает с количеством ребер в M. Все, что нам остается сделать — это такое вершинное покрытие предъявить.

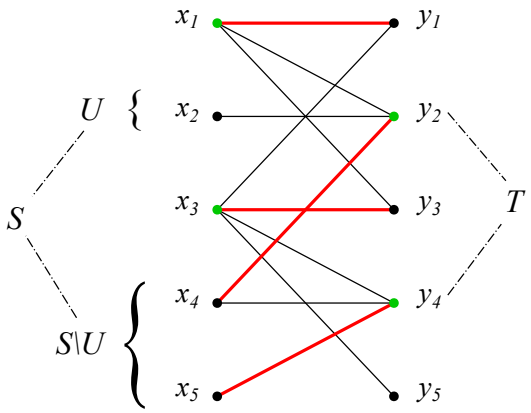


Рис. 98

Обозначим через U подмножество вершин из X, не покрытых паросочетанием M (вершина x_2 на рис.98), а через Z — подмножество всех вершин графа G, связанных M-чередующимися путями с вершинами из U (вершины x_2, y_2, x_4, y_4, x_5 на рис.98). Пусть $S = Z \cap X$, $T = Z \cap Y$, а $K = (X \setminus S) \cup T$ (вершины подмножества K помечены зеленым цветом на рис.98). Покажем, что K представляет собой вершинное покрытие графа G.

Действительно, все ребра, исходящие из $X \setminus S$, покрыты вершинами из K. Нам остается убедиться, что все ребра, исходящие из S, также покрыты вершинами из K. Понятно, что все ребра, исходящие из S в T, вершинами из K покрыты. Мы докажем требуемое, если поймем, что ребра из S в $Y \setminus T$ отсутствуют. Предположим противное, а именно, предположим, что у нас имеется какое-то ребро из вершины $x \in S$ в вершину $y \in Y \setminus T$. Заметим, прежде всего, что такое ребро не может принадлежать M. Действительно, вершины подмножества U ребрами из M не могут быть покрыты по определению U, а вершины из $S \setminus U$ покрыты ребрами из M, которые соединяют их с вершинами из T, а значит, другие ребра, принадлежащие M, из этих вершин выходить не могут. Следовательно, из x выходит ребро $e \notin M$. Но тогда, дойдя до вершины x, алгоритм поиска в глубину прошел бы по e и занес бы вершину y в подмножество T, а это противоречит тому, что $y \in Y \setminus T$.

Теперь сосчитаем количество вершин в K. Заметим для этого, что и вершины из $X \setminus S$, и вершины из T покрыты ребрами паросочетания M. Так как любая вершина из T соединена с какой-то вершиной из S ребром, принадлежащим M, то никакое ребро из M не может соединять вершину из T с вершиной из $X \setminus S$. Как следствие, количество ребер в M больше или равно $|K|$. С учетом леммы (18.14) мы и получаем, что $|K| = |M|$. □

Замечание 19.2. Доказательство корректности алгоритма Куна является, по сути, еще одним, независимым доказательством теоремы Кёнига-Эгервари.

19.3.3. Итак, в результате работы алгоритма Куна мы получаем максимальное паросочетание M в двудольном графе $G[X, Y]$, которое покрывает нам $|M|$ вершин как в X, так и в Y.

Определение 19.3. Паросочетание M в двудольном графе $G[X, Y]$ называется X-насыщенным, если любая вершина из блока X покрыта этим паросочетанием.

Предположим, что построенное нами паросочетание X-насыщенным не является. Это означает, что в X существует некоторое подмножество $U \subset X$ вершин, не покрытых M. Рассмотрим произвольную вершину $x \in U$. В процессе работы алгоритма Куна мы построили M-чередующееся дерево с корнем в вершине x. Давайте поговорим про это дерево немного поподробнее.

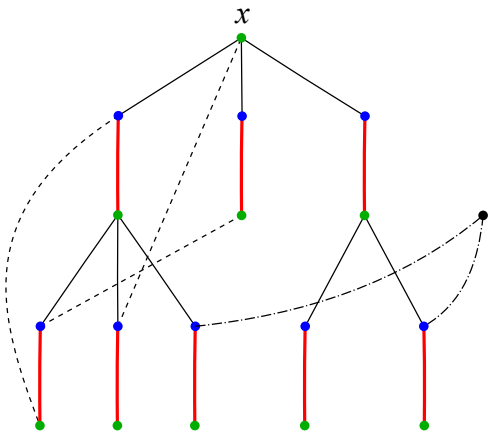


Рис. 99

Характерный пример такого дерева показан на рис.99. Зеленым цветом на этом рисунке помечены вершины, принадлежащие доле X, а синим цветом — вершины, принадлежащие Y. Ребра, входящие в паросочетание M, окрашены в красный цвет. Так как любой исходящий из x M-чередующийся путь обязан заканчиваться в вершинах доли X, то листьями M-чередующегося дерева являются вершины, окрашенные в зеленый цвет. Наконец, ребра паросочетания, входящие в дерево, устанавливают взаимно-однозначное соответствие между всеми синими вершинами и всеми зелеными вершинами, отличными от корневой вершины x.

Понятно, что помимо ребер, входящих в M-чередующееся дерево, в графе $G[X, Y]$ могут существовать и какие-то другие ребра, ему не принадлежащие. Однако имеется принципиальное различие между ребрами, исходящими из синих вершин дерева, и ребрами, исходящими из зеленых вершин. В принципе, ничто не мешает существованию ребер, исходящих из синих вершин дерева и входящих в какие-то другие вершины $x' \in X$ графа $G[X, Y]$ (см. штрихпунктирные ребра на рисунке). Однако для зеленых вершин такая ситуация невозможна — любое ребро $e = \{x', y'\}$, отличное от ребра M-чередующегося дерева и исходящее из зеленой вершины x', обязано заканчиваться в какой-то синей вершине того же дерева (см. пунктирные ребра на рисунке). Действительно, в противном случае, дойдя до вершины x', алгоритм прошел бы по этому ребру ребру в вершину $y' \in Y$ и включил бы ее в M-чередующее дерево. Следствием этого наблюдения является тот факт, что подмножество $N(S)$ вершин, смежных с множеством S зеленых вершин M-чередующегося дерева, совпадает с подмножеством T синих вершин этого дерева.

19.3.4. Сделанные выше наблюдения, связанные со структурой M-чередующегося дерева в двудольном графе $G[X, Y]$, лежат в основе доказательства еще одной очень важной теоремы,

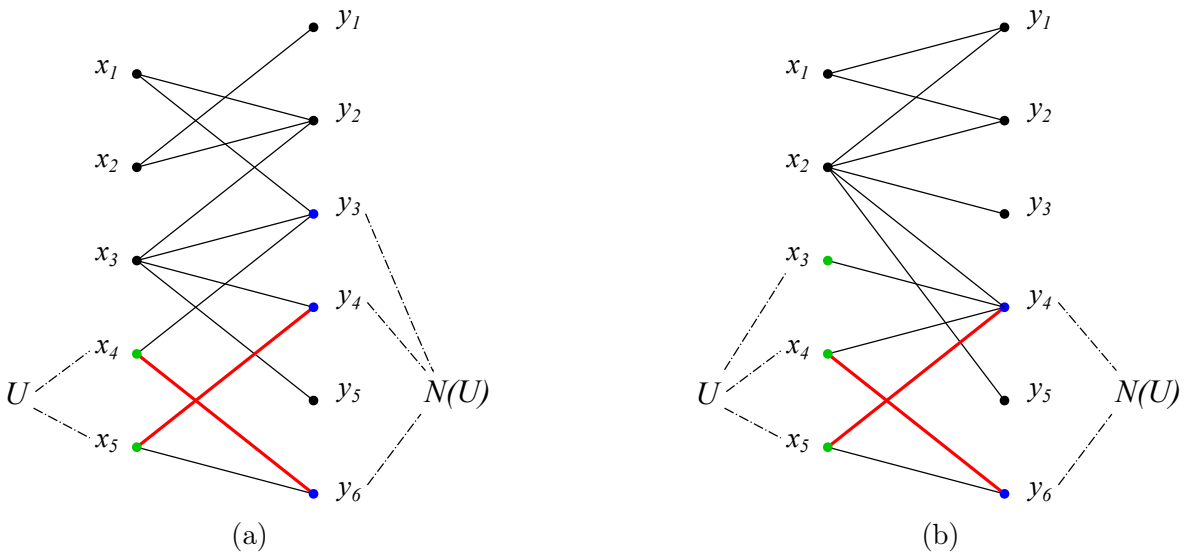


Рис. 100

связанной с паросочетаниями в двудольном графе — теоремой Холла о X-насыщенном паросочетании в графе.

Рассмотрим произвольный двудольный граф G с блоками X и Y (рис.100,a). Напомним, что для любого подмножества U множества вершин мы через $N(U)$ обозначаем подмножество вершин, смежных со всеми вершинами из U. Понятно, что в случае двудольного графа для любого подмножества U вершин блока X (вершины, помеченные зеленым цветом на рис.100,a) все вершины множества $N(U)$ принадлежат блоку Y (синие вершины на рисунке).

Теорема 19.4 (Ph.Hall, 1935)**.** *Для того, чтобы в двудольном графе $G[X,Y]$ существовало X-насыщенное паросочетание, необходимо и достаточно, чтобы для любого подмножества U множества X мощность соответствующего ему подмножества $N(U) \subset Y$ смежных с U вершин была бы больше или равна мощности $|U|$ подмножества U:*

$$|U| \leqslant |N(U)| \qquad \forall\, U \subseteq X. \tag{53}$$

Замечание 19.5. Неформально говоря, теорема Холла утверждает, что каждое подмножество $U \subseteq X$ должно иметь в Y достаточное количество смежных вершин. В частности, совершенно очевидно, что для существования X-насыщенного паросочетания необходимо выполнение условия $|X| \leqslant |Y|$. Теорема 19.4, по сути, обобщает это несложное наблюдение на произвольное подмножество U множества X.

Замечание 19.6. Теорему Холла часто называют также теоремой о деревенских свадьбах. Это название связано со следующей формулировкой этой задачи, восходящей к известному немецкому математику первой половины двадцатого века Герману Вейлю. В деревне относительно каждой пары юноша-девушка известно, дружат они или нет. Тогда, если для любых k юношей объединение подмножеств их подруг содержит по крайней мере k девушек, то каждый юноша сможет выбрать себе будущую жену из числа своих же подруг.

19.3.5. Теорема Холла легко доказывается с использованием уже известных нам минимаксных теорем, таких, как теорема Менгера или теорема Кёнига-Эгервари. Мы же для полноты изложения приведем здесь прямое доказательство теоремы Холла.

Сразу заметим, что необходимость условия (53) совершенно очевидна. Действительно, пусть существует хотя бы одно подмножество U, для которого это условие не выполняется. На рис.100,b изображена подобная ситуация — в этом примере смежное с подмножеством U первых трех вершин подмножество $N(U) \subset Y$ содержит всего две вершины. Рассмотрим тогда двудольный подграф $G'[U, N(U)]$. В таком графе U-насыщенного паросочетания, очевидно, не существует. Но это автоматически означает, что не существует X-насыщенного паросочетания и в исходном графе G. Действительно, все ребра, исходящие из U, приходят только в $N(U)$. Поэтому правильно выбрать ребра, выходящие из подмножества U, нам уже не удастся вне зависимости от того, насколько успешно мы справились с решением этой задачи для ребер, исходящих из подмножества $X \setminus U$.

Докажем теперь, что выполнение условий (53) гарантирует нам существование X-насыщенного паросочетания. Предположим, что в графе $G[X, Y]$ имеется какое-то паросочетание M, количество $|M|$ ребер в котором строго меньше $|X|$. Покажем, что в этом случае мы всегда сможем построить паросочетание M' с количеством $|M'|$ ребер, равным $|M| + 1$. Выберем для этого какую-то вершину $x \in X$, не покрытую паросочетанием M, и запустим из нее алгоритм Куна. Предположим, что в результате его работы алгоритм Куна не нашел M-дополняющий путь. В этом случае алгоритм Куна вместо такого пути выдаст нам M-чередующееся дерево (рис.99). Мы ранее отмечали, что количество $|S|$ вершин подмножества $S \subseteq X$, достижимых из x, на единицу больше количества $|T|$ вершин, достижимых из x и принадлежащих Y. Кроме того, мы показали, что в этом случае $N(S) = T$. Следовательно, при таком развитии событий

$$|N(S)| = |T| = |S| - 1,$$

что противоречит условиям Холла (53). □

Следствие 19.7. *Пусть в двудольном графе $|X| = |Y|$, и для любого подмножества U множества X выполняются условия (53). Тогда в этом графе существует совершенное паросочетание.*

19.3.6. Мы убедились в том, что теоремы Менгера, Форда-Фалкерсона, Кёнига-Эргевари и Холла достаточно тесно связаны между собой — из справедливости одной теоремы достаточно просто доказать справедливость остальных. К этому же классу результатов относятся и еще несколько важных именных теорем, таких, например, как теорема Дилуорса и теорема Мирского в теории частично упорядоченных множеств, матричная переформулировка теоремы Кёнига-Эгервари, теорема Биркгофа и другие (см. упражнения к этому параграфу). Каждая из них была в свое время доказана независимо от остальных, однако легче всего получить доказательства этих теорем, сводя их к одному из уже доказанных выше результатов.

20 Совершенные и максимальные паросочетания в произвольном графе. Теорема Татта. Формула Татта – Бержа

20.1. Вернемся к задаче поиска совершенного паросочетания в произвольном графе G, то есть паросочетания, покрывающего все вершины графа G.

20.1.1. Первый и основной вопрос, возникающий при решении данной задачи, состоит в следующем: а существует ли в принципе в заданном графе G хотя бы одно совершенное паросочетание? Иными словами, нам хотелось бы понимать, какими свойствами должен обладать граф G, чтобы в нем гарантированно существовало совершенное паросочетание.

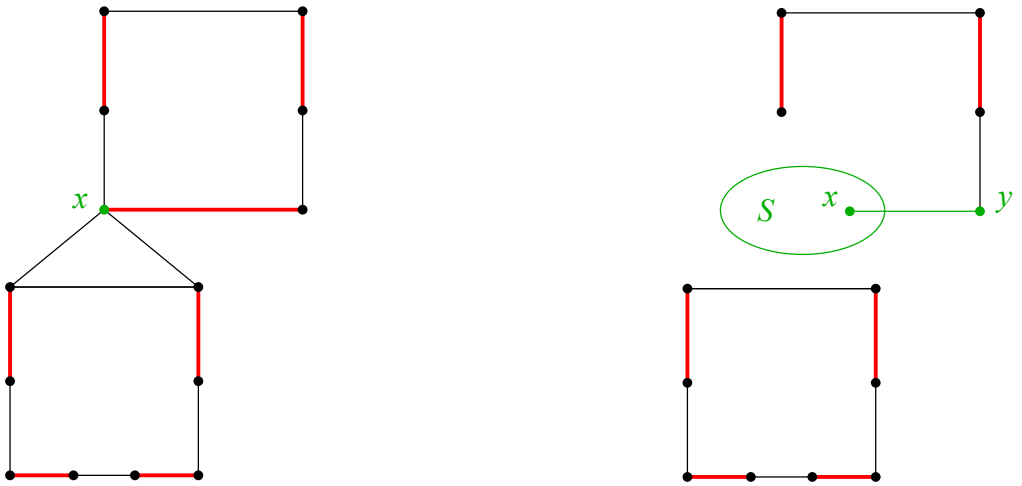
(a) (b)

Рис. 101

Ранее мы уже упоминали, что совершенное паросочетание невозможно построить в случае, когда количество вершин в графе нечетно. К сожалению, довольно простое условие четности вершин в графе G не гарантирует нам существование совершенного паросочетания в G. В качестве примера на рис.101, показан достаточно простые графы, построенные на четном числе вершин, совершенное паросочетание в которых отсутствует.

20.1.2. Необходимое и достаточное условие существования в графе совершенного паросочетания было получено в 1947 году одним из создателей современной теории графов Уильямом Таттом. Прежде чем формулировать соответствующий результат, давайте поймем, из каких соображений его можно получить.

Рассмотрим произвольный граф G, в котором существует совершенное паросочетание M. Выделим в этом графе некоторое абсолютно произвольное подмножество S множества $V(G)$ вершин графа G и удалим все вершины этого подмножества. Напомним, что при удалении вершин в графе G вместе с ними удаляются и все инцидентные им ребра. В полученном в результате такой операции графе $G - S$ могут оказаться несколько компонент связности. Будем называть отдельную компоненту связности, содержащую четное количество вершин, *четной* компонентой, а компоненту, содержащую нечетное количество вершин — *нечетной* компонентой.



(a) (b)

Рис. 102

На рис.102,a в качестве примера показан граф G, построенный на четырнадцати вершинах. Совершенное паросочетание M в таком графе состоит из семи ребер, помеченных красным цветом на рисунке. Включим в подмножество S единственную вершину x, помеченную зеленым цветом на рис.102,a. После удаления этой вершины исходный граф G распадется на две компоненты связности — четную и нечетную (рис. 102,b).

Давайте обратим внимание на произвольную нечетную компоненту графа $G - S$. В этой компоненте у нас остались ребра, входившие в исходное совершенное паросочетание M. Они, очевидно, покрывают только четное количество вершин этой компоненты. Следовательно, среди вершин рассматриваемой нечетной компоненты обязательно найдется хотя бы одна вершина y, покрытая в исходном графе G ребром, входящим в совершенное паросочетание M и соединяющим эту вершину y с одной из вершин удаленного подмножества S (ребро $\{x, y\}$ на рис.102,b).

Теперь рассмотрим все нечетные компоненты графа $G - S$. Как мы убедились выше, из каждой такой компоненты в исходном графе G у нас обязано исходить принадлежащее M ребро, соединяющее какую-то вершину x нечетной компоненты с какой-то *своей* вершиной y подмножества S. Следовательно, количество вершин в S должно быть не меньше количества нечетных компонент связности в графе $G - S$:

$$c_{\text{odd}}(G - S) \leqslant |S| \qquad\qquad \forall\, S \subset V(G). \tag{54}$$

Здесь $c_{\text{odd}}(G - S)$ — количество нечетных компонент связности графа $G - S$.

Заметим сразу, что из условий (54) следует, в частности, тот факт, что и сам граф G, и любая его связная компонента имеют четное число вершин. Действительно, выбирая в качестве S пустое множество $\varnothing$, мы получаем, что $c_{\text{odd}}(G) \leqslant |\varnothing| = 0$, а это и означает, что в графе G отсутствуют компоненты связности, имеющие нечетное число вершин.

20.1.3. Оказывается, что условие (54) является не только необходимым, но и достаточным для существования в графе G совершенного паросочетания.

Теорема 20.1 (Татт, 1947)**.** *В графе G существует совершенное паросочетание тогда и только тогда, когда для любого $S \subset V(G)$ количество $c_{\text{odd}}(G - S)$ нечетных компонент связности графа $G - S$ не превосходит мощности $|S|$ множества S.*

Доказательство. Необходимость условий (54) мы уже доказали, так что нам осталось доказать достаточность условий (54) для существования совершенного паросочетания в графе G. Доказывать это мы будем от противного. Именно, мы предположим, что у нас имеется граф G, в котором условия Татта (54) выполнены, а совершенное паросочетание отсутствует, и получим где-то противоречие.

Так как условия Татта для графа G выполнены, то этот граф построен на четном количестве вершин $n = |V(G)|$. Кроме того, этот граф отличен от полного графа K_n — в полном графе на четном количестве вершин совершенное паросочетание, очевидно, существует. Следовательно, в G имеется хотя бы одна пара несмежных между собой вершин. Соединим эту пару вершин ребром и покажем, что в результате этой операции условия Татта (54) не нарушатся.

Действительно, добавление к графу G ребра изменит количество нечетных компонент лишь в том случае, когда это ребро соединит две нечетные компоненты связности в графе $G-S$. Но при этом количество $c_{\text{odd}}(G-S)$ таких нечетных компонент уменьшится, а количество $|S|$ удаляемых вершин от добавления к графу ребра никак не изменится. Поэтому, если до добавления ребра условия (54) были выполнены, то они тем более останутся выполненными и после добавления к графу G ребра.

Давайте тогда возьмем исходный граф G и начнем добавлять к нему ребра, соединяющие несмежные вершины этого графа. На каком-то шаге мы обязательно придем к так называемому *насыщенному* графу G^* — графу, у которого совершенное паросочетание еще отсутствует, но прибавление к которому *любого* ребра приведет к появлению в новом графе совершенного паросочетания. Действительно, соединив все несмежные в графе G вершины, мы прийдем

к полному графу K_n на четном количестве n вершин, в котором совершенное паросочетание гарантированно существует. Следовательно, добавляя к G ребра так, чтобы в новых графах совершенное паросочетание отсутствовало, мы на каком-то шаге обязательно придем к ситуации, когда соединить *любую* пару несмежных вершин ребром и не получить в результате этой операции граф без совершенного паросочетания окажется невозможным, то есть придем к насыщенному графу G^*.

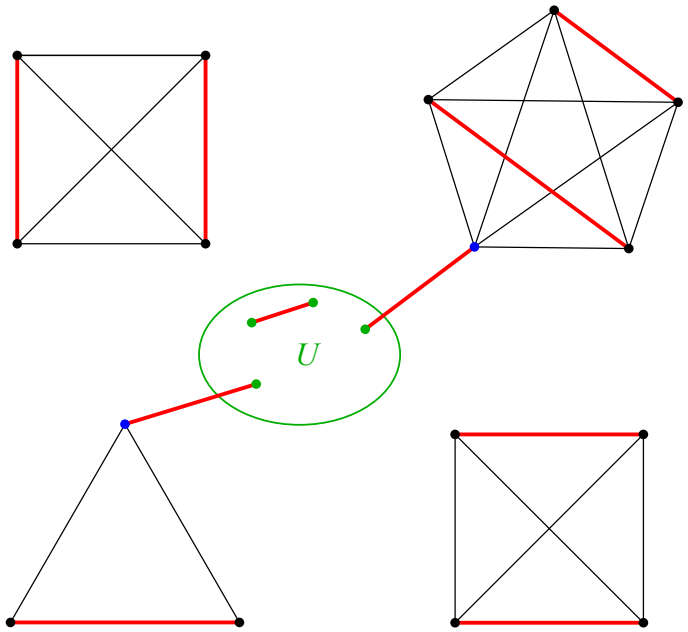


Рис. 103

Выделим теперь в полученном из G насыщенном графе G^* подмножество U вершин, каждая из которых соединена со всеми вершинами графа G^*, то есть подмножество U вершин, каждая из которых имеет степень $n-1$. Ниже мы покажем (см. лемму 20.2), что граф G^*-U, полученный после удаления такого подмножества U вершин, представляет собой объединение нескольких несвязных друг с другом полных графов. Так как в силу условий (54) в графе G^*-U не более $|U|$ нечетных компонент связности, то мы всегда сможем совершить следующие операции (см. рис.103):

1. построить в каждой компоненте связности графа G^*-U, содержащей четное число вершин, совершенное паросочетание (ребра, помеченные красным цветом на рис.103);
2. построить в каждой нечетной компоненте связности этого графа паросочетание, покрывающее все вершины, кроме одной (красные ребра на рис.103);
3. соединить каждую оставшуюся вершину в каждой из компонент нечетной связности (синие вершины на рис.103) с какой-то своей вершиной из множества U;
4. разбить на пары оставшееся множество $\widetilde{U}$ вершин в U и соединить их ребрами (красные ребра, соединяющие пары зеленых вершин на рис.103).

Последнюю операцию мы можем совершить потому, что, во-первых, все вершины в $\widetilde{U}$ смежны друг с другом по построению множества U, а во-вторых, количество таких вершин четно. Действительно, удаляя из четного числа вершин в графе G^* вершины, входящие в компоненты с четным числом вершин, мы вновь получаем четное число. Далее, вычитая из остатка нечетное число вершин любой компоненты нечетной связности вместе с одной из вершин множества U, мы вновь получаем четное число.

Таким образом, мы построили в графе G^* совершенное паросочетание, что противоречит определению насыщенного графа. Полученное противоречие доказывает достаточность выполнения условий (54) для существования совершенного паросочетания в графе. □

20.1.4. Итак, для доказательства теоремы Татта нам осталось доказать следующее утверждение.

Лемма 20.2. *Пусть U есть подмножество вершин насыщенного графа G^*, каждая из которых соединена со всеми вершинами графа G^*, то есть подмножество вершин, каждая из которых имеет степень $n-1$, $n = |V(G^*)|$. Тогда в графе $G^* - U$ любая компонента связности является полным графом.*

Доказательство. Будем доказывать это утверждение от противного. Именно, мы предположим, что это не так, то есть что в графе $G^* - U$ имеется компонента связности, не являющаяся полным графом. В такой компоненте имеется как минимум три вершины (компоненты связности, содержащие одну или две вершины, изоморфны полным графам K_1 и K_2), и в ней обязана существовать пара несмежных между собой вершин a и c, имеющих общую смежную вершину b (см. одно из упражнений к первой главе). Наша задача — показать, что отсутствие ребра $\{a, c\}$ противоречит насыщенности графа G^*.

Так как вершина $b \notin U$, то степень этой вершины отлична от $n-1$. Следовательно, в графе G^* найдется хотя бы одна вершина d, несмежная с b. Так как любая вершина из U смежна со всеми вершинами графа G^*, то вершина d также принадлежит $V(G^* - U)$. Таким образом, отсутствие ребра $\{a, c\}$ в графе $G^* - U$ означает, что в таком графе имеются по меньшей мере две пары несмежных между собой вершин.

Согласно определению насыщенного графа G^*, добавление к нему любого ребра e превращает $G^* + e$ в граф, в котором совершенное паросочетание уже существует. Важно отметить, что добавленное ребро e обязано этому совершенному паросочетанию принадлежать — в противном случае совершенное паросочетание можно было бы построить и в исходном графе G^*, что, согласно нашему предположению, невозможно.

Рассмотрим теперь графы $G_1 := G^* + \{a, c\}$ и $G_2 := G^* + \{b, d\}$, полученные добавлением к графу G^* ребер $\{a, c\}$ и $\{b, d\}$ соответственно. Как мы заметили выше, в таких графах существуют некоторые совершенные паросочетания M_1 и M_2, причем ребро $\{a, c\}$ входит в паросочетание M_1, а ребро $\{b, d\}$ принадлежит паросочетанию M_2. Образуем симметрическую разность $M_1 \Delta M_2 =: F$ двух паросочетаний и рассмотрим остовный подграф H графа G^* с $E(H) = F$. Так как паросочетания M_1 и M_2 являются совершенными, то любая вершина $x \in V(G^*) = V(G_1) = V(G_2)$ покрыта как ребром $e_1 \in M_1$, так и ребром $e_2 \in M_2$. В случае, если $e_1 = e_2$, эта вершина остается непокрытой ребрами из F. Если же $e_1 \neq e_2$, то степень вершины x в H равна двум. Следовательно, любое ребро в H обязано принадлежать некоторому циклу.

Возможны два случая. В первом из них ребра $\{a, c\}$ и $\{b, d\}$ принадлежат двум различным циклам C_1 и C_2 (рис.104,a). Во втором случае эти ребра лежат на одном и том же цикле $C_1 = C_2 =: C$ (рис.104,b). Покажем, что в обоих случаях мы всегда сможем сконструировать из паросочетаний M_1 и M_2 некоторое новое паросочетание M, покрывающее все вершины в графе G^*. Тем самым мы получим противоречие с тем, что граф G^* является насыщенным, доказывающее лемму.

Предположим вначале, что циклы C_1 и C_2 различны (рис.104,a). Выделим в цикле C_1 ребра, принадлежащие паросочетанию M_2 (синие ребра на рис.104,a), а в цикле C_2 — ребра из паросочетания M_1 (красные ребра на рис.104,a). Эти ребра, во-первых, покрывают все вершины

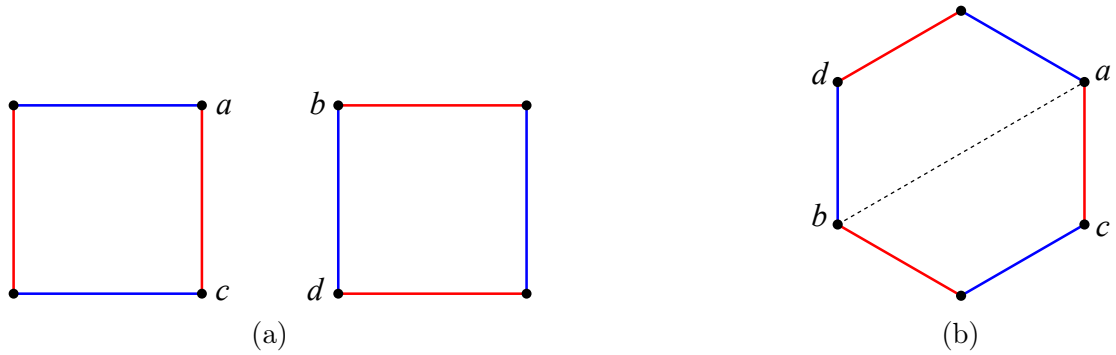


Рис. 104

циклов C_1 и C_2. Во-вторых, этот набор ребер не содержит ни ребро $\{a, c\}$, ни ребро $\{b, d\}$. Добавляя к полученному набору ребра, к примеру, паросочетания M_1, не входящие ни в цикл C_1, ни в цикл C_2, мы получаем некоторое совершенное паросочетание M в графе G^*.

Разберем теперь вариант, при котором ребра $\{a, c\}$ и $\{b, d\}$ принадлежат одному и тому же циклу C (рис.104,b). Будем двигаться по этому циклу от вершины b в направлении, задаваемым ребром $\{b, d\}$, до тех пор, пока не встретим одну из вершин ребра $\{a, c\}$. Пусть для определенности это будет вершина a (смотри рис.104,b). Пройденная цепочка ребер P_1 начинается и заканчивается ребрами, принадлежащими паросочетанию M_2 (синие ребра на 104,b). Оставшаяся же часть цикла C — цепочка P_2 — ограничена ребрами паросочетания M_1 (красные ребра на рисунке). Рассмотрим тогда ребро $\{a, b\} \in E(G^*)$, разбивающее цикл C на две половины. Добавим к нему ребра цепочки P_1, принадлежащие паросочетанию M_1, а также ребра из P_2, относящиеся к паросочетанию M_2. В результате мы получим паросочетание, покрывающее все вершины цикла C и не содержащее ни ребро $\{b, d\}$, ни ребро $\{a, c\}$. Дополняя это паросочетание ребрами из M_1, не входящими в цикл C, мы вновь получим совершенное паросочетание M в насыщенном графе G^*. □

20.2. Теперь перейдем к рассмотрению графов, в которых совершенные паросочетания отсутствуют.

20.2.1. Рассмотрим в графе G какое-то максимальное паросочетание M, $|M| = \alpha'(G)$. Так как это паросочетание совершенным не является, то оно оставляет непокрытым часть вершин графа G.

Определение 20.3. Дефицитом $\mathrm{def}(G)$ графа G называется количество его вершин, не покрытых максимальным паросочетанием M:

$$\mathrm{def}(G) := |V(G)| - 2|M| = n - 2\alpha'(G).$$

Предположим, что в графе G совершенное паросочетание отсутствует. Последнее, на основании теоремы Татта, означает, что в графе G найдутся такие подмножества S, для которых

$$c_{\mathrm{odd}}(G - S) - |S| > 0.$$

Определение 20.4. Подмножество $B \subseteq V(G)$, на котором достигается максимум функции $c_{\mathrm{odd}}(G - S) - |S|$, называется барьером.

Оказывается, для любого графа G имеется очень интересная связь между дефицитом графа и максимумом функции $c_{\mathrm{odd}}(G - S) - |S|$.

Теорема 20.5 (Берж, 1958)**.** *Для любого графа G справедливо равенство*

$$\operatorname{def}(G) = \max_{S \subset V(G)} \left[c_{\text{odd}}(G - S) - |S| \right]. \tag{55}$$

20.2.2. Прежде чем доказывать эту теорему, давайте вначале поймем, в чем ее смысл, а также в чем состоит важность этой теоремы с практической точки зрения. Заметим, прежде всего, что с учетом равенства $\operatorname{def}(G) = n - 2\alpha'(G)$ формулу (55) можно переписать в следующем виде:

$$\alpha'(G) = \frac{1}{2} \min_{S \subset V(G)} \left\{ n - (c_{\text{odd}}(G - S) - |S|) \right\}. \tag{56}$$

Рассмотрим теперь произвольное паросочетание M в графе G и произвольное подмножество S множества $V(G)$ вершин этого графа.

Лемма 20.6. *Для любого паросочетания M в графе G и произвольного подмножества S множества $V(G)$ вершин этого графа справедливо неравенство*

$$|M| \leqslant \frac{1}{2} \left(n - (c_{\text{odd}}(G - S) - |S|) \right). \tag{57}$$

Доказательство. Разобъем входящие в паросочетание M ребра на два блока — блок, содержащий ребра $e \in M$, оба конца которых принадлежат $G - S$, и блок, в котором хотя бы один конец $e \in M$ принадлежит S. Количество k_S ребер во втором блоке ограничен сверху размером множества S — максимум k_S достигается в случае, когда каждая вершина подмножества S соединена в G ребром $e \in M$ с некоторой (нечетной) компонентой графа $G - S$. Максимальное же значение числа k_C ребер в первом блоке достигается в случае, когда ребра из M покрывают каждую вершину в любой из четных компонент графа $G - S$ и все кроме одной вершины в каждой из нечетных компонент графа $G - S$. Следовательно,

$$k_C \leqslant \frac{|V(G)| - |S| - c_{\text{odd}}(G - S)}{2},$$

а отсюда с учетом $k_S \leqslant |S|$ и $|M| = k_S + k_C$ мы и получаем нужное нам неравенство (57). □

Доказанная лемма говорит нам о том, что у нас имеются две связанные между собой экстремальные задачи — задача поиска максимального паросочетания в графе G и задача поиска в этом графе барьера B, то есть подмножества множества $V(G)$ вершин, доставляющего минимум функции, стоящей в правой части неравенства (57). При этом очевидным следствием этой леммы является условие слабой оптимальности вида

$$\alpha'(G) \leqslant \frac{1}{2} \min_{S \subset V(G)} \left\{ n - (c_{\text{odd}}(G - S) - |S|) \right\}.$$

Как и любое другое условие слабой оптимальности, его можно использовать для доказательства того, что то или иное паросочетание M является максимальным. Именно, если нам удалось найти такое M и такое подмножество $B \subseteq V(G)$, для которых неравенство (57) обращается в равенство, то M гарантированно является максимальным паросочетанием, а B — барьером в графе G.

В качестве примера рассмотрим граф Сильвестра, показанный на рис.**??**. Помеченное красным цветом паросочетание M оставляет непокрытыми две вершины такого графа. Нам хочется доказать, что это паросочетание является максимальным. Для этого заметим, что после удаления

центральной вершины x графа G мы получаем три нечетные компоненты в графе $G-S$, $S=$ $=\{x\}$. Отсюда на основании формулы (55) мы заключаем, что M действительно является максимальным паросочетанием, а подмножество $B=\{x\}$ — барьером для этого графа.

Заметим теперь, что мы уже имели оценку сверху на $\alpha'(G)$ в терминах вершинного покрытия — именно, в первом параграфе мы доказали, что $\alpha'(G)\leqslant\beta(G)$. Основная проблема с этой оценкой состоит в том, что в общем случае это неравенство может оказаться строгим. При этом мы знаем, что в частном случае двудольного графа такой неприятности произойти не может — в этом случае мы гарантированно имеем равенство $\alpha'(G)=\beta(G)$. Последний факт позволяет нам, в частности, построить эффективный алгоритм поиска максимального паросочетания в двудольном графе — алгоритм Куна.

Вернемся теперь к теореме Бержа, а точнее, к ее переформулировке — равенству (56), часто называемому формулой Татта-Бержа. Эта формула, по сути, представляет собой условие сильной оптимальности — оно говорит нам о том, что в отличие от условия $\alpha'(G)\leqslant\beta(G)$ количество $\alpha'(G)$ ребер в максимальном паросочетании обязано совпадать с минимумом функции, стоящей в правой части неравенства (57), для любого, не обязательно двудольного графа G. И, как мы покажем чуть ниже, именно это условие позволит нам построить эффективный алгоритм поиска максимального паросочетания в произвольном графе G.

20.2.3. При доказательстве формулы Татта-Бержа нам понадобится следующий результат.

Лемма 20.7. *Для любого подмножества S множества $V(G)$ вершин графа G четность числа $c_{\mathrm{odd}}(G-S)-|S|$ совпадает с четностью количества $n=|V(G)|$ вершин в графе G:*

$$c_{\mathrm{odd}}(G-S)-|S|\equiv n\pmod 2.$$

Доказательство. Рассмотрим произвольное подмножество S множества $V(G)$ вершин графа G. Любое такое подмножество разбивает все множество $V(G)$ вершин на $c_{\mathrm{odd}}(G-S)$ блоков V_i, отвечающих нечетным компонентам графа $G-S$, $c_{\mathrm{even}}(G-S)$ блоков U_j, соответствующих четным компонентам графа $G-S$, а также на блок, состоящий из вершин самого подмножества S:

$$V(G)=V_1\cup V_2\cup\ldots\cup V_{c_{\mathrm{odd}}(G-S)}\cup U_1\cup U_2\cup\ldots\cup U_{c_{\mathrm{even}}(G-S)}\cup S.$$

Так, для изображенного на рисунке 103 графа G и подмножества S мы имеем $c_{\mathrm{odd}}(G-S)=2$ нечетных блока и $c_{\mathrm{even}}(G-S)=2$ четных блока.

Количество вершин во всех блоках U_j четно, поэтому в результате удаления из графа G этих вершин мы получим количество вершин той же четности, что и n. Возьмем теперь в каждом нечетном блоке все вершины, кроме одной, и удалим эти вершины. Так как число удаляемых вершин в каждом таком блоке четно, то четность оставшегося количества вершин по-прежнему будет совпадать с четностью n.

Заметим теперь, что у нас осталось по одной вершине из каждой нечетной компоненты (вершины, помеченные синим цветом на рис.103), а также все вершины подмножества S (зеленые вершины на рис.103). Следовательно, общее количество оставшихся вершин равно $c_{\mathrm{odd}}(G-S)+$ $+|S|$, и это число совпадает по четности с количеством n всех вершин:

$$c_{\mathrm{odd}}(G-S)+|S|\equiv n\pmod 2.$$

Вычтем теперь из числа $c_{\mathrm{odd}}(G-S)+|S|$ четное число, равное удвоенному количеству вершин подмножества S. Полученное в результате число $c_{\mathrm{odd}}(G-S)-|S|$ будет иметь ту же четность,

что и число $c_{\text{odd}}(G-S)+|S|$, а следовательно, что и n:

$$c_{\text{odd}}(G-S)-|S| \equiv n \pmod 2.$$

Лемма доказана.

20.2.4. Приступим к доказательству теоремы Бержа. С учетом леммы 20.6 нам достаточно убедиться в справедливости обратного неравенства

$$\alpha'(G) \geqslant \frac{1}{2}\min_{S\subset V(G)}\{n-(c_{\text{odd}}(G-S)-|S|)\} = \frac{n}{2}-\frac{1}{2}\max_{S\subset V(G)}\{c_{\text{odd}}(G-S)-|S|\}.$$

Следствием леммы 20.7 является тот факт, что четность числа

$$m = \max_{S\subset V(G)}[c_{\text{odd}}(G-S)-|S|] = c_{\text{odd}}(G-B)-|B|$$

совпадает с четностью количества n вершин в графе. Кроме того, величина m всегда больше или равна нуля. Действительно, выбирая в качестве подмножества S пустое подмножество, мы получаем неравенство $c_{\text{odd}}(G)\geqslant 0$, а значит, $m\geqslant c_{\text{odd}}(G)\geqslant 0$.

Добавим тогда к исходному графу G подмножество W, состоящего из m новых вершин, и соединим каждую вершину подмножества W со всеми остальными вершинами — как новыми, так и старыми (рис.105). В полученном таким образом графе $\widetilde{G}$ имеется достаточно большое количество ребер, поэтому есть шанс, что в нем существует совершенное паросочетание. Покажем, что это действительно так.

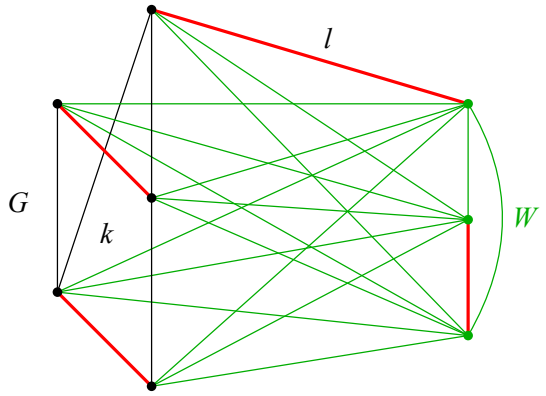


Рис. 105

Заметим, прежде всего, что необходимое условие существование совершенного паросочетания в графе $\widetilde{G}$ выполняется — так как четность чисел m и $n=|V(G)|$ совпадает, то количество $|V(\widetilde{G})|=n+m$ вершин в новом графе $\widetilde{G}$ четно. Иными словами, для подмножества $T=\varnothing$ условие Татта (54) в графе $\widetilde{G}$ выполняется. Покажем, что эти условия выполняются и для любого подмножества $T\neq\varnothing$.

Если при удалении подмножества T вершин останется хотя бы одна вершина из множества W, то граф $\widetilde{G}-T$ останется связным, и условие (54) выполнено:

$$c_{\text{odd}}(\widetilde{G}-T)\leqslant 1\leqslant |T|.$$

В противном случае, то есть в случае, когда $T=W\cup S$, $S\subset V(G)$, мы по построению графа $\widetilde{G}$ получаем, что $\widetilde{G}-T=G-S$, и поэтому

$$c_{\text{odd}}(\widetilde{G}-T)=c_{\text{odd}}(G-S).$$

Но

$$c_{\text{odd}}(G-S)-|S| \leqslant c_{\text{odd}}(G-B)-|B| = m,$$

поэтому

$$c_{\text{odd}}(\widetilde{G}-T) = c_{\text{odd}}(G-S) \leqslant m+|S| = |W \cup S| = |T|.$$

20.2.5. Итак, для графа $\widetilde{G}$ условия Татта (54) выполняются, то есть в таком графе существует совершенное паросочетание $\widetilde{M}$ (ребра, окрашенные в красный цвет на рис.105). Обозначим через l количество ребер в паросочетании $\widetilde{M}$, исходящих из W в G (см.рис.105). Так как в W содержится m вершин, то $l \leqslant m$.

Пусть теперь k есть количество ребер в $\widetilde{M}$, оба конца которых лежат в G (рис.105). Ясно, что

$$2k = n - l.$$

Так как эти k ребер образуют какое-то паросочетание в исходном графе G, то $k \leqslant |M|$, где M — максимальное паросочетание в графе G, $|M| = \alpha'(G)$. Следовательно,

$$2\alpha'(G) \geqslant 2k = n - l \geqslant n - m.$$

Теорема Бержа доказана. □

Раскраска графов

21 k-раскрашиваемые графы. Теорема Брукса

21.1. Как мы уже неоднократно упоминали выше, любой двудольный граф мы можем рассматривать как двураскрашиваемый граф, то есть граф, вершины которого могут быть правильно окрашены в два цвета. Логично предположить, что наряду с двураскрашиваемыми должны существовать и графы более общего вида, а именно, k-раскрашиваемые графы. К изучению таких графов мы сейчас и перейдем.

21.1.1. Под раскраской вершин графа G понимается разбиение множества V его вершин на блоки, называемые цветами. Задать такое разбиение можно, например, с помощью функции $\varphi : V \to C$, отображающей множество V вершин на некоторое множество $C = \{1, \ldots, k\}$, называемое множеством цветов.

Определение 21.1. Раскраска вершин простого графа G называется *правильной,* если любые две смежные вершины графа окрашены в разные цвета. Любой граф, который допускает правильную раскраску своих вершин в k цветов, называется *k-раскрашиваемым графом.*

Замечание 21.2. При анализе k-раскрашиваемых графов нам достаточно ограничиться простыми графами. Действительно, любая петля соединяет вершину саму с собой. Такую вершину мы одновременно в разные цвета раскрасить, конечно же, не сможем. Как следствие, графы с петлями правильно раскрасить невозможно в принципе, так что мы их сразу можем исключить из рассмотрения. Далее, нас, как правило, будет интересовать вопрос, соединены ли у нас какие-то две вершины ребром или нет. При этом нам совершенно не принципиально, соединены ли эти вершины одним или несколькими ребрами. Так что любое мультиребро мы можем заменить на единственное ребро и рассматривать, таким образом, только лишь простые графы.

21.1.2. Рассмотрим граф G, построенный на n вершинах. Мы можем взять n цветов и каждую из вершин графа G окрасить в свой цвет. При этом мы, конечно же, получим правильную раскраску вершин графа. Однако такой способ окраски вершин нам не очень интересен — на практике нас, как правило, интересует *минимальное* количество цветов, в которые мы можем правильно раскрасить вершины графа G.

Определение 21.3. Наименьшее количество k цветов, в которое можно правильно покрасить вершины графа G, называется *хроматическим числом* $\chi(G)$ этого графа. Сам граф при этом часто называют *k-хроматическим.*

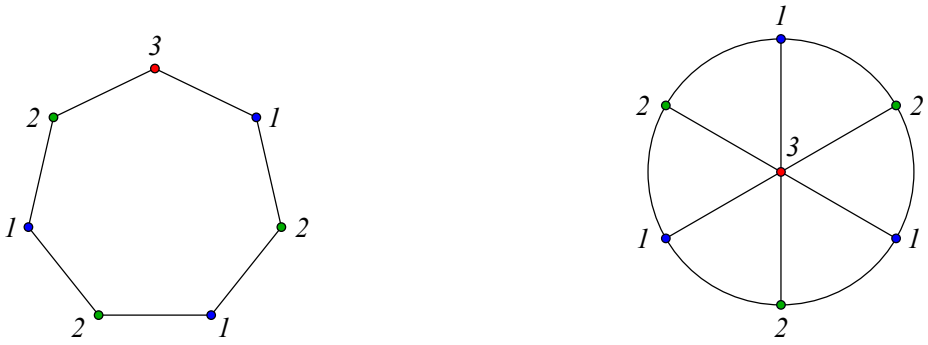


Рис. 106: 3-хроматические графы

21.1.3. Понятно, что простой граф G является 1-раскрашиваемым тогда и только тогда, когда G представляет собой пустой граф $\bar{K}_n$, и 2-раскрашиваемым тогда и только тогда, когда он двудольный. Любой простой цикл нечетной длины является простейшим примером 3-хроматического графа (смотри рис.106,a). Еще одним простым примером 3-раскрашиваемого графа является граф “колесо” (рис.106,b).

Как мы уже заметили выше, хроматическое число любого графа, построенного на n вершинах, ограничено сверху значением n. Эта верхняя граница достигается, например, на полном графе. Действительно, так как любые две его вершины соединены ребром, то никакие две вершины K_n нельзя окрасить в один и тот же цвет. Следовательно, хроматическое число полного графа $\chi(K_n) = n$.

21.1.4. Задачи, связанные с правильной окраской вершин графа G в как можно меньшее количество цветов, достаточно часто встречаются на практике. Приведем несколько характерных примеров такого рода задач [**?**, **?**].

Пример 21.4. Предположим, что студенты в некотором университете учатся по индивидуальным программам и сдают в конце года экзамены по всем предметам, которые они изучали в течение года. Учебный отдел должен так составить расписание экзаменов, чтобы экзамены, на которые должен прийти один и тот же студент, стояли в разные дни. При этом расписание хочется составить так, чтобы количество экзаменационных дней было бы минимальным.

Для формализации данной задачи рассмотрим граф G, множество вершин которого совпадает с множеством читаемых в университете курсов. Соединим две вершины ребром в случае, если хотя бы один студент слушает оба эти курса. Тогда любая правильная раскраска вершин графа G даст нам бесконфликтное расписание, а хроматическое число графа G определит нам минимальное количество экзаменационных дней.

Пример 21.5. Химическая компания производит набор $X = \{x_1, \ldots, x_n\}$ химикатов. Некоторые пары этих химикатов взрываются, если приходят в контакт друг с другом. В качестве меры предосторожности компания делит свой склад на отсеки, помещая в каждый отсек лишь те препараты, которые не взрываются при контакте друг с другом. Задача состоит в нахождении минимального количества отсеков для данного набора химических веществ.

Для решения данной задачи построим граф G на n вершинах, помеченных элементами множества X. Соединим любые две вершины графа ребром в случае, если соответствующие этим вершинам химикаты взрываются при контакте друг с другом. Тогда минимальное количество отсеков, на которые следует разделить склад, совпадает с хроматическим числом $\chi(G)$ графа.

21.2. Любую задачу, связанную с раскраской вершин графа, можно разбить на две подзадачи. Первая подзадача состоит в проверке данного графа G на k-раскрашиваемость, вторая — в определении хроматического числа $\chi(G)$ графа G. Мы знаем, что в случае $k = 2$ существует достаточно простой критерий двураскрашиваемости графа, на основе которого мы можем построить простой алгоритм проверки графа G на двудольность. К сожалению, для любого $k > 2$ никаких простых и удобных критериев проверки графа на k-раскрашиваемость не существует. Говоря формальным языком, задача проверки графа G на k-раскрашиваемость является NP-полной задачей. Как следствие, более сложная задача определения хроматического числа графа NP-сложна. В частности, к настоящему времени не известно никакого алгоритма, работающего за полиномиальное время и позволяющего для произвольно взятого графа определить его хроматическое число. В этой связи на практике довольствуются обычно какими-то эвристическими алгоритмами, позволяющими более или менее эффективно определить верхнюю границу хроматического числа $\chi(G)$. Опишем наиболее очевидный и популярный из них — так называемый жадный алгоритм окраски вершин графа.

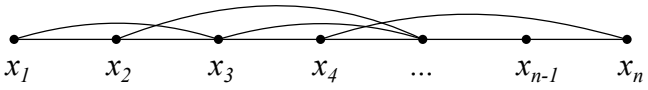


Рис. 107

21.2.1. Линейно упорядочим вершины простого связного графа G (рис.107). Мы знаем, что любой граф на n вершинах мы можем всегда правильно окрасить цветами из множества $Y = \{1, \ldots, n\}$. Возьмем это множество цветов и начнем окрашивать вершины графа следующим образом (рис.108):

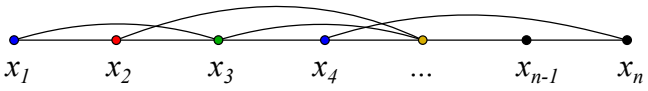


Рис. 108

– окрасим вершину x_1 в цвет 1;
– рассмотрим вторую вершину — вершину x_2; если она смежна с вершиной x_1, то окрасим ее в цвет 2; в противном случае вновь окрасим ее в цвет 1;
– выберем теперь третью вершину x_3; если она смежна с вершинами x_1 и x_2, и если вершины x_1 и x_2 смежны между собой, то x_3 мы сможем окрасить только в новый цвет 3; если она смежна с вершинами x_1 и x_2, а сами эти вершины не смежны, то мы окрасим ее в цвет 2; если она не смежна ни с одной из вершин x_1 и x_2, то мы можем ее окрасить в цвет 1; наконец, если она смежна лишь с одной из двух вершин x_1 и x_2, то мы можем окрасить x_3 в цвет той из вершин x_1 и x_2, которая с x_3 не смежна;

– опишем теперь общий, k-й шаг алгоритма: рассмотрим вершину x_k, а также все смежные с ней вершины с меньшими индексами; эти вершины как-то нами окрашены на предыдущих шагах алгоритма; исключим тогда из множества Y цветов все те цвета, которые были использованы при окраске смежных с x_k вершин с меньшими индексами; в оставшемся множестве цветов выберем минимальный цвет и окрасим этим цветом вершину x_k.

21.2.2. Описанный нами жадный алгоритм далеко не всегда работает оптимально и может очень сильно зависеть от способа линейного упорядочивания вершин. Несмотря на это, максимальное количество использованных в данном алгоритме цветов никогда не превысит величины $\Delta + 1$, где Δ — максимальная из степеней вершин графа G.

Действительно, в жадном алгоритме для любой вершины x_i количество уже использованных цветов, в которые нельзя окрасить вершину x_i, никогда не превысит величины $\deg(x_i) \leqslant \Delta$. Худший случай с точки зрения окраски вершин наступит у нас в том случае, когда мы встретили вершину x степени Δ, и у этой вершины все Δ смежных с ней вершин оказались окрашенными на предыдущих шагах нашего алгоритма. Тогда вершину x мы будем вынуждены окрасить в новый цвет $\Delta+1$. Во всех остальных случаях количество использованных цветов будет меньше величины $\Delta + 1$.

Проведенные рассуждения доказывают, в частности, следующий важный результат.

Теорема 21.6. *Хроматическое число $\chi(G)$ графа G ограничено сверху величиной $\Delta + 1$, то есть*

$$\chi(G) \leqslant \Delta + 1, \tag{58}$$

где Δ — наибольшая из степеней вершин графа G.

Следствие 21.7. *Любой k-хроматический граф обязательно содержит по-крайней мере одну вершину, степень которой больше или равна $k - 1$.*

21.3. Проанализируем верхнюю оценку (58) на хроматическое число $\chi(G)$ графа G.

21.3.1. Заметим, прежде всего, что эта оценка может быть сколь угодно далека от реального значения хроматического числа $\chi(G)$ графа. В качестве характерного примера рассмотрим произвольное дерево T. Так как любое дерево представляет собой двудольный граф, то хроматическое число $\chi(T)$ дерева T равно двум. Однако в дереве могут быть вершины, степени которых сколь угодно велики. Как следствие, число $\Delta + 1$ может быть сколь угодно большим, то есть для таких объектов оценка (58) может быть сколь угодно далека от реальности.

21.3.2. В связи со сделанным выше наблюдением естественным образом возникает следующий вопрос — а можем ли мы как-то улучшить данную оценку? Оказывается, однако, что существуют целые классы графов, для которых эта оценка неулучшаема.

Первый такой класс графов — это полные графы K_n. Мы знаем, что у таких графов $\Delta = n-1$, а окрасить эти графы меньше, чем в n цветов, невозможно. Следовательно, для полных графов

$$\chi(K_n) = n = \Delta(K_n) + 1.$$

Ко второму классу графов относятся простые циклы C_{2n+1} нечетной длины (рис.106,а). Для таких циклов $\Delta = 2$, а хроматическое число равно трем. Следовательно, и для таких графов верхняя оценка (58) достигается.

Во всех остальных случаях эту оценку можно уменьшить, но только лишь на единицу. Именно, справедлива следующая

Теорема 21.8 (Brooks, 1941)**.** *Пусть G есть простой связный граф, не являющийся полным графом или же простым нечетным циклом. Тогда*

$$\chi(G) \leqslant \Delta, \qquad \textit{где} \qquad \Delta = \max_{x_i \in V(G)} \deg(x_i).$$

Рис. 109

21.3.3. Мы приведем доказательство этой теоремы, данное Ласло Ловасом [**?**]. Основная идея этого доказательства состоит в следующем: нам нужно придумать такой способ нумерации вершин графа, при котором в результате запуска жадного алгоритма на выходе получается граф, вершины которого оказываются окрашенными не более чем в Δ цветов. Заметим сразу же, что при произвольной нумерации вершин жадный алгоритм может окрасить вершины графа более, чем в Δ цветов. В качестве примера рассмотрим двудольный граф, показанный на рис.109,a. Максимальная степень Δ его вершин равна двум. Если же мы пронумеруем вершины графа так, как это указано на рис.109,b, то мы получим окраску вершин этого графа в три цвета.

21.3.4. Легче всего подобную нумерацию вершин придумать в графе G, не являющимся k-регулярным. Действительно, возьмем в графе G произвольную вершину, степень которой строго меньше Δ, и присвоим ей номер n (см.рис.110). Так как граф G связный, то мы, например, поиском в ширину, можем построить остовное дерево T с корнем в вершине n (ребра, помеченные синим цветом на рис.110), назначая вершинам этого дерева номера i, убывающие по мере удаления этих вершин от корня дерева.

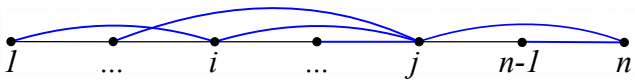
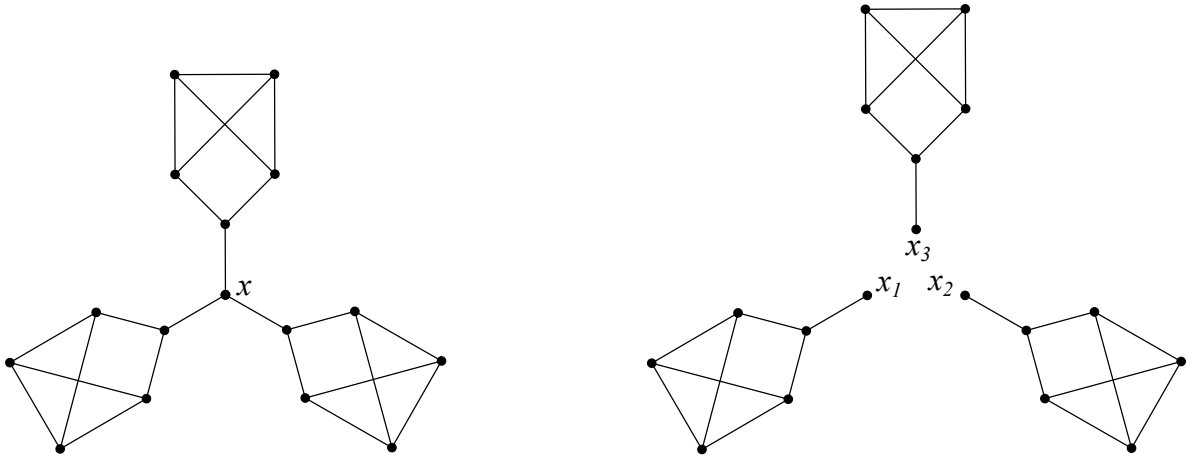


Рис. 110

В этом случае любая вершина i, отличная от n, обязательно будет иметь по крайней мере одну смежную с ней в остовном дереве T вершину, номер j который строго больше i. Как следствие, при подходе жадного алгоритма к вершине $i \neq n$ хотя бы одна смежная с i вершина, а именно, вершина j, обязательно останется неокрашенной. Оставшиеся же смежные с i вершины будут окрашены не более чем в $\deg(i) - 1 \leqslant \Delta - 1$ цветов. Поэтому жадный алгоритм всегда сможет назначить вершине $i \neq n$ один из цветов $1, 2, \ldots, \Delta$.

Наконец, так как степень вершины $n < \Delta$, то мы всегда сможем на последнем шаге жадного алгоритма назначить этой вершине хотя бы один неиспользованный цвет из списка $1, 2, \ldots, \Delta$.

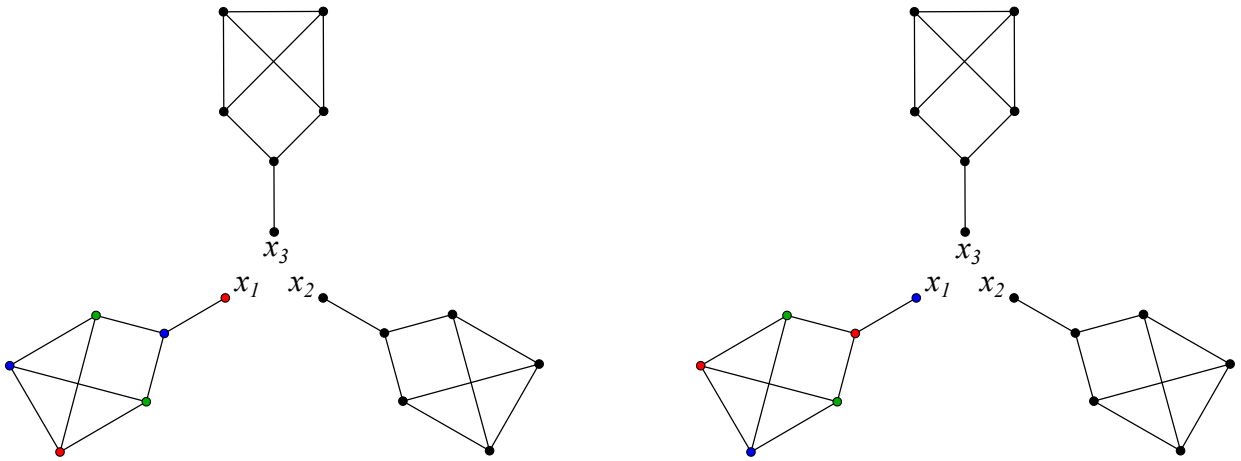
21.3.5. Рассмотрим теперь Δ-регулярный односвязный граф, имеющий хотя бы одну точку сочленения x (рис.111,a). Обозначим через H_i подграфы графа G, полученные в результате удаления вершины x. Расщепим вершину x на Δ вершин x_i и рассмотрим графы G_i, индуцированные



(a) (b)

Рис. 111

подмножествами вершин $\{x_i\} \cup V(H_i)$ (рис.111,b). В каждом таком графе G_i, рассматриваемом отдельно от остальных, степень вершины x_i равна единице. Так как у нас $\Delta > 1$, то любой такой граф G_i можно правильно окрасить в не более чем Δ цветов, используя алгоритм, описанный в предыдущем пункте для нерегулярных графов. Покажем теперь, что данную окраску всегда можно продолжить на весь граф G, окрасив правильно все вершины G не более чем в Δ цветов.



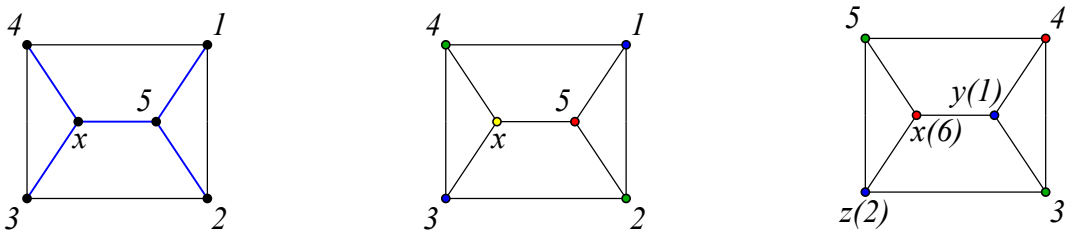
(a) (b)

Рис. 112

Предположим, что в результате работы жадного алгоритма вершина x_i в графе G_i окрасилась в цвет k (красный цвет на рис.112,a). Рассмотрим все вершины данного графа, окрашенные в цвет k, а также все вершины этого блока, окрашенные в цвет 1 (синий цвет на рис.112,a). Поменяем цвета этих вершин на противоположные (рис.112,b). Правильность окраски при этом, очевидно, не нарушится. В результате такой замены вершина x_i окажется окрашенной в цвет 1. Повторяя эту процедуру для каждого из графов G_i, мы, как говорят, согласуем окраску вершин x_i — для любого i вершины x_i окажутся окрашенными в один и тот же цвет 1. Склеивая теперь обратно Δ вершин x_i в одну вершину x, мы получим правильную окраску исходного графа G не более чем в Δ цветов.

21.3.6. Осталось рассмотреть случай простого связного Δ-регулярного графа G без точек со-

членения, то есть Δ-регулярного двусвязного графа. Мы можем считать, что $\Delta \geqslant 3$, так как случаи $\Delta = 0$ и $\Delta = 1$ отвечают элементарным полным графам K_1 и K_2 соответственно, а случай $\Delta = 2$ соответствует либо нечетному циклу, который мы также, как и полный граф, исключили из рассмотрения, либо четному циклу, для которого теорема верна.



(a) (b) (c)

Рис. 113

Выбирая в рассматриваемом графе G в качестве корня остовного дерева произвольную вершину x (рис.113,a), присваивая ей номер n и размечая оставшиеся $(n-1)$ вершин с помощью поиска в ширину, мы получим некоторое линейно упорядоченное множество вершин графа G. Как и в случае нерегулярного графа, запуская жадный алгоритм на такой разметке вершин, мы гарантируем, что для окраски произвольной вершины $i \neq n$ нам всегда хватит Δ цветов. При окраске же последней вершины, то есть вершины с номером n, у нас могут возникнуть проблемы (см.рис.113,b). Действительно, у такой вершины все Δ смежных с ней вершин уже оказываются окрашенными на предыдущих шагах жадного алгоритма. Если при этом все Δ смежных с n вершин окажутся к этому моменту окрашенными в разные цвета, то жадный алгоритм окрасит вершину с номером n в цвет $\Delta+1$. Иными словами, нам нужно подправить алгоритм разметки вершин так, чтобы подобная ситуация никогда не возникла.

Оказывается, сделать это достаточно просто. Ниже мы покажем, что в любом двусвязном k-регулярном графе обязательно найдется вершина x, имеющая пару соседних несмежных между собой вершин y и z, таких, что граф $G-y-z$ остается связным (см. рис.113,c). Назначим такой вершине x номер n, а вершинам y и z — номера 1 и 2. Удаляя вершины y и z, мы получаем связный граф, вершины которого можно разметить с помощью поиска в ширину, стартуя из вершины $x = n$ и назначая вершинам метки i с номерами, убывающими по мере удаления этих вершин от корня дерева. Добавляя затем обратно вершины y и z, мы получаем граф G, в котором любая вершина, отличная от x, по-прежнему имеет как максимум $\Delta - 1$ смежных с ней вершин, имеющих меньшие номера.

Запустив тогда жадный алгоритм для вершин, упорядоченных в порядке $(1, 2, \ldots, n)$, мы окрасим вершины графа в не более чем Δ цветов. Действительно, вершины 1 и 2 алгоритм окрасит в цвет 1 (синий цвет на рис.113,c). Так как любая вершина i, $i = 3, \ldots, (n-1)$, обязательно имеет по крайней мере одну смежную вершину j с номером, большим, чем i, на своем пути к корню n остовного дерева, то жадному алгоритму хватит для ее окраски цветов из множества $\{1, \ldots, \Delta\}$. Наконец, среди Δ смежных с n вершин найдутся как минимум две вершины, окрашенные в один и тот же цвет, а именно, вершины 1 и 2. Поэтому для правильной окраски n также окажется достаточно цветов из множества $\{1, 2, \ldots, \Delta\}$.

21.3.7. Итак, для полного доказательства теоремы Брукса осталось убедиться в том, что описанная в предыдущем пункте тройка вершин x, y, z в двусвязном Δ-регулярном графе с $\Delta \geqslant 3$ всегда существует. Легче всего сделать это в случае, когда граф трехсвязен. Так как G отличен от полного графа, то в нем найдется пара несмежных между собой вершин y, z, смежных с

некоторой общей для них вершиной x. Так как граф трехсвязный, то удаление y и z оставляет граф связным, так что в этом случае искомая тройка вершин существует.

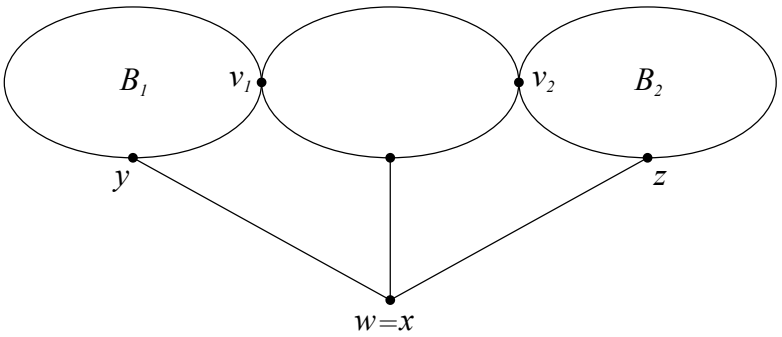


Рис. 114

Теперь предположим, что $\kappa(G) = 2$. Выберем в G вершинно разделяющее множество S, $|S| = 2$, и рассмотрим вершину $x \in S$. Так как $\kappa(G - x) = 1$, то граф $G - x$ можно разбить на блоки B_j и точки сочленения v_j (смотри рис.114). Рассмотрим произвольную пару блоков B_1 и B_2, отвечающих листьям дерева блоков и точек сочленения графа $G - x$. Обозначим через v_1 и v_2 принадлежащие этим двум блокам точки сочленения. Так как исходный граф G двусвязный, то существует вершина $y \in (B_1 - v_1)$, смежная с x в графе $G - v_1$, а также вершина $z \in (B_2 - v_2)$, смежная с x в графе $G - v_2$. По построению, вершины y и z несмежны. Кроме того, степень Δ в графе больше или равна 3, так что вершина x, помимо y и z, имеет хотя бы одного соседа в $G - y - z$. Как следствие, граф $G - y - z$ остается связным, а значит, вершины x, y, z, образуют искомую тройку вершин в графе G. Теорема Брукса доказана.

22 Нижние оценки на хроматическое число

22.1. Помимо верхних оценок, для хроматического числа $\chi(G)$ можно пытаться строить и какие-то нижние оценки.

22.1.1. Одна из наиболее очевидных нижних оценок связана с понятием кликового числа $\omega(G)$ графа G, под которым понимается количество вершин в наибольшей клике графа G. Очевидным является следующее утверждение.

Утверждение 22.1. *Хроматическое число $\chi(G)$ любого графа G не может быть меньше его кликового числа:*

$$\chi(G) \geqslant \omega(G). \tag{59}$$

Отсюда, в частности, следует простой способ построения графа G с большим хроматическим числом — поместить в граф G клику большого размера. Оказывается, однако, что граф с большим хроматическим числом вовсе не обязан иметь одновременно и большое кликовое число.

22.1.2. Рассмотрим, к примеру, так называемые графы без треугольников, то есть графы, не содержащие простых циклов длины три. Заметим, что любой полный граф K_n, построенный на $n \geqslant 3$ вершинах, содержит в качестве своих подграфов клики K_k любых размеров $k \in [1, \ldots, n-1]$, и, в частности, обязательно содержит треугольники K_3. Следовательно, графы без треугольников — это графы с кликовым числом $\omega(G) \leqslant 2$, то есть графы, которые в качестве своих подграфов не содержат никаких клик помимо ребер K_2 и вершин K_1.

Теорема 22.2 (Mycielski, 1955)**.** *Для любого натурального числа k существует k-хроматический граф без треугольников.*

Доказательство данной теоремы основано на следующей конструкции, позволяющей получить из произвольного простого графа G_k некоторый простой граф G_{k+1}, содержащий G_k в качестве своего подграфа:

- добавляем к множеству $V(G_k) = \{x_1, \ldots, x_n\}$ вершин графа G_k вершины $Y = \{y_1, \ldots, y_n\}$, а также еще одну вершину z;
- для любого $i = 1, \ldots, n$ соединяем ребрами вершину y_i со всеми вершинами исходного графа G_k, смежными с x_i;
- соединяем вершину z со всеми вершинами множества Y.

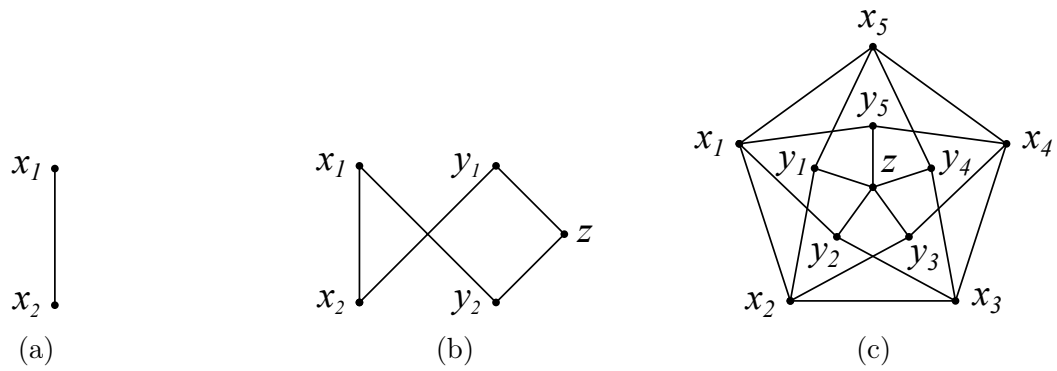


Рис. 115: Графы без треугольников

Пример 22.3. Рассмотрим в качестве примера 2-хроматический граф $G_2 = K_2$ (рис.115,a). Первая итерация описанной выше конструкции получает получить из него 3-хроматический граф $G_3 = C_5$ (рис.115,b). Следующая итерация, примененная к графу C_5, дает нам так называемый граф Грётцша G_4 (рис.115,c).

22.1.3. Покажем по индукции, что граф Грётцша G_4, а также любые графы G_{k+1}, полученные в результате применения данной процедуры к графам G_k, являются $(k+1)$-хроматическими графами без треугольников.

Заметим прежде всего, что граф G_{k+1} не содержит никаких треугольников. Действительно, так как все вершины множества Y несмежны друг с другом, то любой потенциальный треугольник в графе G' может содержать лишь одну вершину из множества Y. Как следствие, вершину z такой треугольник содержать уже не может. Поэтому единственный возможный вариант такого треугольника — это простой цикл вида $x_i y_j x_k x_i$. Однако и это невозможно — в противном случае в исходном графе G_k существовал бы треугольник $x_i x_j x_k x_i$, чего быть не может по индукционному предположению.

Теперь покажем, что граф G_{k+1} является $(k+1)$-раскрашиваемым. Для этого рассмотрим произвольную правильную окраску графа G_k в k цветов и продолжим ее на граф G_{k+1} следующим образом: окрасим вершины y_i графа G_{k+1} в те же цвета, что и вершины x_i, а вершину z окрасим в цвет $(k+1)$. В результате получим правильную окраску графа G_{k+1}.

Осталось доказать, что граф G_{k+1} не является k-раскрашиваемым. Предположим, что граф G_{k+1} все же можно правильно окрасить в k цветов. Без потери общности мы можем считать, что вершина z окрашена в цвет k. При таком способе окраски никакая вершина множества

Y не может быть окрашена в этот же цвет k, то есть все вершины y_j окрашены в цвета из подмножества $\{1, \ldots, k-1\}$. Вершины же x_i могут быть, в принципе, окрашены в правильной окраске графа G_{k+1} в цвета $1, \ldots, k$. Обозначим через S подмножество вершин x_i, окрашенных в цвет k. Возьмем тогда каждую из вершин x_i, окрашенную в цвет k, и перекрасим ее в цвет, в который окрашена соответствующая ей вершина y_i. Так как множество смежных с x_i вершин совпадает для любого i с множеством вершин, смежных с y_i, то такой способ окраски графа G_k окажется правильным (при этом может испортиться правильная окраска графа G_{k+1}, однако для нас сейчас это не важно). Однако этот способ окраски требует лишь $(k-1)$ цветов графа G_k, что противоречит индукционному предположению. □

22.1.4. Итак, мы установили, что отсутствие в графе G треугольников (а следовательно, и клик большего размера) еще не гарантирует того, что хроматическое число $\chi(G)$ будет маленьким. Посмотрим еще раз на контрпримеры, построенные Мицельским. В каждом из графов G_k треугольники (то есть клики, изоморфные K_3) отсутствуют. Заметим, однако, что в этих графах довольно много индуцированных подграфов, изоморфных циклам C_l, $l \geqslant 4$. Иными словами, обхват графа G_k мал. В связи с этим может показаться, что большое значение $\chi(G_k)$ вызвано наличием в них достаточно большого количества циклов не слишком большой длины. Однако и это предположение оказывается неверным. Именно, справедлива

Теорема 22.4 (Erdos, 1961)**.** *Для любого натурального k существует k-хроматический граф, обхват которого (то есть длина наименьшего простого цикла в нем) больше или равен k.*

Эрдёш доказал эту теорему с помощью вероятностных методов. Его доказательство было неконструктивным: он лишь показал, что подобные графы существуют. Несколько лет спустя появились работы, в которых были предложены и конструктивные алгоритмы построения подобного рода графов.

22.1.5. Давайте теперь поймем, в чем же заключается проблема с графами, в которых отсутствуют треугольники, и в которых обхват графа велик. Локально (в небольшой окрестности заданной вершины x) такие графы очень похожи на деревья. Мы знаем, что хроматическое число дерева равно двум. Поэтому и для графов с большим обхватом можно было бы ожидать, что хроматическое число графа окажется небольшим. Однако, как показали Мицельский и Эрдёш, это предположение неверно — хроматическое число таких графов может быть сколь угодно велико. Иными словами, такая характеристика графа, как его хроматическое число, является глобальной характеристикой графа G, и давать какие-то оценки для $\chi(G)$, исходя только из локальных характеристик графа, практически невозможно.