



Рис. 1.1. Пример использования алгоритма сортировки вставками (шкала времени направлена вниз)

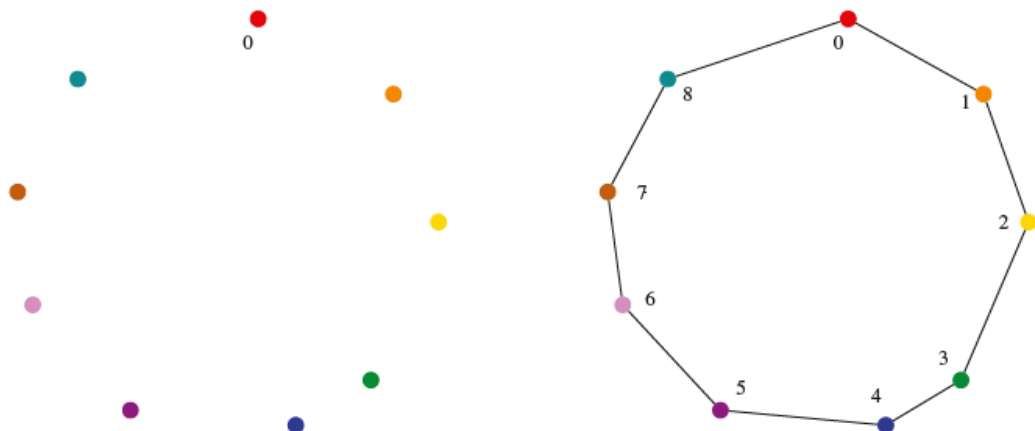


Рис. ЦВ-1.2. Случай удачного входного экземпляра для эвристического алгоритма ближайшего соседа. Цветная разметка точек, соответствующая порядку следования цветов радуги (от красного к фиолетовому), отражает порядок перемещения манипулятора

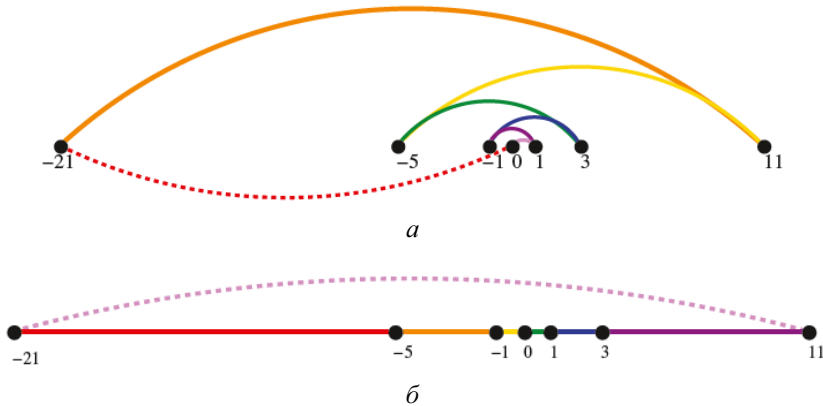


Рис. ЦВ-1.3. Пример неудачного входного экземпляра для эвристического алгоритма ближайшего соседа (*а*) и оптимальное решение (*б*). Цвета последовательности перемещений манипулятора соответствуют порядку цветов спектра

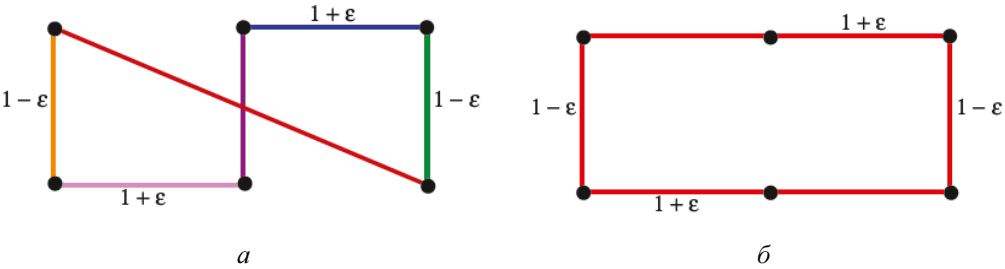


Рис. 1.4. Неудачный входной экземпляр для алгоритма ближайших пар (*а*) и оптимальное решение (*б*)

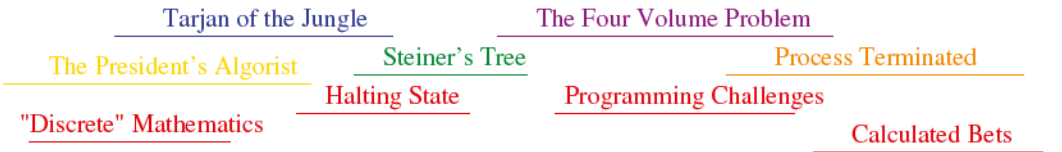


Рис. 1.5. Экземпляр задачи планирования непересекающихся календарных периодов. Оптимальное решение представлено четырьмя красными названиями

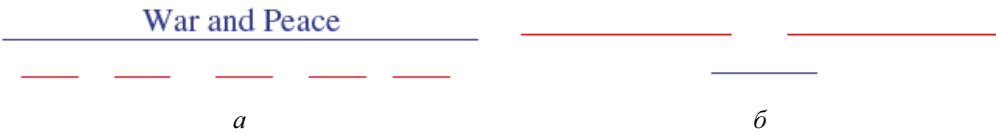


Рис. 1.6. Неудачные экземпляры задач для применения эвристики: *a* — самые ранние периоды первые; *б* — самые короткие периоды первые. Оптимальное решение представлено красным цветом



Рис. 1.7. Контрпример для «жадного» эвристического алгоритма календарного планирования. Выбор фиолетового интервала, который перекрывает наименьшее количество других интервалов, блокирует оптимальное решение (четыре красных интервала)



Рис. 1.8. Оптимальное решение (красные прямоугольники) до и после внесения изменений (обозначены пунктиром) в экземпляр задачи

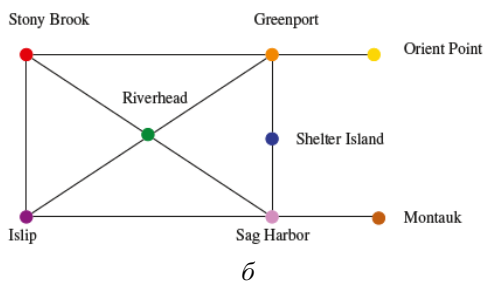
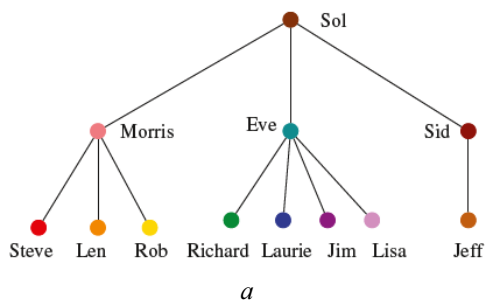
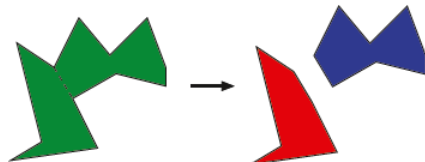


Рис. 1.9. Моделирование реальных структур с помощью деревьев (*a*) и графов (*b*)

$\{4,1,5,2,3\} \xrightarrow{4-} 4 \{1,4,2,3\}$

$\{1,2,7,9\} \xrightarrow{9-} 9 \{1,2,7\}$



$\text{ALGORITHM} \rightarrow \text{A} | \text{LGORITHM}$

a

b

Рис. 1.10. Рекурсивное разложение комбинаторных объектов: *a* — перестановки, подмножества, деревья, графы; *b* — множества точек, многоугольники и строки. Обратите внимание на то, что после удаления одного элемента из перестановки $\{1, \dots, n\}$ оставшиеся элементы перенумеровываются, чтобы сохранить перестановку $\{1, \dots, n-1\}$ элементов

Билеты	Выигрышные пары		
1 2 3	1 2	2 3	3 4
1 4 5	1 3	2 4	3 5
2 4 5	1 4	2 5	4 5
3 4 5	1 5		

Рис. ЦВ-1.11. Покрытие всех пар множества $\{1, 2, 3, 4, 5\}$ номерами билетов $\{1, 2, 3\}$, $\{1, 4, 5\}$, $\{2, 4, 5\}$, $\{3, 4, 5\}$. Цвет пары представляет покрывающий билет

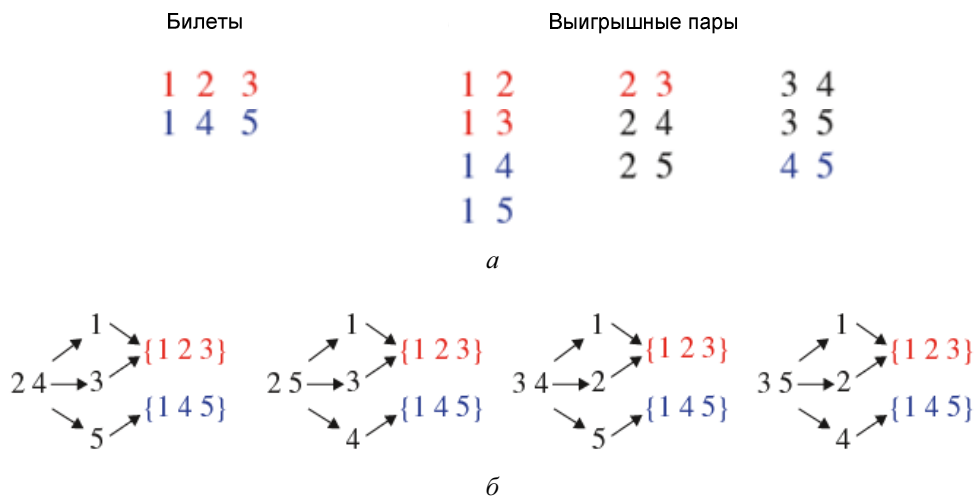


Рис. ЦВ-1.12. *a* — гарантия выигрышной комбинации из двух номеров множества $\{1, 2, 3, 4, 5\}$ при использовании только комбинации номеров $\{1, 2, 3\}$ и $\{1, 4, 5\}$;

б — все отсутствующие пары подразумевают пару, входящую в покрытие при разложении.
Цвет пары представляет покрывающий билет

ГЛАВА 2

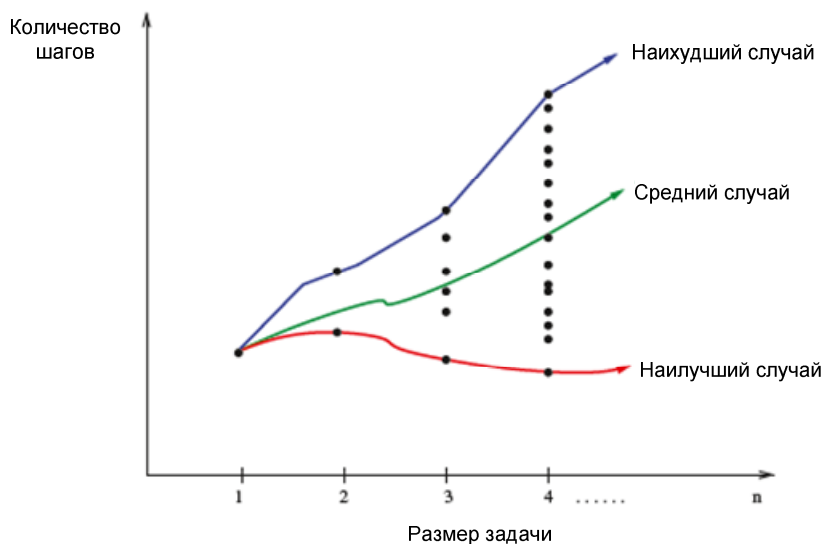


Рис. 2.1. Наилучший, наихудший и средний случаи сложности алгоритма

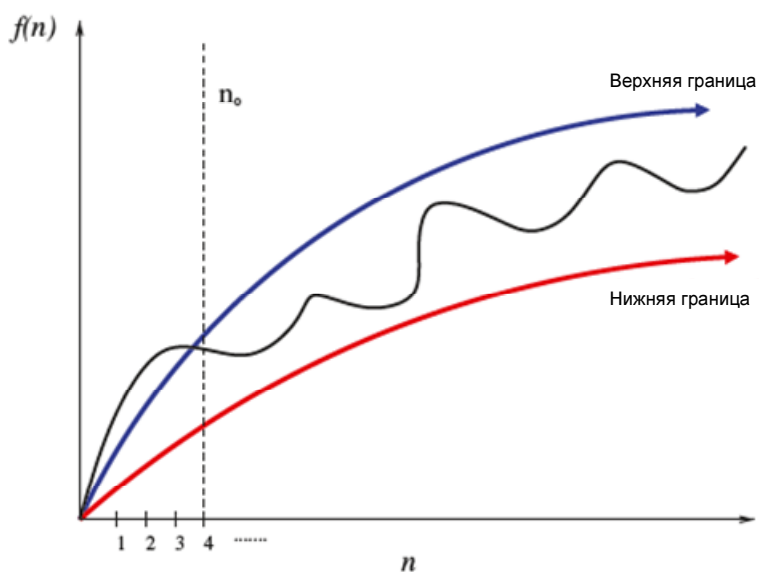


Рис. 2.2. Верхняя и нижняя границы, действенные для $n > n_0$, сглаживают волнистость сложных функций

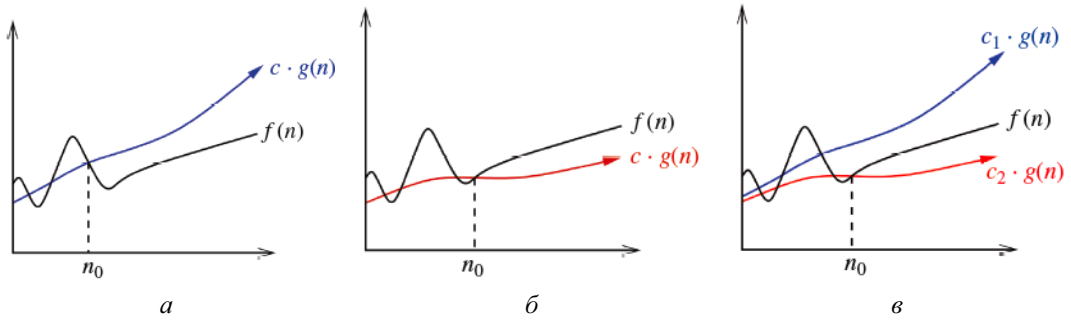


Рис. 2.3. Графическая иллюстрация асимптотических обозначений:
a — $f(n) = O(g(n))$; *б* — $f(n) = \Omega(g(n))$; *в* — $f(n) = \Theta(g(n))$

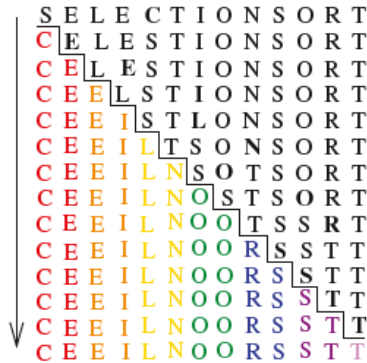


Рис. 2.4. Графическая иллюстрация работы алгоритма сортировки методом выбора

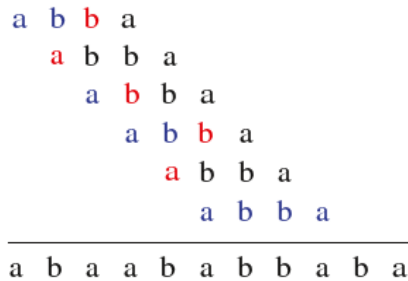


Рис. ЦВ-2.5. Пример поиска подстроки *abba* в тексте *abaababbaba*. Буквы синего цвета представляют совпадения с требуемой подстрокой, а красного — несовпадения. Поиск прекращается при обнаружении первого полного совпадения

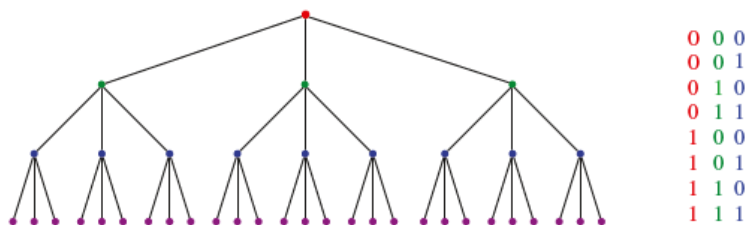


Рис. 2.6. Дерево высотой h и с количеством потомков d для каждого узла имеет d^h листьев.

В нашем случае $h = 3$, $d = 3$ (слева). Количество комбинаций битов (справа) растет экспоненциально с увеличением длины комбинации. Эти комбинации можно описать как пути от корня к листу в двоичном дереве высотой $h = 3$

ГЛАВА 3



Рис. 3.1. Пример связного списка с полями данных и указателями

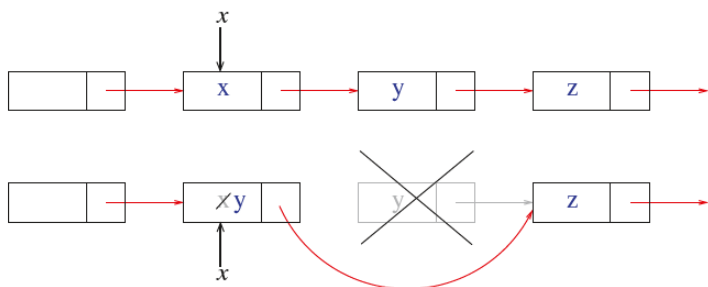


Рис. 3.2. Элемент можно удалить без обращения к предшествующему ему элементу списка, перезаписав его содержимое и удалив следующий за ним первоначально элемент

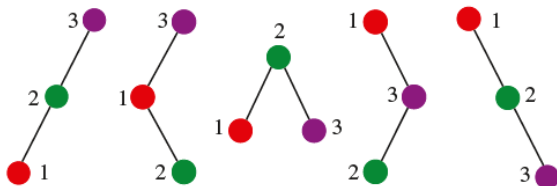
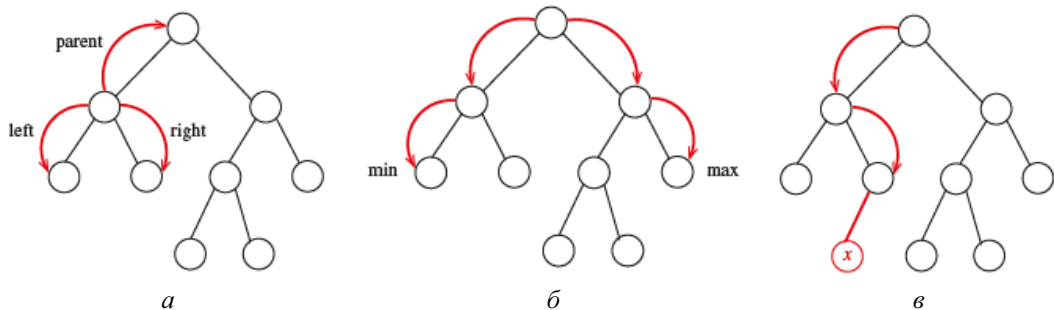


Рис. 3.3. Пять разных двоичных деревьев поиска с тремя узлами.

Ключи всех элементов в левом поддереве узла x имеют значение $< x$ (а правом — $> x$)

Рис. 3.4. Отношения в двоичном дереве: a — родительский и дочерние элементы;

$б$ — поиск в двоичном дереве наименьшего и наибольшего элементов;

$в$ — вставка нового элемента в правильное место

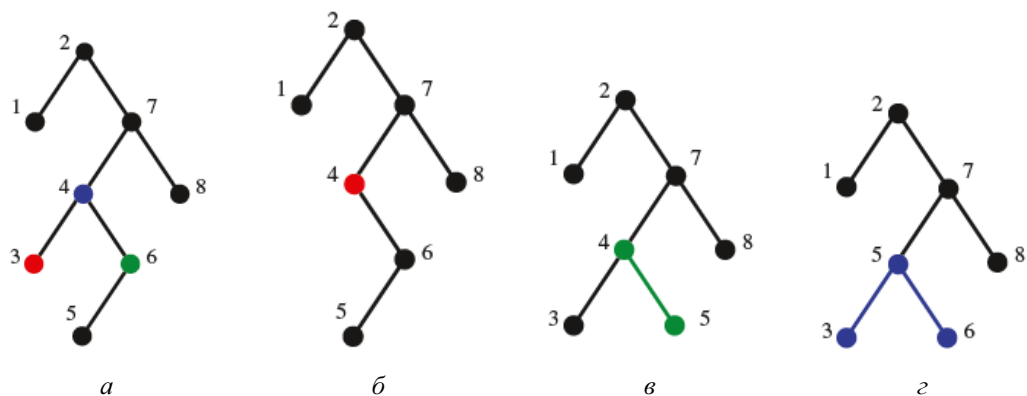


Рис. ЦВ-3.5. *а* — исходное дерево; *б* — удаление узла дерева без потомков (3); *в* — удаление узла дерева с одним потомком (6); *г* — удаление узла дерева с двумя потомками (4). Затронутые в результате удаления узлы обозначены разными цветами

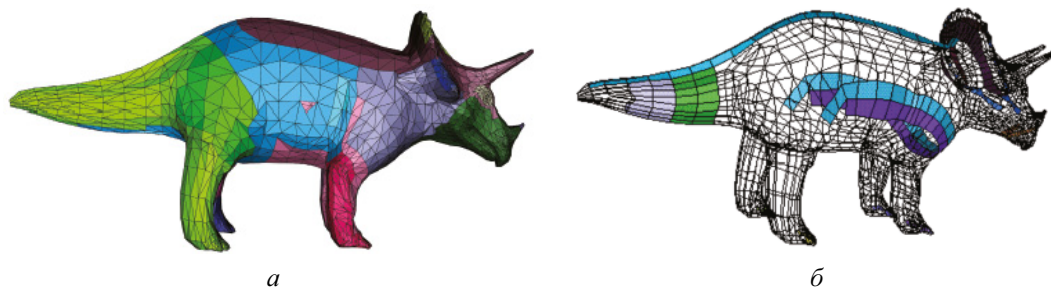


Рис. 3.6. Триангуляционная модель динозавра (*а*); использование нескольких полос треугольников (*б*)

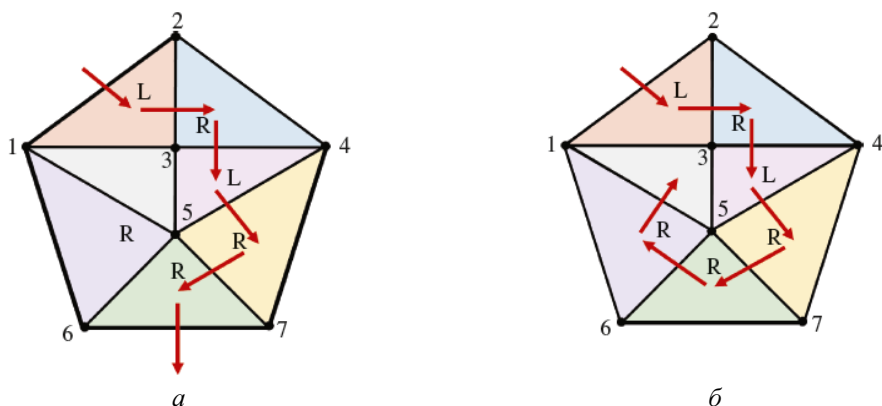


Рис. 3.7. Извлечение полосы треугольников из треугольной сетки: полоса с частичным покрытием с использованием чередующихся левых и правых поворотов (*а*) и полоса с полным покрытием с использованием гибкости произвольных поворотов (*б*)

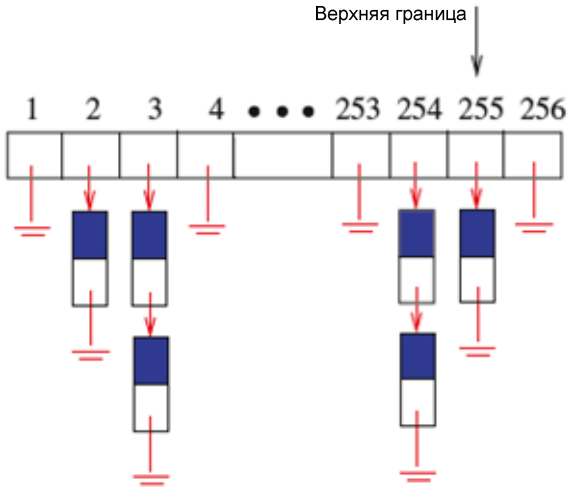


Рис. 3.8. Ограниченная по высоте очередь с приоритетами для полос треугольников

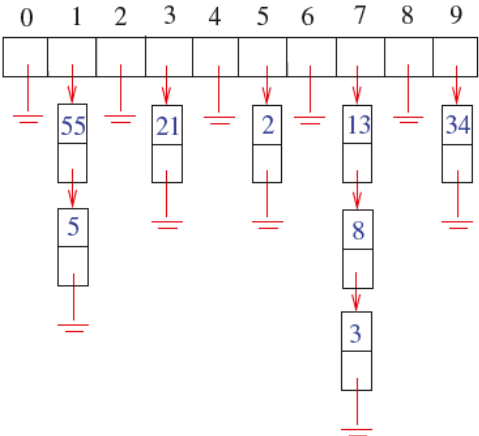


Рис. 3.9. Разрешение коллизий при помощи цепочек после хэширования первых восьми чисел Фибоначчи в возрастающем порядке посредством функции хэширования $H(x) = (2x + 1) \bmod 10$. На рисунке вставки делаются в начале каждого списка

0	1	2	3	4	5	6	7	8	9
34	5	55	21		2		3	8	13

Рис. ЦВ-3.10. Разрешение коллизий методом открытой адресации и последовательного исследования после вставки первых восьми чисел Фибоначчи в возрастающем порядке посредством функции хэширования $H(x) = (2x + 1) \bmod 10$. Элементы, выделенные красным, были вставлены в первую свободную ячейку после недоступной желаемой ячейки

					T	A	T	C	C
			T		T	A	T	C	
		G	T		T	A	T		
	C	G	T		T	A			
A	C	G	T	T	A	T	C	C	A

Рис. 3.11. Конкатенация двух строк может входить в строку S , только если в нее входят все объединяемые строки

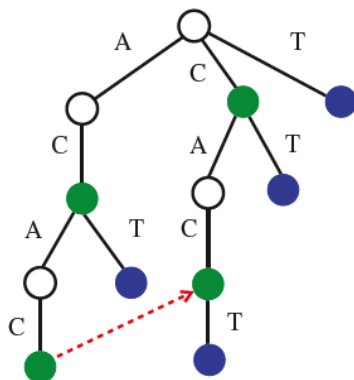


Рис. ЦВ-3.12. Суффиксное дерево строк $ACAC$ и $CACT$ с указателем на суффикс строки $ACAC$. Узлы зеленого цвета соответствуют суффиксам $ACAC$, а синего цвета — суффиксам $CACT$

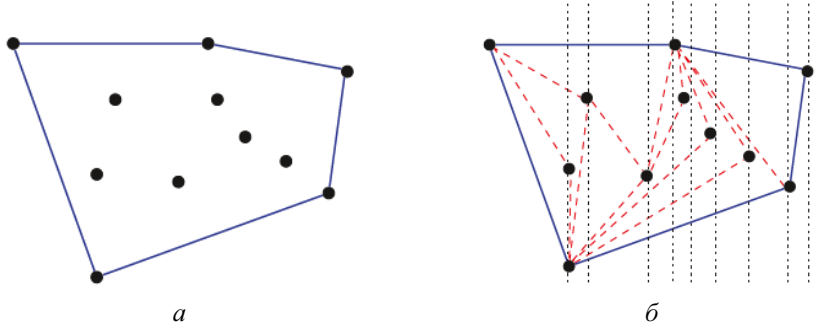


Рис. 4.1. Создание выпуклой оболочки: резиновой нитью (а), вставкой точек слева направо (б)

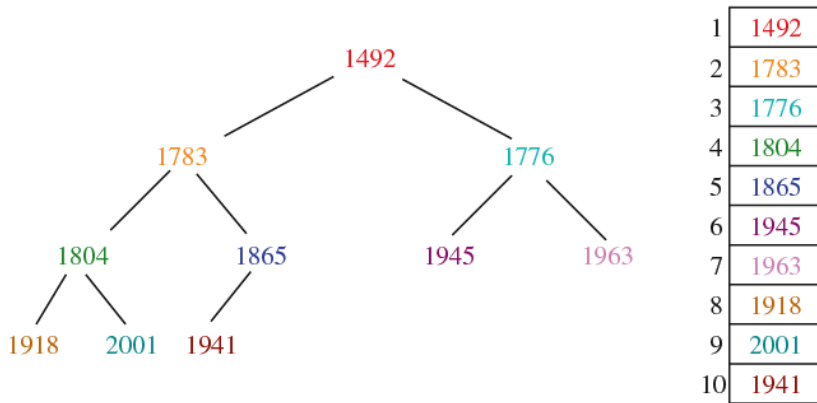


Рис. 4.2. Пирамидальное дерево дат важных событий в американской истории (слева) и соответствующий массив (справа)

		X+Y
X	Y	
\$100	\$50	\$150 (1,1)
		\$160 (2,1)
		\$175 (1,2)
		\$180 (3,1)
\$110	\$75	\$185 (2,2)
		\$205 (3,2)
\$130	\$125	\$225 (1,3)
		\$235 (2,3)
		\$255 (3,3)

Рис. 4.3. Сортировка в возрастающем порядке сумм X и Y

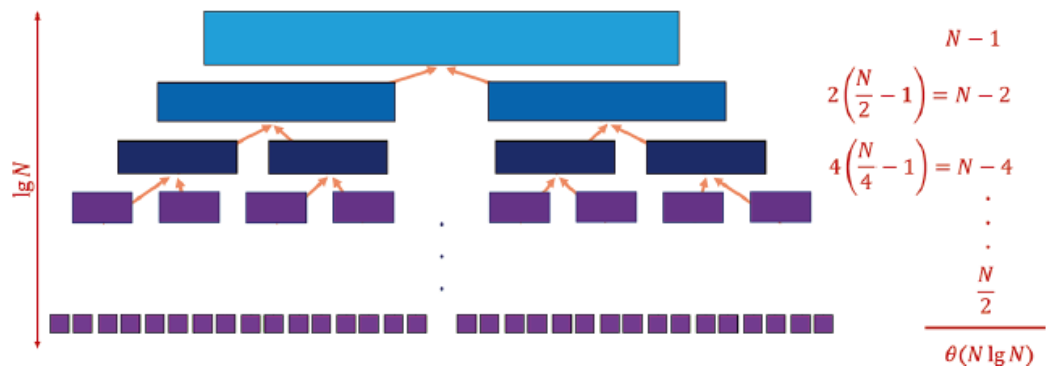


Рис. 4.4. Рекурсивное дерево для алгоритма сортировки слиянием. Высота дерева составляет $\lceil \lg_2 n \rceil$, а стоимость операций слияний на каждом уровне равна $\Theta(n)$, что дает нам алгоритм с временной сложностью, равной $\Theta(n \log n)$

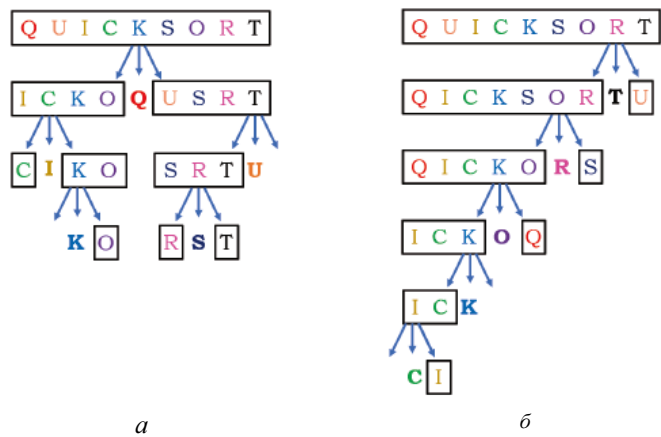


Рис. 4.5. Графическая иллюстрация работы алгоритма быстрой сортировки: сначала в каждом подмассиве в качестве разделителя выбирается первый элемент (а), а затем последний (б)

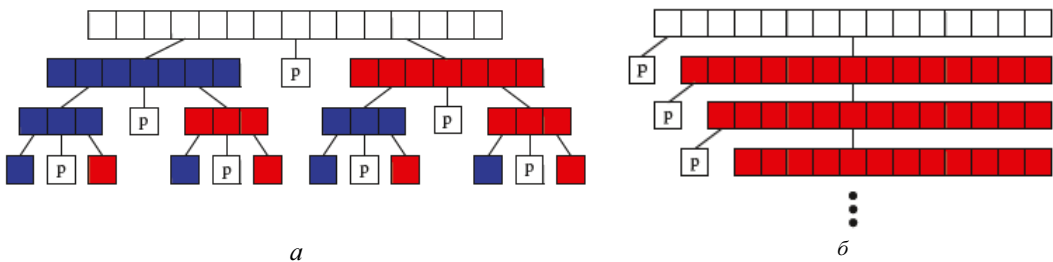


Рис. ЦВ-4.6. Рекурсивные деревья алгоритма быстрой сортировки: наилучший случай (а) и наихудший случай (б). Левая часть обозначена синим цветом, а правая — красным

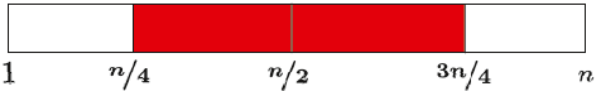


Рис. 4.7. В половине случаев разделительный элемент расположен ближе к середине массива, чем к его краям

Shifflett Debbie K Ruckersville	985-7957	Shifflett James 2219 Williamsburg Rd
Shifflett Debra S SR 617 Quinque	985-8813	Shifflett James B 801 Stonehenge Av
Shifflett Delma SR609	985-3688	Shifflett James C Stanardsville
Shifflett Delmas Crozet	823-5901	Shifflett James E Earlysville
Shifflett Dempsey & Marilyn		Shifflett James E Jr 552 Cleveland Av
100 Greenbrier Ter	973-7195	Shifflett James F & Lois Longmeadow
Shifflett Denise Rt 627 Dyke	985-8097	Shifflett James F & Vernell Rt671
Shifflett Dennis Stanardsville	985-4560	Shifflett James J 1430 Rugby Av
Shifflett Dennis H Stanardsville	985-2924	Shifflett James K St George Av
Shifflett Dewey E Rt667	985-6576	Shifflett James L SR33 Stanardsville
Shifflett Dewey O Dyke	985-7269	Shifflett James O Earlysville
Shifflett Diana 508 Bainbridge Av	979-7035	Shifflett James O Stanardsville
Shifflett Doby & Patricia Rt6	286-4227	Shifflett James R Old Lynchburg Rd ..
Shifflett Don&Ola Rt 621	974-7463	Shifflett James R Rt753 Esmork

Рис. 4.8. Фрагмент телефонного справочника города Шарлотсвилл в штате Вирджиния

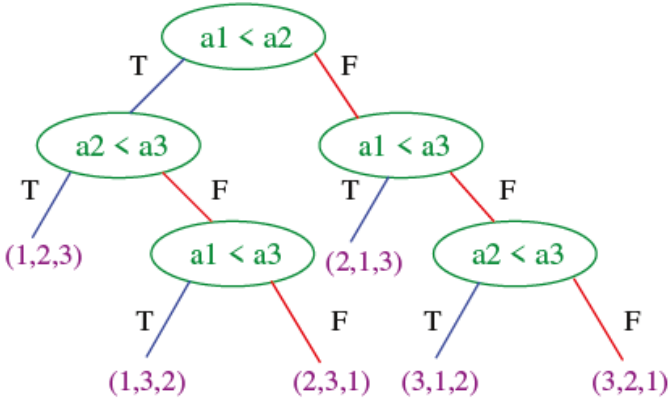


Рис. 4.9. Демонстрация работы упорядочивания элементов массива **a** сортировкой вставками с помощью дерева решений. Каждый узел дерева представляет конкретную входную перестановку, а путь от корня к конечному узлу описывает последовательность сравнений, выполняемых алгоритмом для ее упорядочивания



Рис. ЦВ-5.1. Проекты четырех синтетических генов, используемых для локализации места конкретной последовательности. Зеленые области взяты из жизнеспособной последовательности, а красные — из летально дефектной последовательности. Гены проектов II, III и IV были жизнеспособны, тогда как ген проекта I оказался дефектным. Такой результат можно объяснить только наличием летального места в пятой справа области

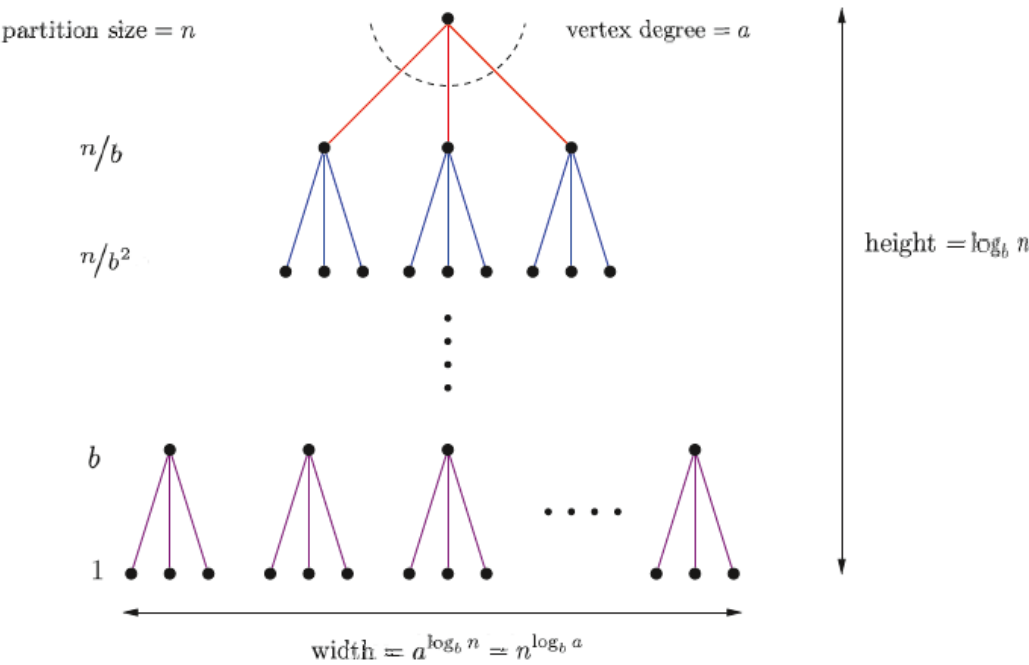


Рис. 5.2. Рекурсивное дерево, полученное в результате разложения каждой задачи размером n на a задач размером n/b

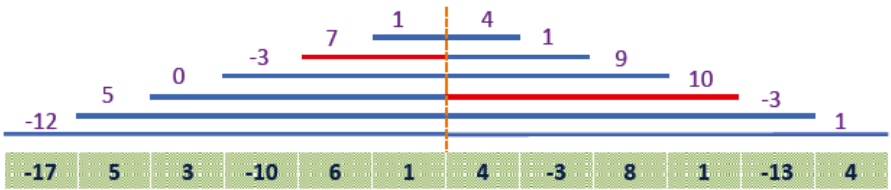


Рис. ЦВ-5.3. Поддиапазон с наибольшей суммой значений будет находиться или полностью в левой, или полностью в правой части, или же, как в этом случае, будет суммой наибольших поддиапазонов слева и справа от центра всего диапазона

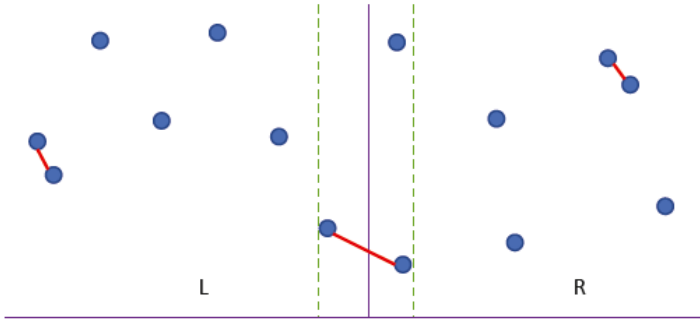


Рис. 5.4. Ближайшая пара точек в двумерном пространстве находится или слева, или справа от центра, или же в узкой центральной полосе



Рис. 5.5. При изменении образца на обратный свертка строк становится эквивалентной сравнению строк

ГЛАВА 6

$$S = \{(1, 1), (1, 2), (1, 3), (1, 4), (1, 5), (1, 6), \\ (2, 1), (2, 2), (2, 3), (2, 4), (2, 5), (2, 6), \\ (3, 1), (3, 2), (3, 3), (3, 4), (3, 5), (3, 6), \\ (4, 1), (4, 2), (4, 3), (4, 4), (4, 5), (4, 6), \\ (5, 1), (5, 2), (5, 3), (5, 4), (5, 5), (5, 6), \\ (6, 1), (6, 2), (6, 3), (6, 4), (6, 5), (6, 6)\}.$$

Рис. 6.1. 36 результатов бросания двух шестигранных игральных костей

$$E = \{(1, 6), (2, 5), (3, 4), (4, 3), (5, 2), (6, 1), (5, 6), (6, 5)\}$$

Рис. 6.2. Подмножество событий, при которых сумма значений костей будет равной 7 или 11

$$A - B = \{(1, 2), (1, 4), (2, 1), (2, 2), (2, 3), (2, 4), (2, 6), (3, 2), (3, 6), (4, 1), \\ (4, 2), (4, 4), (4, 5), (4, 6), (5, 4), (6, 2), (6, 3), (6, 4), (6, 6)\}$$

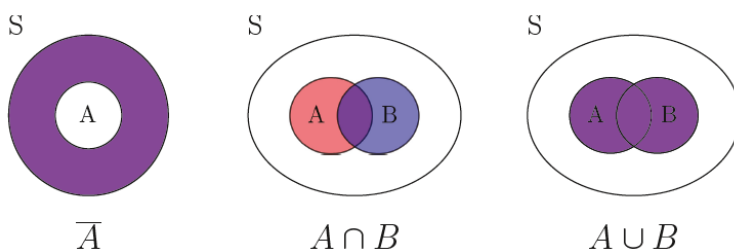
Рис. 6.3. Результаты события A , не являющиеся результатами события B 

Рис. 6.4. Диаграммы Венна, иллюстрирующие разность (слева), пересечение (в центре) и объединение (справа) множеств

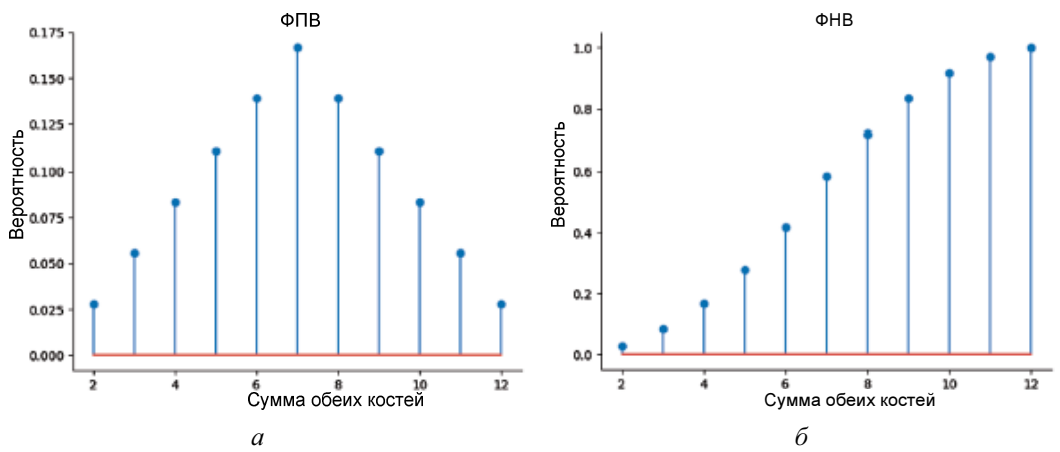


Рис. 6.5. Функция плотности вероятностей (ФПВ) (а) суммы двух костей содержит точно такую же информацию, что и функция накопленных вероятностей (ФНВ) (б), но выглядит совершенно иначе

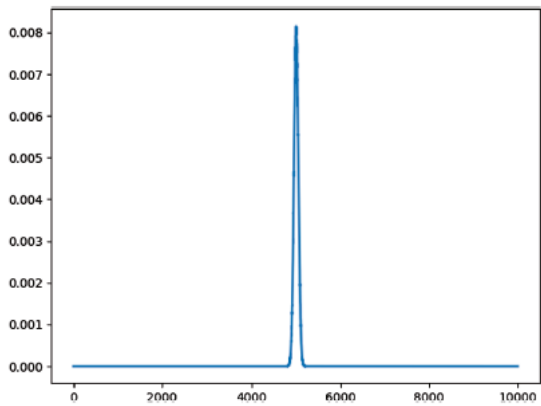


Рис. 6.6. Распределение вероятности выпадения h орлов при n бросках правильной монеты плотно отцентрировано вокруг среднего значения и равно $n/2$. В нашем примере для $n = 10\,000$ распределение $h = n/2 = 5\,000$

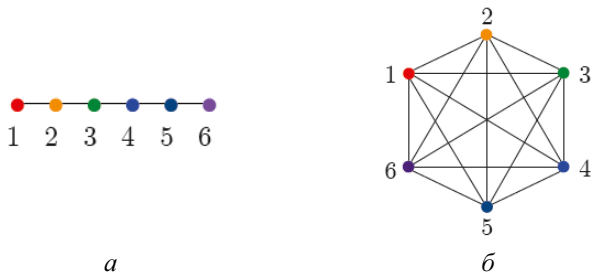


Рис. 6.7. Сколько займет проход случайным образом по всем n вершинам маршрута (а) и графа (б)?

<i>k</i>	Количество корзин, содержащих <i>k</i> объектов		
	<i>n</i> = 10 ⁶	<i>n</i> = 10 ⁷	<i>n</i> = 10 ⁸
0	367,899	3,678,774	36,789,634
1	367,928	3,677,993	36,785,705
2	183,926	1,840,437	18,392,948
3	61,112	613,564	6,133,955
4	15,438	152,713	1,531,360
5	3130	30,517	306,819
6	499	5,133	51,238
7	56	754	7,269
8	12	107	972
9		8	89
10			10
11			1

Рис. 6.8. Результаты для таблиц хэширования, содержащих от одного миллиона до ста миллионов объектов

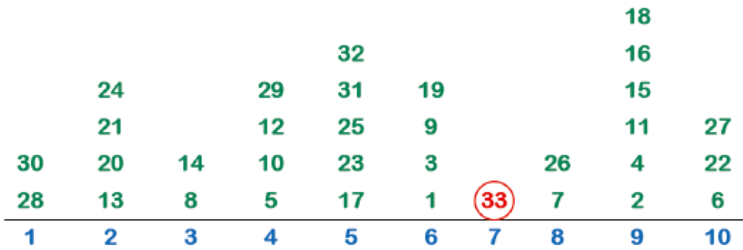


Рис. 6.9. Результаты эксперимента по бросанию мячиков в контейнеры случайным образом. Все контейнеры заполняются только к 33-му броску

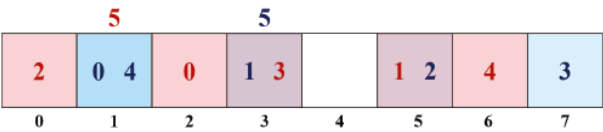


Рис. ЦВ-6.10. Хэширование целых чисел 0, 1, 2, 3 и 4 в фильтре Блума с $n = 8$ с использованием функции хэширования $h(x) = 2x + 1$ (синий цвет) и $h(x) = 3x + 2$ (красный цвет). При поиске $x = 5$ будет возвращен ложный положительный результат, поскольку соответствующие два бита были установлены другими элементами

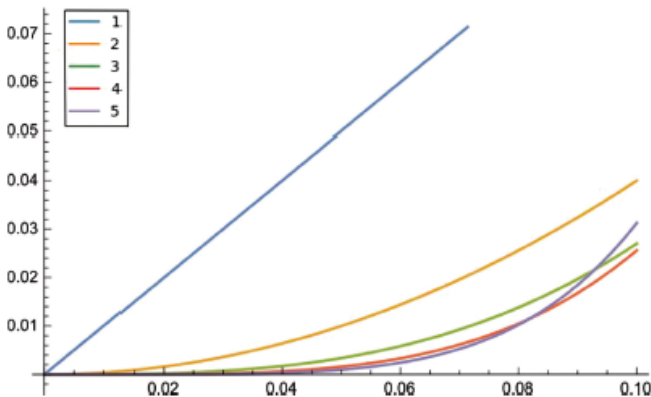


Рис. ЦВ-6.11. Вероятность ошибки фильтра Блума в зависимости от нагрузки для значений k от 1 до 5. Выбрав правильное значение k для заданной нагрузки, можно значительно снизить частоту ошибок ложных коллизий, не увеличивая при этом размер таблицы

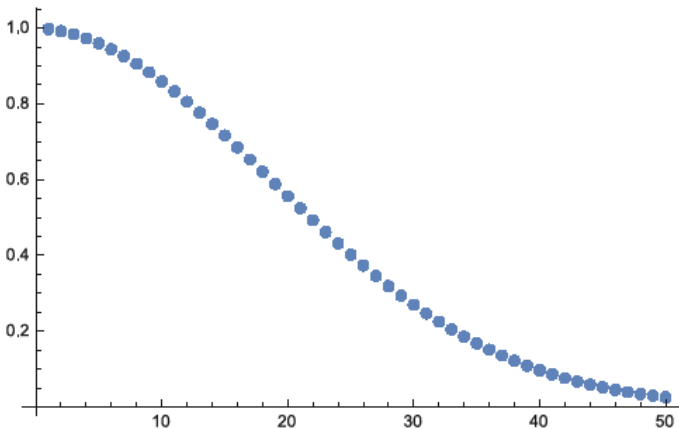


Рис. 6.12. Вероятность отсутствия коллизий быстро нарастает с увеличением количества ключей n для хэширования. В рассматриваемом случае размер таблицы хэширования $m = 365$

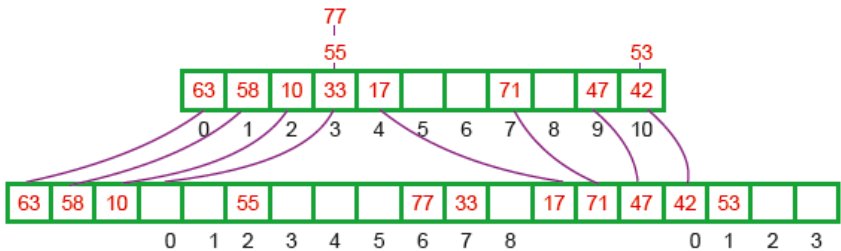


Рис. 6.13. В идеальном хэшировании используется двухуровневая таблица хэширования, что гарантирует постоянное время поиска. В таблице первого уровня кодируется диапазон индексов элементов таблицы второго уровня количеством l_i^2 , выделенных для хранения l_i элементов в корзине i первого уровня

s:	The	cat	in	the	hat	The	hat	in	the	store
h(s):	17	128	56	17	4	17	4	56	17	96

Рис. 6.14. Если два документа очень похожи, то слова, соответствующие хэш-кодам с минимальным значениями в каждом из этих документов, будут, скорее всего, одинаковыми

$H(s, j)$			0	1	2	5	3	6	5
$\sum C_i + j$			0	1	1	2	2	2	2
	A	A	A	B	A	B	B	A	B

Рис. ЦВ-6.15. Функция хэширования по алгоритму Рабина-Карпа $H(s, j)$ выдает различные хэш-коды для разных подстрок (отображены синим цветом), тогда как менее мощная функция хэширования, которая просто суммирует коды символов, создает много коллизий (отображены фиолетовым цветом). В нашем случае длина искомой строки (BBA) составляет $m = 3$, а коды символов равны $A = 0$ и $B = 1$

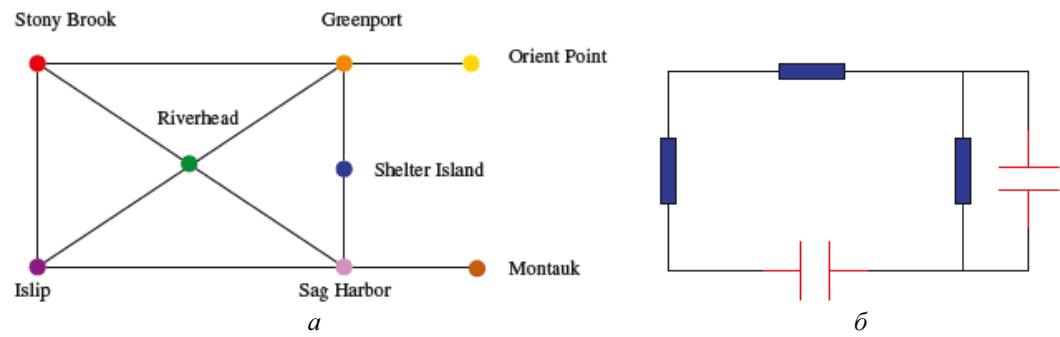


Рис. 7.1. Моделирование посредством графов сети дорог (а) и электрических схем (б)

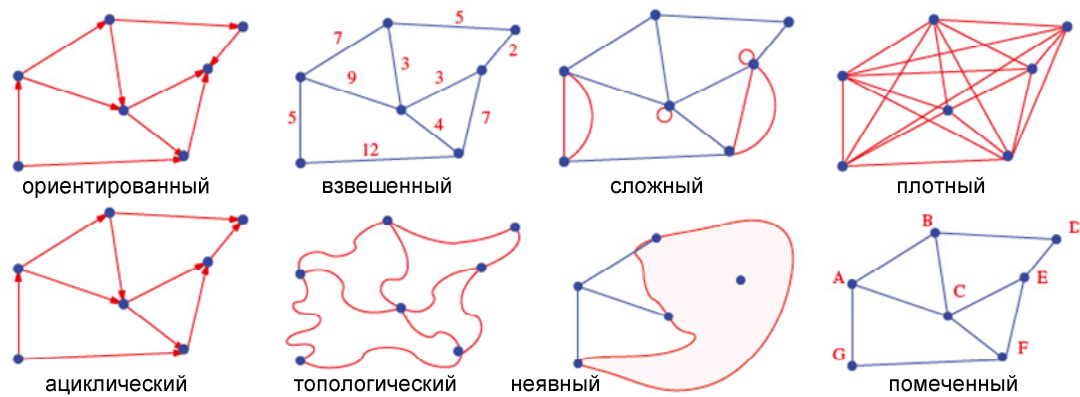


Рис. 7.2. Основные разновидности графов

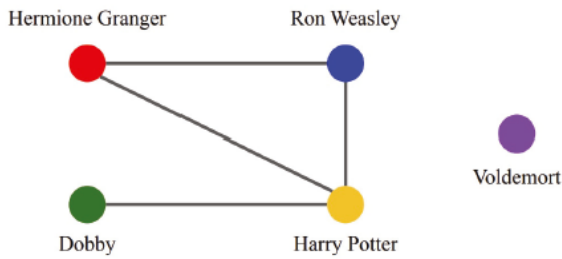


Рис. 7.3. Часть графа дружеских отношений персонажей книги о Гарри Поттере

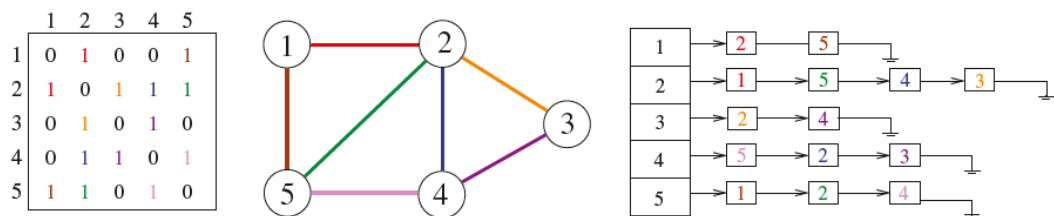


Рис. ЦВ-7.4. Представление графа (в центре) матрицей смежности (слева) и списком смежности (справа). Конкретные ребра обозначены соответствующими цветами

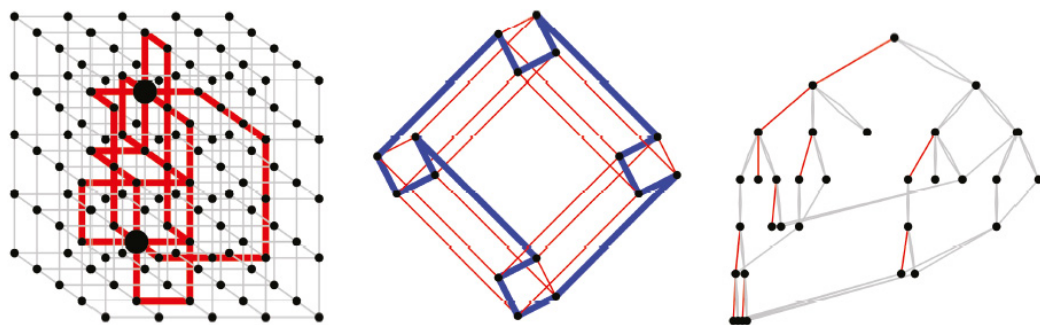


Рис. 7.5. Репрезентативные графы в Combinatorica: пути, не имеющие общих ребер (слева), Гамильтонов цикл в гиперкубе (в центре), обход в глубину дерева поиска (справа)

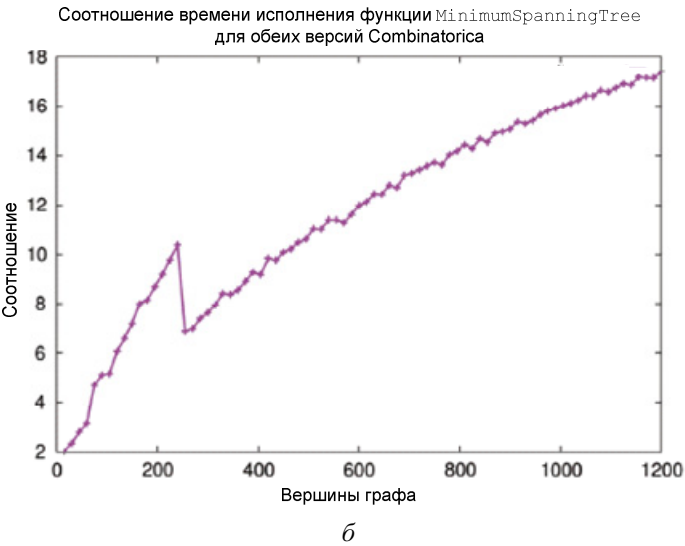
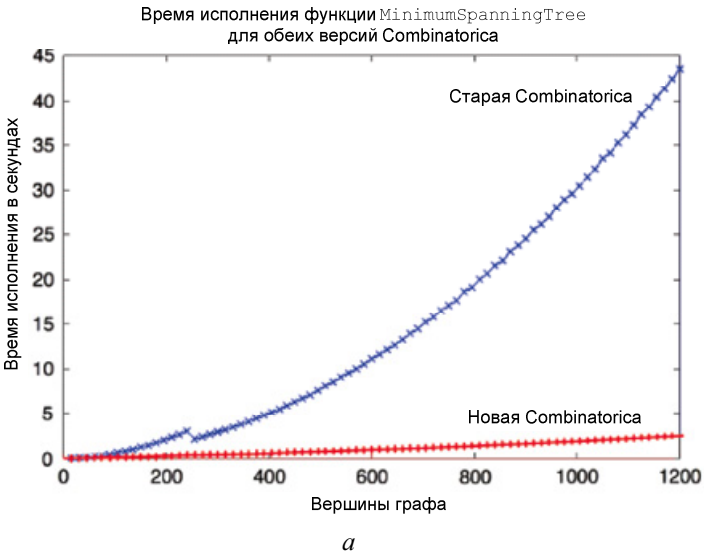


Рис. 7.6. Сравнение производительности старой и новой версий `Combinatorica`: абсолютное время исполнения для каждой версии (*a*) и соотношение времени исполнения (*b*)

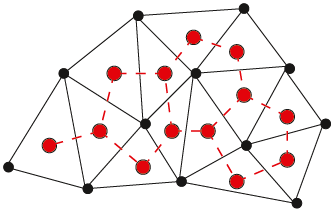


Рис. 7.7. Двойственный граф (пунктирные линии) триангулированной поверхности

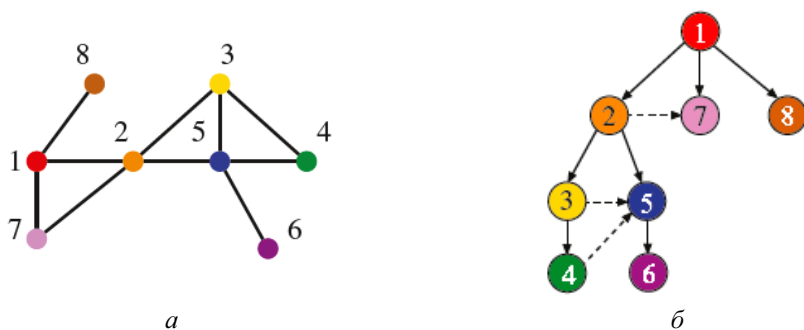


Рис. 7.8. Неориентированный граф (а) и его дерево обхода в ширину (б).
Пунктирные линии не являются частью дерева, а обозначают ребра графа,
идущие к открытым или обработанным вершинам

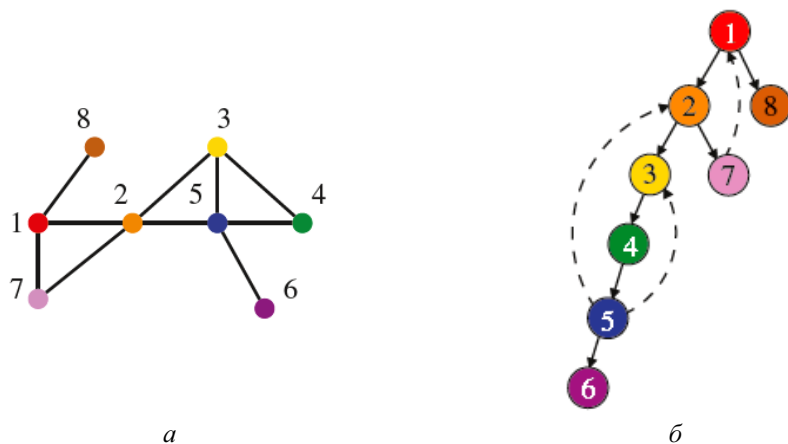


Рис. 7.9. Неориентированный граф (а) и его дерево обхода в глубину (б).
Пунктирные линии не являются частью дерева, а лишь обозначают обратные ребра

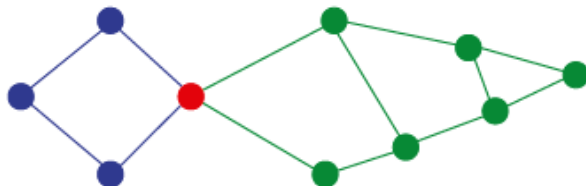


Рис. 7.10. Граф телефонной сети врага, которую нужно вывести из строя

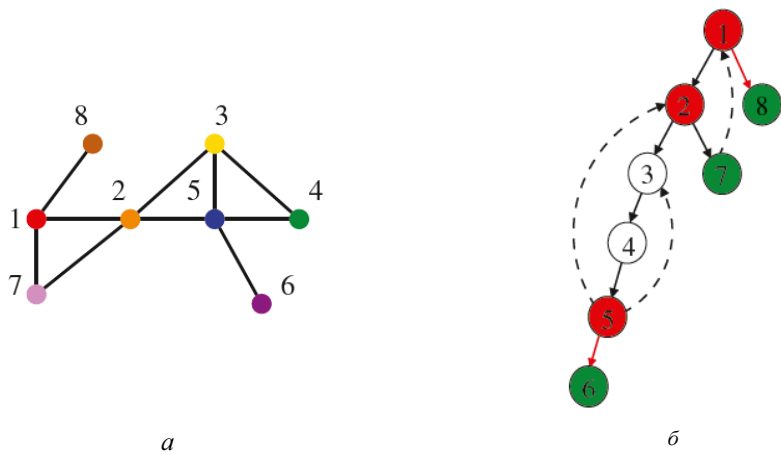


Рис. ЦВ-7.11. Связный граф (а) телефонной сети и его дерево обхода в глубину (б), содержащее шарнирные вершины 1, 2 и 5. Обратные ребра не дают вершинам 3 и 4 быть шарнирами, а зеленые вершины 6, 7 и 8 не могут быть шарнирами, потому, что они листья (концевые узлы) дерева обхода графа в глубину. Красные ребра (1,8) и (5,6) являются мостовыми ребрами, удаление которых превращает граф в несвязный

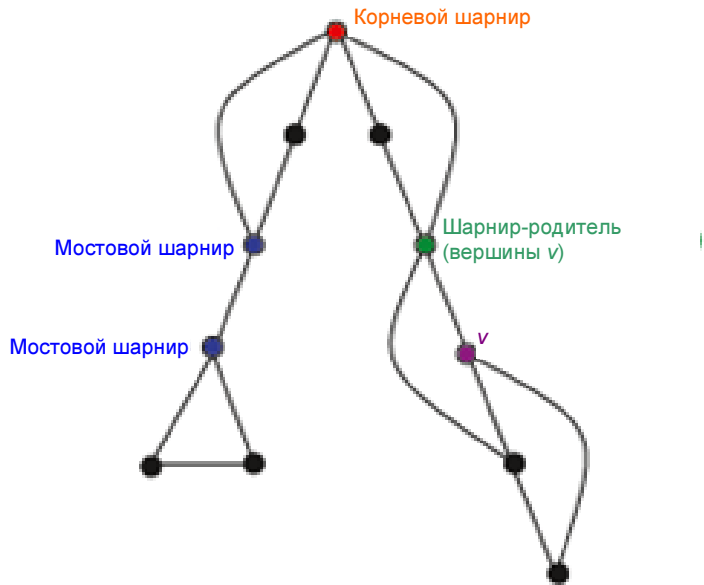


Рис. 7.12. Типы шарниров

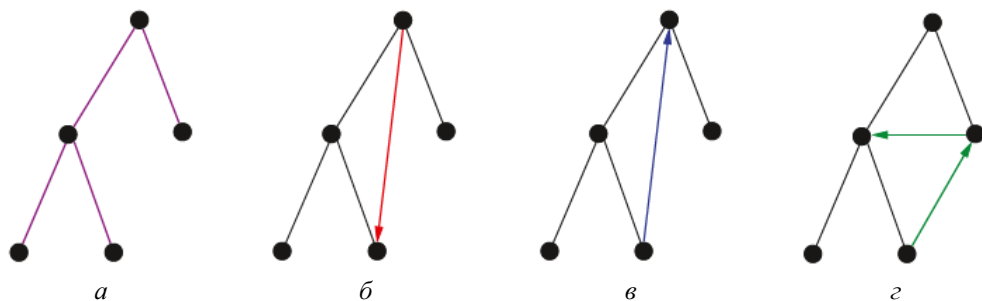


Рис. 7.13. Возможные типы ребер при обходе графа: a — древесные ребра; $б$ — прямое ребро; $в$ — обратное ребро; $г$ — поперечные ребра. Прямые и поперечные ребра могут встречаться при обходе в глубину *только* в ориентированных графах

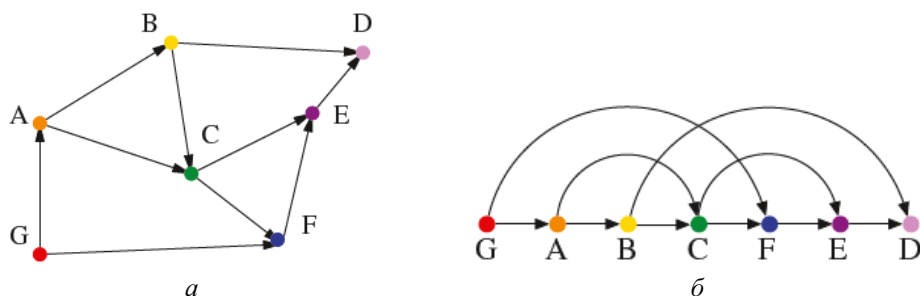


Рис. 7.14. Бесконтурный ориентированный граф (a) с единственным топологическим упорядочиванием: G, A, B, C, F, E, D ($б$)

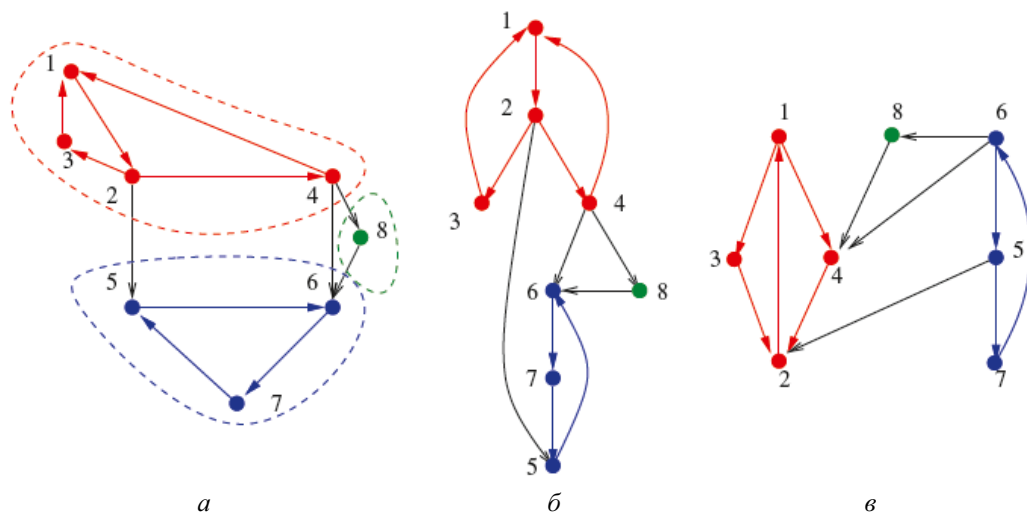


Рис. 7.15. Сильно связанные компоненты графа (a) и соответствующее дерево обхода в глубину ($б$). Инвертируя конечный порядок вершин обхода в глубину с вершины 1, получим множество $[3, 5, 7, 6, 8, 4, 2, 1]$, что определяет порядок вершин для второго обхода, на этот раз транспонированного графа G^T ($в$)

ГЛАВА 8

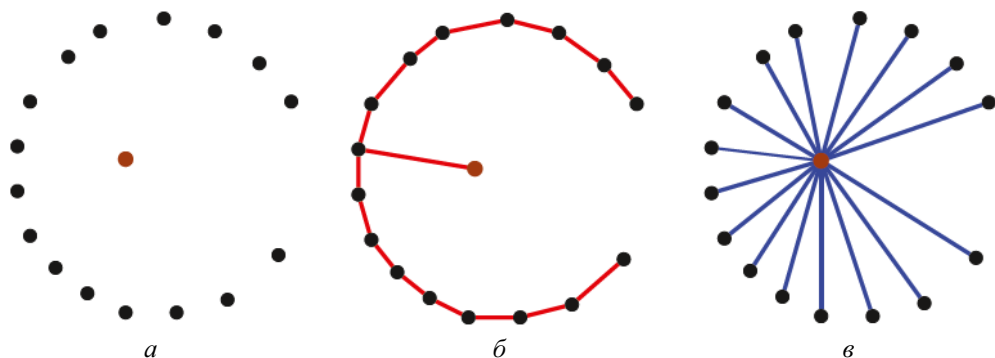


Рис. 8.1. Полный взвешенный граф, определяемый расстоянием между набором точек (а), его минимальное остовное дерево (б) и кратчайший маршрут от центра дерева (в)

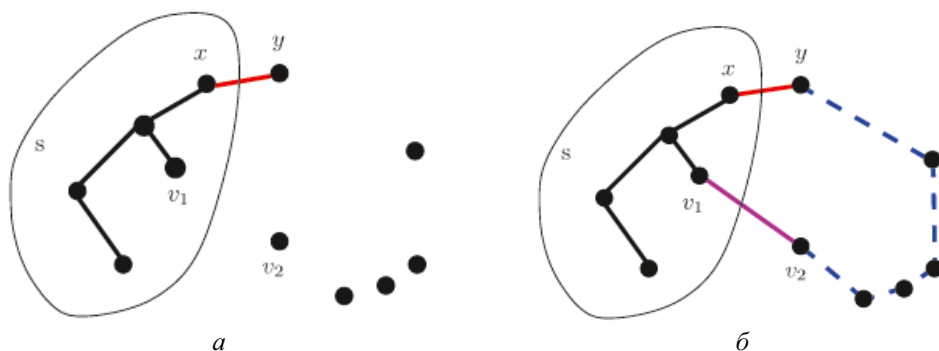


Рис. 8.2. Выдает ли алгоритм Прима неправильное решение? Нет, поскольку выбор ребра (x,y) перед ребром (v_1,v_2) подразумевает, что $\text{вес}(v_1, v_2) \geq \text{вес}(x, y)$

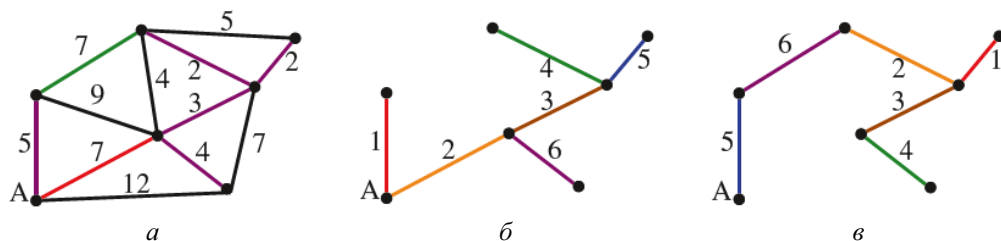


Рис. 8.3. Граф G (а) и минимальные остовные деревья, создаваемые алгоритмами Прима (б) и Крускала (в). Цифры и цвета ребер указывают порядок их вставки, приоритет вставки равнозначных ребер определяется произвольным путем

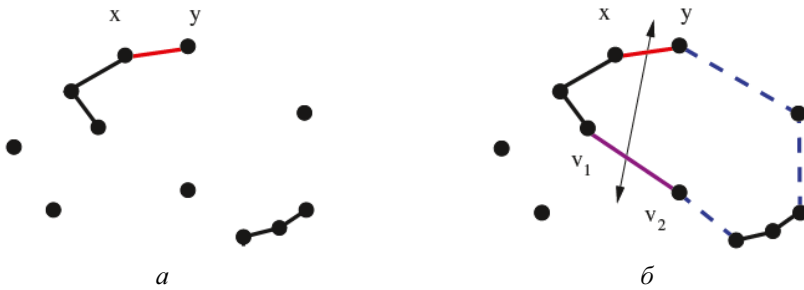


Рис. 8.4. Мог бы алгоритм Крускала выдать неправильное решение, выбрав красное ребро (x, y) (а)?

Нет, не мог бы, потому что вставляемое позже ребро (v_1, v_2) (б) должно быть тяжелее, чем ребро (x, y)

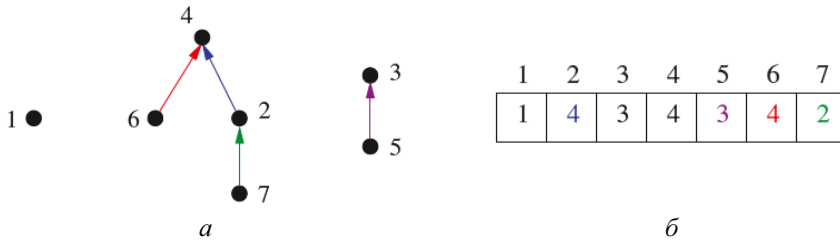


Рис. 8.5. Пример структуры данных «поиск-объединение»: в виде леса деревьев (а) и в виде массива указателей на родительские элементы (б)

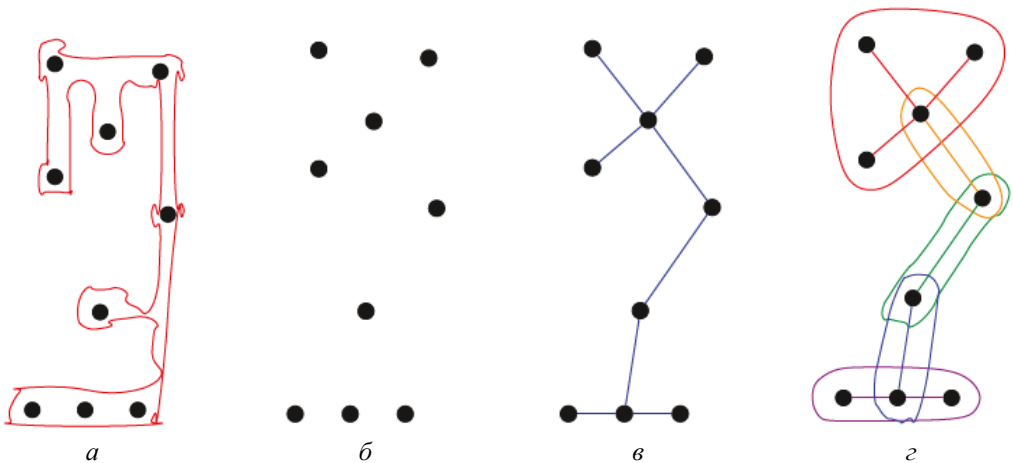


Рис. 8.6. Пример сети печатной платы: металлизированная соединяющая дорожка (а), контактные точки (б), минимальное остовное дерево контактных точек (в), контактные точки, разбитые на кластеры (г)

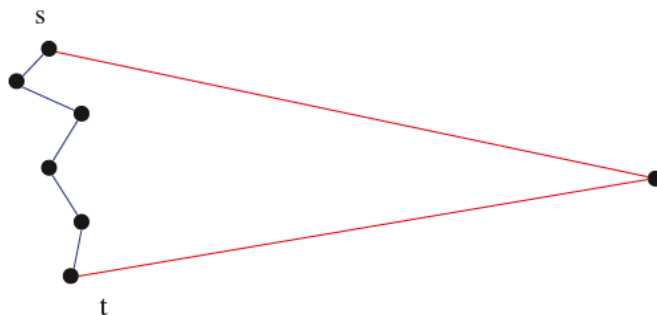


Рис. 8.7. Кратчайший путь от точки s к точке t может проходить через несколько промежуточных точек вместо того, чтобы использовать наименьшее количество ребер

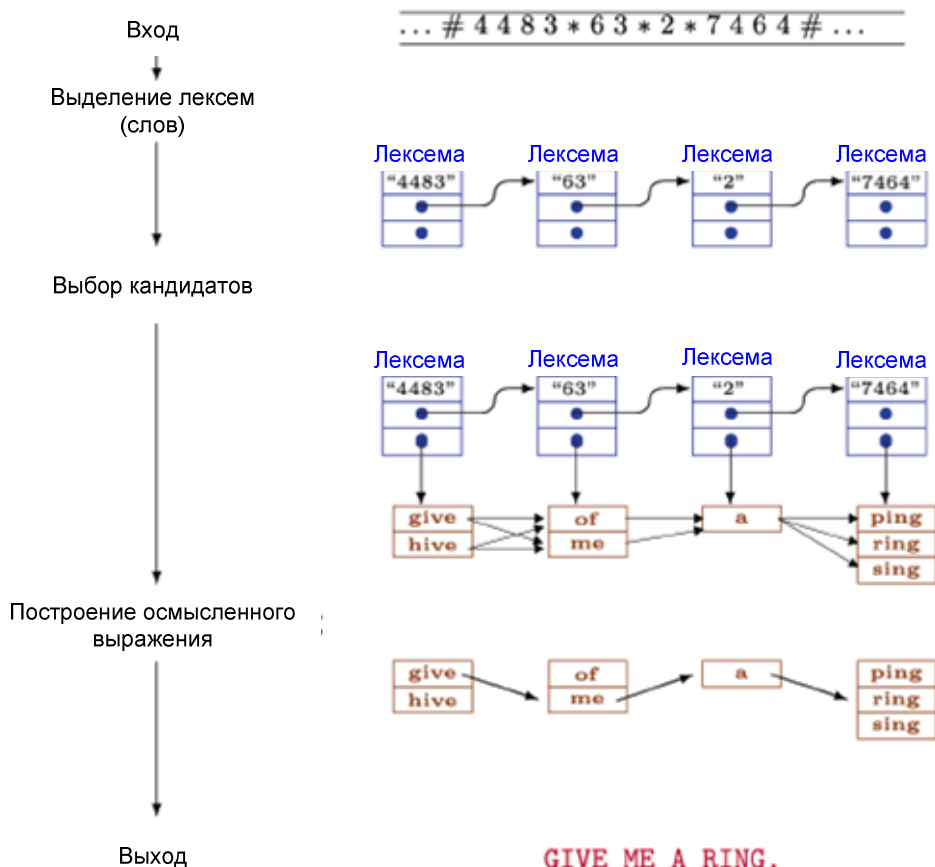


Рис. 8.8. Этапы восстановления слов из последовательностей нажатых кнопок номеронабирателя

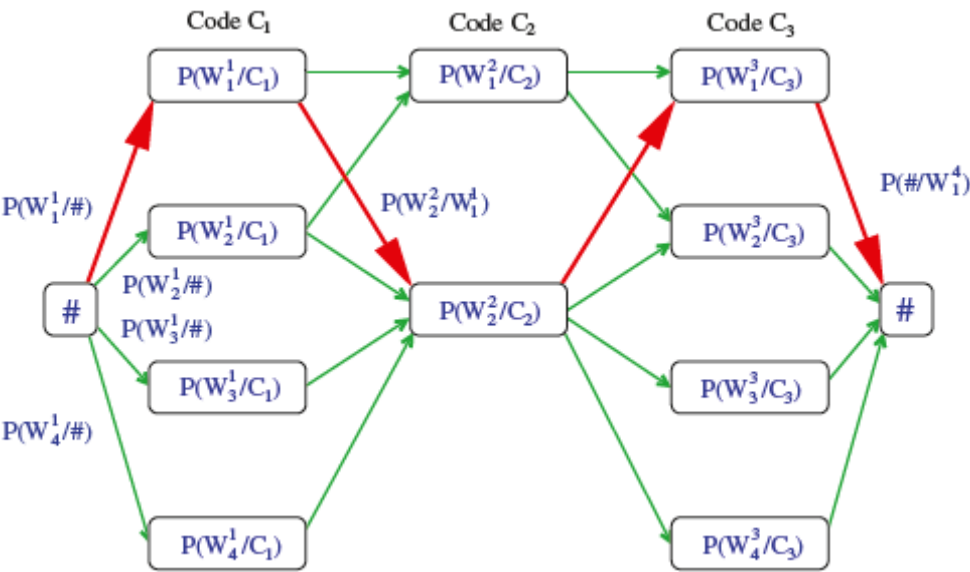


Рис. 8.9. Путь с наименьшей стоимостью представляет самую лучшую интерпретацию предложения

FOURSCORE AND SEVEN YEARS AGO OUR FATHERS BROUGHT FORTH ON THIS CONTINENT A NEW NATION CONCEIVED IN LIBERTY AND DEDICATED TO THE PROPOSITION THAT ALL MEN ARE CREATED EQUAL. NOW WE ARE ENGAGED IN A GREAT CIVIL WAR TESTING WHETHER THAT NATION OR ANY NATION SO CONCEIVED AND SO DEDICATED CAN LONG ENDURE. WE ARE MET ON A GREAT BATTLEFIELD OF THAT WAS. WE HAVE COME TO DEDICATE A PORTION OF THAT FIELD AS A FINAL SERVING PLACE FOR THOSE WHO HERE HAVE THEIR LIVES THAT THE NATION MIGHT LIVE. IT IS ALTOGETHER FITTING AND PROPER THAT WE SHOULD DO THIS. BUT IN A LARGER SENSE WE CAN NOT DEDICATE WE CAN NOT CONSECRATE WE CAN NOT HALLOW THIS GROUND. THE BRAVE MEN LIVING AND DEAD WHO STRUGGLED HERE HAVE CONSECRATED IT FAR ABOVE OUR POOR POWER TO ADD OR DETRACT. THE WORLD WILL LITTLE NOTE NOR LONG REMEMBER WHAT WE SAY HERE BUT IT CAN NEVER FORGET WHAT THEY DID HERE. IT IS FOR US THE LIVING RATHER TO BE DEDICATED HERE TO THE UNFINISHED WORK WHICH THEY WHO FOUGHT HERE HAVE THUS FAR SO NOBLY ADVANCED. IT IS RATHER FOR US TO BE HERE DEDICATED TO THE GREAT TASK REMAINING BEFORE US THAT FROM THESE HONORED DEAD WE TAKE INCREASED DEVOTION TO THAT CAUSE FOR WHICH THEY HERE HAVE THE LAST FULL MEASURE OF DEVOTION THAT WE HERE HIGHLY RESOLVE THAT THESE DEAD SHALL NOT HAVE DIED IN VAIN THAT THIS NATION UNDER GOD SHALL HAVE A NEW BIRTH OF FREEDOM AND THAT GOVERNMENT OF THE PEOPLE BY THE PEOPLE FOR THE PEOPLE SHALL NOT PERISH FROM THE EARTH.

Рис. 8.10. Результат ввода Геттисбергского послания с номеронабирателя телефона

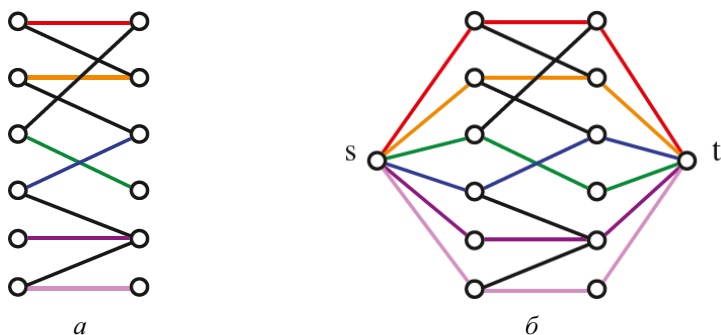


Рис. 8.11. Двудольный граф с максимальным паросочетанием (а) и соответствующий экземпляр потоков в сети (б)

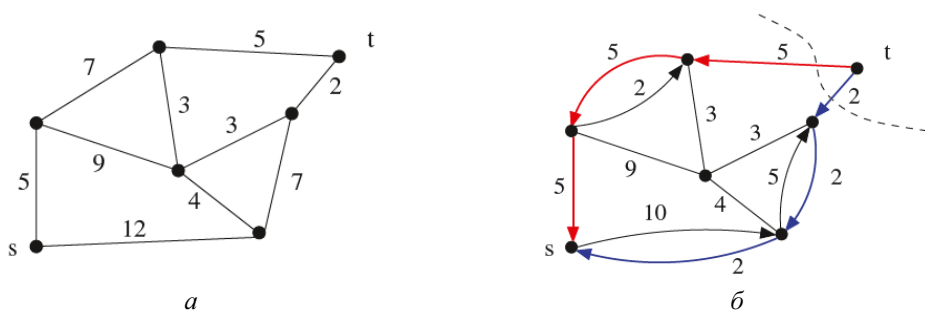


Рис. 8.12. a — максимальный поток от s к t в графе G ;
 $\bar{b}$ — связный граф остаточного потока $R(G, f)$ и минимальный разрез $s-t$ (пунктирная линия рядом с t). Неориентированные ребра в $R(G, f)$ имеют нулевой поток, поэтому они обладают остаточной пропускной способностью в обоих направлениях

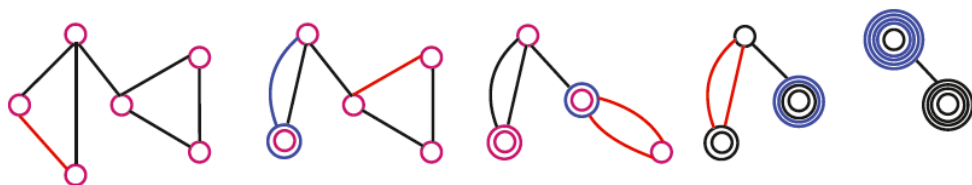


Рис. 8.13. Если нам повезет, последовательные стягивания ребер графа не увеличат размер минимального разреза

ГЛАВА 9

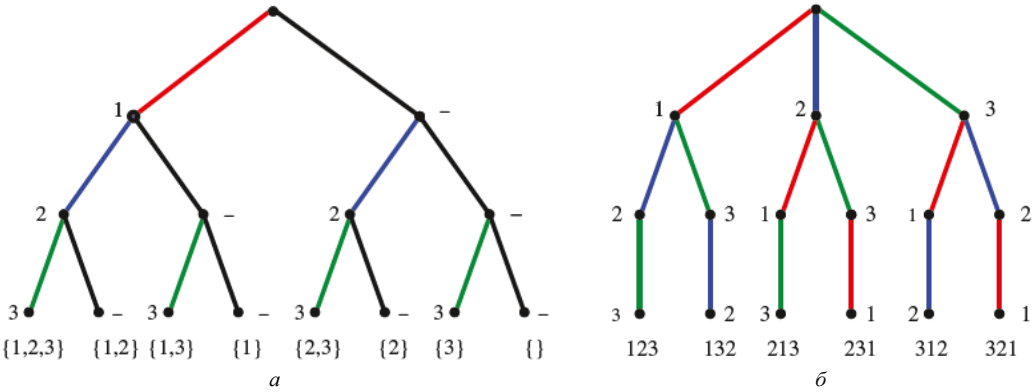


Рис. ЦВ-9.1. Дерево поиска с перечислением всех подмножеств (а) и перестановок (б) множества $\{1, 2, 3\}$. Цвет каждого ребра дерева отображает вставку элемента в частичное решение

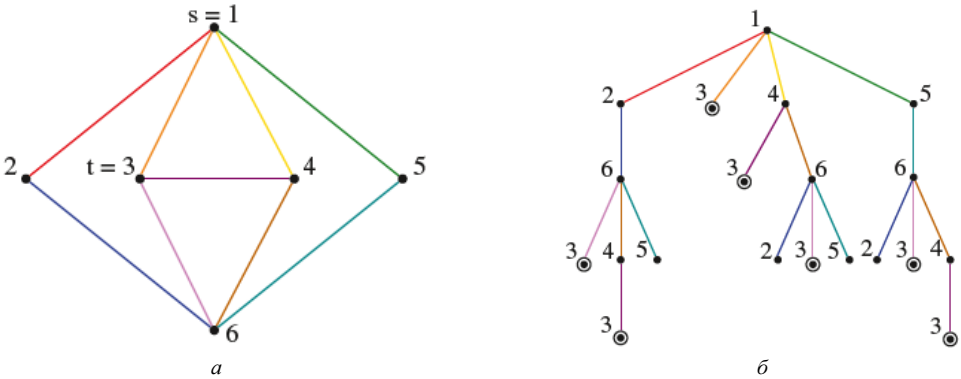


Рис. ЦВ-9.2. Граф (а) и дерево поиска со всеми путями между вершинами s и t (б). Цвет каждого ребра дерева поиска сопоставлен цвету соответствующего ребра графа

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

а

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

б

Рис. 9.3. Головоломка sudoku высокого уровня сложности (а) и ее решение (б)

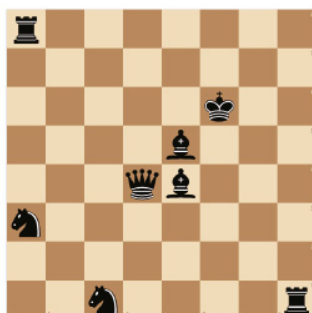
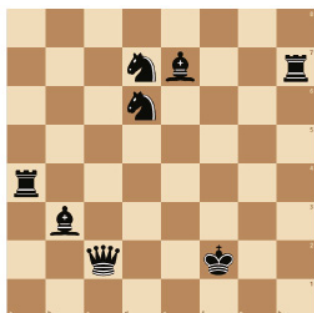


Рис. 9.4. Расстановки фигур, ставящие под удар 63 поля, но не все 64

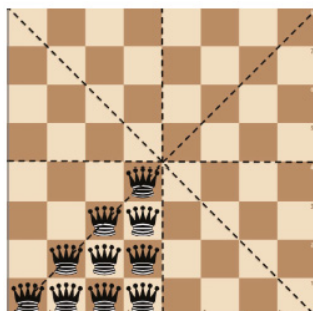


Рис. 9.5. Десять уникальных положений ферзя с учетом симметрии вращения и отражения

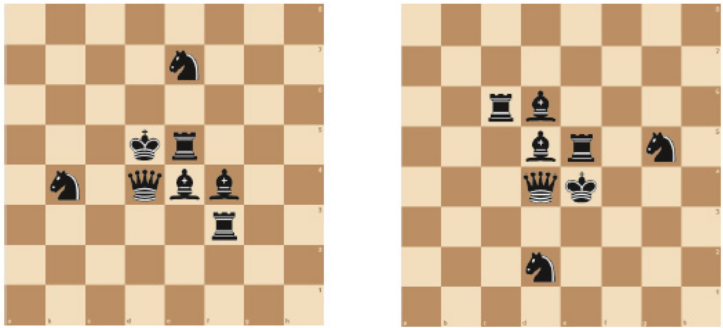


Рис. 9.6. Слабая атака всех 64 полей шахматной доски восемью основными фигурами

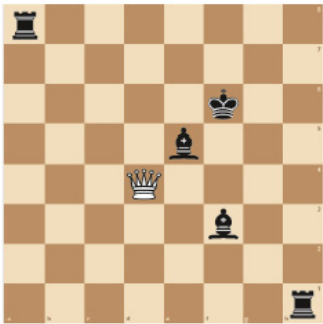


Рис. 9.7. Семи фигур достаточно при размещении ферзя и коня на одном поле (на этом поле сейчас расположен белый ферзь)

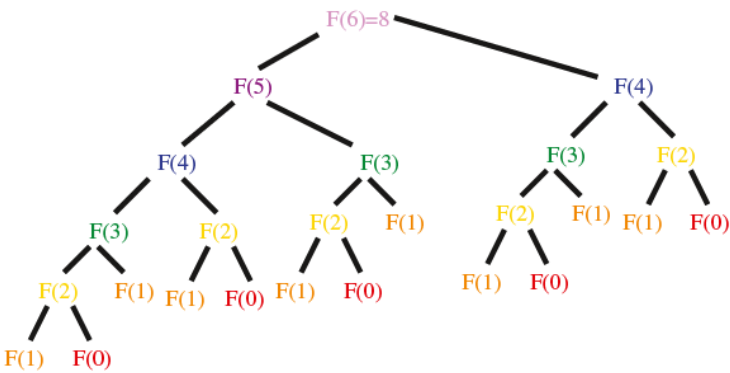


Рис. 10.1. Дерево для рекурсивного вычисления чисел Фибоначчи

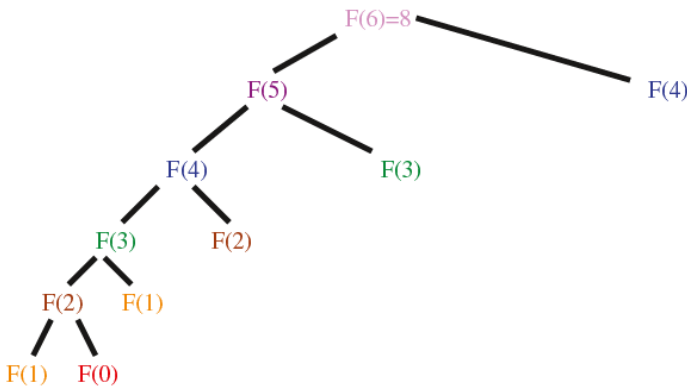


Рис. 10.2. Дерево вычисления чисел Фибоначчи методом кэширования

n / k	0	1	2	3	4	5
0	A					
1	B	G				
2	C	1	H			
3	D	2	3	I		
4	E	4	5	6	J	
5	F	7	8	9	10	K

a

n / k	0	1	2	3	4	5
0	1					
1	1	1				
2	1	2	1			
3	1	3	3	1		
4	1	4	6	4	1	
5	1	5	10	10	5	1

б

Рис. 10.3. *a* — порядок вычисления биномиальных коэффициентов в $M[5, 4]$.
Условия инициализации помечены буквами А–К, а рекуррентные вычисления — числами 1–10;
б — содержимое матрицы после оценки



Рис. 10.4. В одной операции редактирования строки над последним символом можно выполнить одно из следующих действий: замену, вставку или удаление

T		y o u - s h o u l d										
P	pos	0	1	2	3	4	5	6	7	8	9	10
:		<u>0</u>	1	2	3	4	5	6	7	8	9	10
t:	1	<u>1</u>	1	2	3	4	5	6	7	8	9	10
h:	2	2	<u>2</u>	2	3	4	5	5	6	7	8	9
o:	3	3	3	<u>2</u>	3	4	5	6	5	6	7	8
u:	4	4	4	3	<u>2</u>	3	4	5	6	5	6	7
-:	5	5	5	4	3	<u>2</u>	3	4	5	6	6	7
s:	6	6	6	5	4	3	<u>2</u>	3	4	5	6	7
h:	7	7	7	6	5	4	3	<u>2</u>	3	4	5	6
a:	8	8	8	7	6	5	4	3	3	<u>4</u>	5	6
l:	9	9	9	8	7	6	5	4	4	4	<u>4</u>	5
t:	10	10	10	9	8	7	6	5	5	5	5	<u>5</u>

Рис. ЦВ-10.5. Пример матрицы стоимости преобразования строки $P = \text{thou shalt}$ в строку $T = \text{you should}$ за пять шагов. Значения на пути оптимального выравнивания выделены подчеркиванием. Синим цветом обозначены вставки, зеленым — удаления, а красным — замены

T		y o u - s h o u l d										
P	pos	0	1	2	3	4	5	6	7	8	9	10
	0	<u>-1</u>	1	1	1	1	1	1	1	1	1	1
t:	1	<u>2</u>	0	0	0	0	0	0	0	0	0	0
h:	2	2	<u>0</u>	0	0	0	0	0	1	1	1	1
o:	3	2	0	<u>0</u>	0	0	0	0	0	1	1	1
u:	4	2	0	2	<u>0</u>	1	1	1	1	0	1	1
-:	5	2	0	2	2	<u>0</u>	1	1	1	1	0	0
s:	6	2	0	2	2	2	<u>0</u>	1	1	1	1	0
h:	7	2	0	2	2	2	2	<u>0</u>	1	1	1	1
a:	8	2	0	2	2	2	2	2	0	<u>0</u>	0	0
l:	9	2	0	2	2	2	2	2	0	0	<u>0</u>	1
t:	10	2	0	2	2	2	2	2	0	0	0	<u>0</u>

Рис. ЦВ-10.6. Родительская матрица для вычисления расстояния редактирования. Значения на пути оптимального выравнивания выделены подчеркиванием. Синим цветом обозначены вставки, зеленым — удаления, а красным — замены

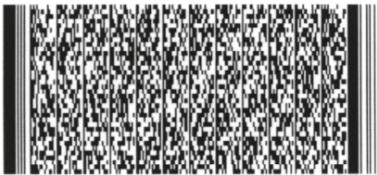


Рис. 10.7. Этикетка с Геттисбергским посланием, закодированном двумерным штрихкодом по технологии PDF-417

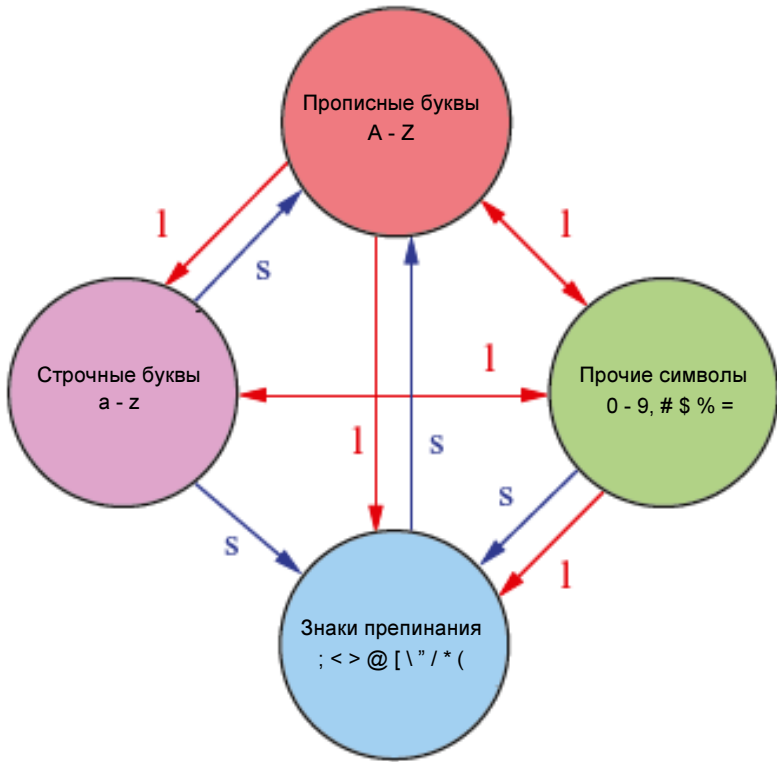


Рис. 10.8. Схема переключения режимов в PDF-417

<i>i</i>	<i>s_i</i>	0	1	2	3	4	5	6	7	8	9	10	11
0	0	T	F	F	F	F	F	F	F	F	F	F	F
1	1	T	T	F	F	F	F	F	F	F	F	F	F
2	2	T	T	T	T	F	F	F	F	F	F	F	F
3	4	T	T	T	T	T	T	T	T	F	F	F	F
4	8	T	T	T	T	T	T	T	T	T	T	T	T

Рис. 10.9. Таблица *sum* для входного множества $S = \{1, 2, 4, 8\}$ и целевой суммы $k = 11$

<i>i</i>	<i>s_i</i>	0	1	2	3	4	5	6	7	8	9	10	11
0	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
1	1	-1	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
2	2	-1	-1	0	1	-1	-1	-1	-1	-1	-1	-1	-1
3	4	-1	-1	-1	-1	0	1	2	3	-1	-1	-1	-1
4	8	-1	-1	-1	-1	-1	-1	-1	0	1	2	3	

Рис. ЦВ-10.10. Массив родительских элементов для решения, равного 11. Красным цветом выделены ячейки, обнаруженные на обратном проходе в процессе восстановления решения

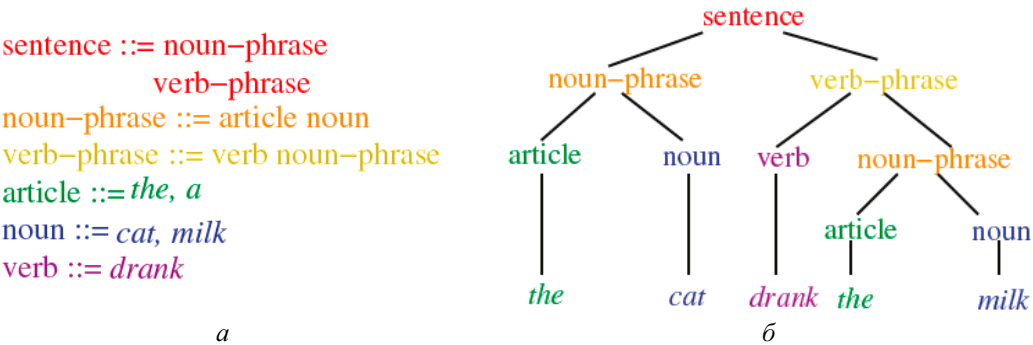
<i>M</i>	<i>k</i>				<i>D</i>	<i>k</i>		
<i>s</i>	1	2	3		<i>s</i>	1	2	3
1	1	1	1		1	—	—	—
1	2	1	1		1	—	1	1
1	3	2	1		1	—	1	2
1	4	2	2		1	—	2	2
1	5	3	2		1	—	2	3
1	6	3	2		1	—	3	4
1	7	4	3		1	—	3	4
1	8	4	3		1	—	4	5
1	9	5	3		1	—	4	6

a

<i>M</i>	<i>k</i>				<i>D</i>	<i>k</i>		
<i>s</i>	1	2	3		<i>s</i>	1	2	3
1	1	1	1		1	—	—	—
2	3	2	2		2	—	1	1
3	6	3	3		3	—	2	2
4	10	6	4		4	—	3	3
5	15	9	6		5	—	3	4
6	21	11	9		6	—	4	5
7	28	15	11		7	—	5	6
8	36	21	15		8	—	5	6
9	45	24	17		9	—	6	7

б

Рис. ЦВ-10.11. Матрицы динамического программирования для двух входных экземпляров задачи упорядоченного разбиения: *a* — разбиение множества (1, 1, 1, 1, 1, 1, 1, 1, 1) на подмножества ((1, 1, 1), (1, 1, 1), (1, 1, 1)) и *б* — множества (1, 2, 3, 4, 5, 6, 7, 8, 9) на подмножества ((1, 2, 3, 4, 5), (6, 7), (8, 9)) (справа). Префиксные суммы выделены красным цветом, а позиции разделителей оптимального решения — синим



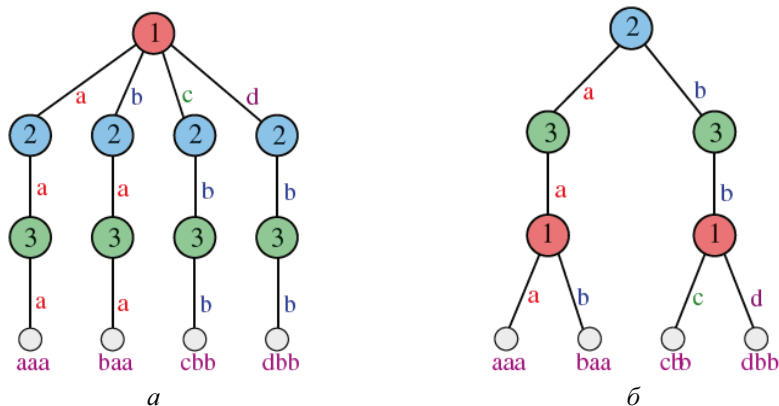


Рис. 10.13. Два разных нагруженных дерева для одного и того же набора правил Prolog, где нагруженное дерево (б) содержит ребер на четыре меньше

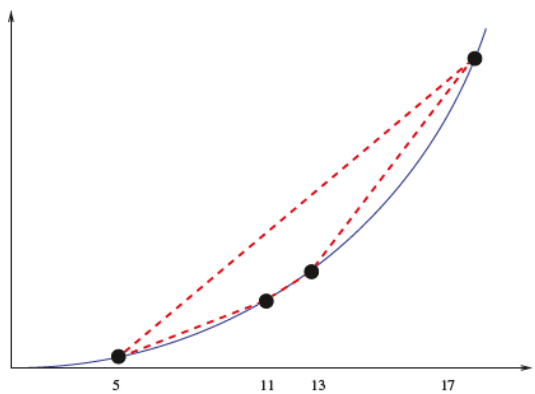


Рис. 11.1. Сведение задачи сортировки к задаче поиска выпуклой оболочки посредством отображения точек на параболу



Рис. 11.2. Часть дерева сводимости для NP-полных задач.
Синие линии обозначают сводимости, рассматриваемые в этой главе

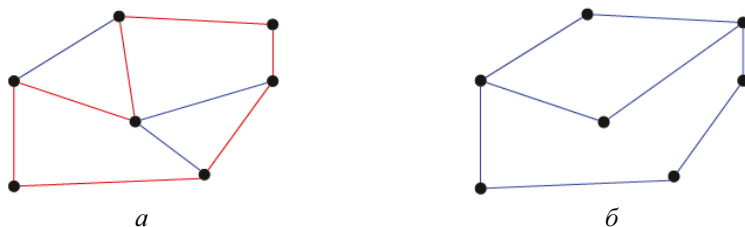


Рис. 11.3. Примеры графов: *a* — граф содержит гамильтонов цикл; *б* — не содержит его

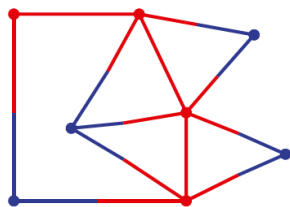


Рис. ЦВ-11.4. Обозначенные красным цветом вершины составляют вершинное покрытие, поэтому остальные вершины должны составлять независимое множество

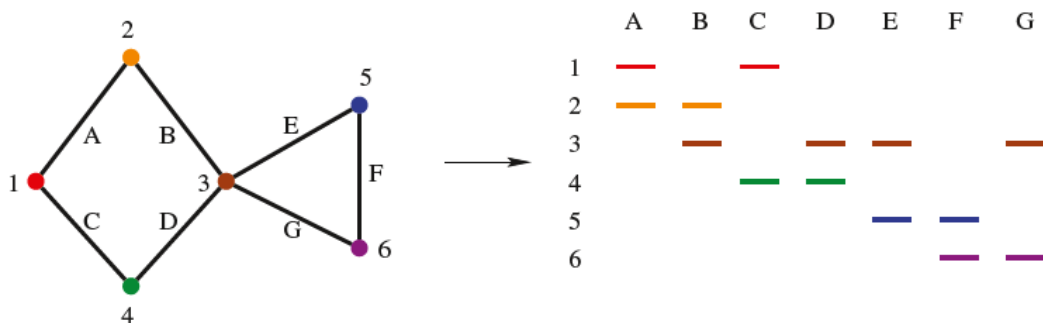


Рис. 11.5. Сведение задачи о независимом множестве к задаче календарного планирования (вершины обозначены цифрами, а ребра — буквами)

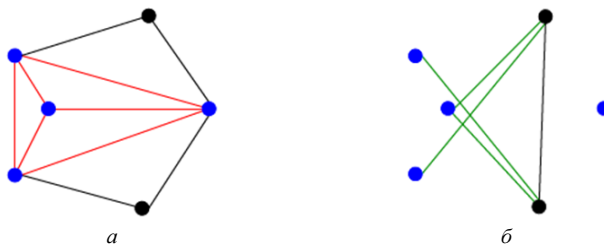


Рис. ЦВ-11.6. *a* — пример небольшого графа с четырехвершинной кликой (выделена синим цветом); *б* — соответствующее независимое множество (выделено черным цветом), создающее двухвершинную клику в дополнении графа

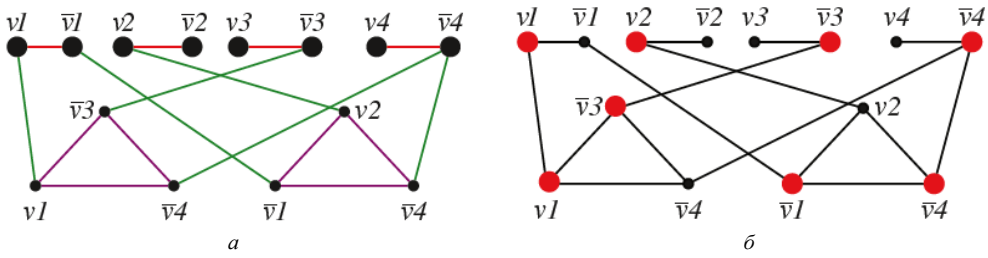


Рис. ЦВ-11.7. *a* — сведение 3-SAT экземпляра $\{\{v_1, \bar{v}_3, \bar{v}_4\}, \{\bar{v}_1, v_2, \bar{v}_4\}\}$ задачи выполнимости к задаче о вершинном покрытии; *б* — красные вершины определяют минимальное вершинное покрытие, следовательно, красные вершины верхних переменных определяют выполняющий набор значений истинности

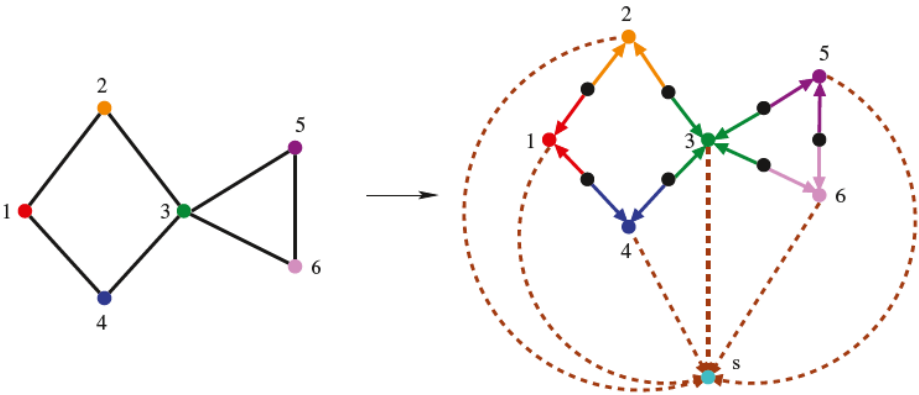


Рис. 11.8. Сведение задачи о вершинном покрытии к задаче односвязного подграфа за счет разделения ребер и добавления узла стока

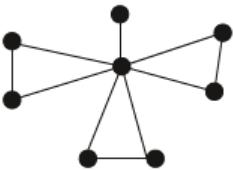


Рис. 11.9. Пример графа для упражнения 12 из главы 11

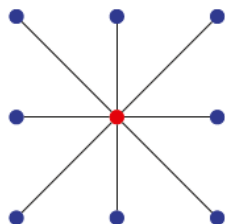


Рис. 12.1. Звездообразный граф

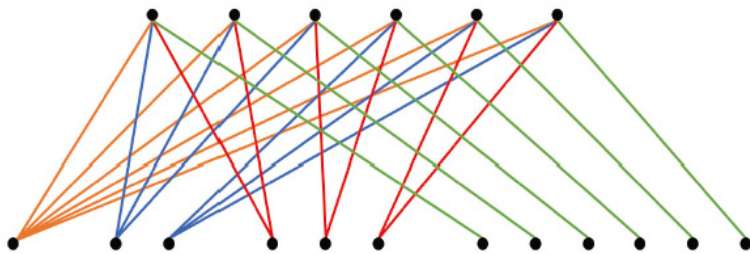
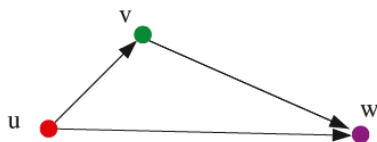
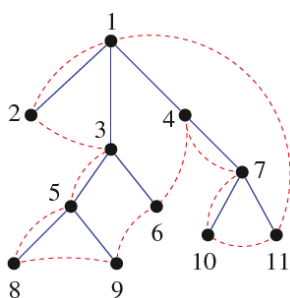
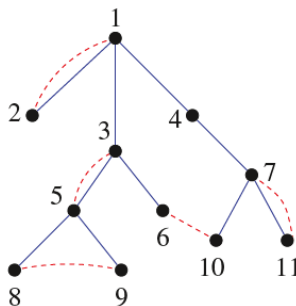


Рис. 12.2. Пример нахождения эвристическим алгоритмом плохого вершинного покрытия

Рис. 12.3. Аксиома треугольника: $d(u, w) \leq d(u, v) + d(v, w)$ — применима для расстояний между точками геометрических фигур

а



б

Рис. 12.4. Обход в глубину остовного дерева по укороченному маршруту (а).

То же самое дерево обхода в глубину при сравнении минимальных весов вершин нечетных степеней, создающем эйлеров граф для эвристического алгоритма Кристофидеса (б)

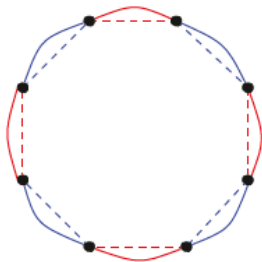


Рис. ЦВ-12.5. Любой маршрут коммивояжера в графе с четным количеством вершин, удовлетворяющем условиям аксиомы треугольника, можно разделить на красные и синие паросочетания, стоимость каждого из которых должна быть не более чем половина стоимости всего маршрута

Промежуточный этап	6			5			4					3			2	1	0
Непокрытые элементы	64	51	40	30	25	22	19	16	13	10	7	4	2	1			
Размер выбранного подмножества	13	11	10	5	3	3	3	3	3	3	3	2	1	1			

Рис. 12.6. Работа «жадного» алгоритма на экземпляре задачи о покрытии множества. Ширина w определяется пятью подмножествами на промежуточном этапе 4, когда количество не входящих в покрытие элементов уменьшится наполовину с не менее чем 2^5-1 до не более чем 2^4

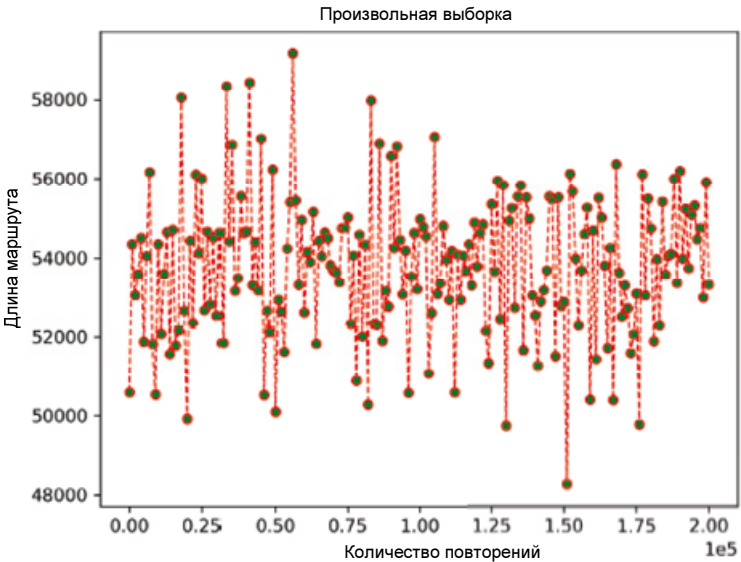


Рис. 12.7. Соотношения между временем поиска решений и их качеством при решении задачи коммивояжера методом произвольной выборки. Процесс нахождения каждого нового наилучшего (здесь — наименьшего) решения идет очень медленно

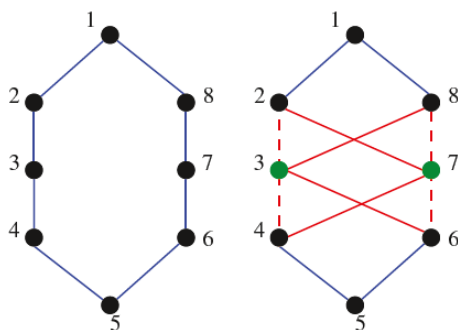


Рис. 12.8. Улучшение маршрута коммивояжера за счет обмена местами вершин 3 и 7 с заменой четырех старых ребер маршрута четырьмя новыми

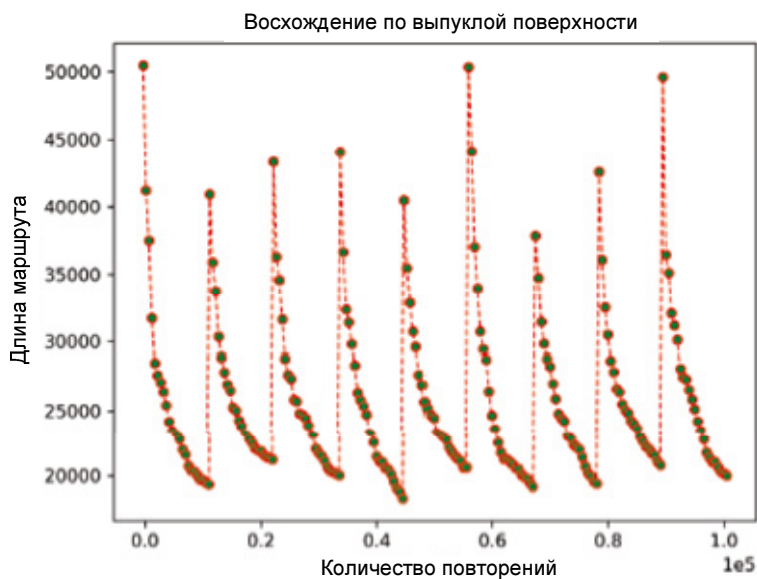


Рис. 12.9. Отношение времени поиска к качеству решений задачи коммивояжера для метода восхождения по выпуклой поверхности

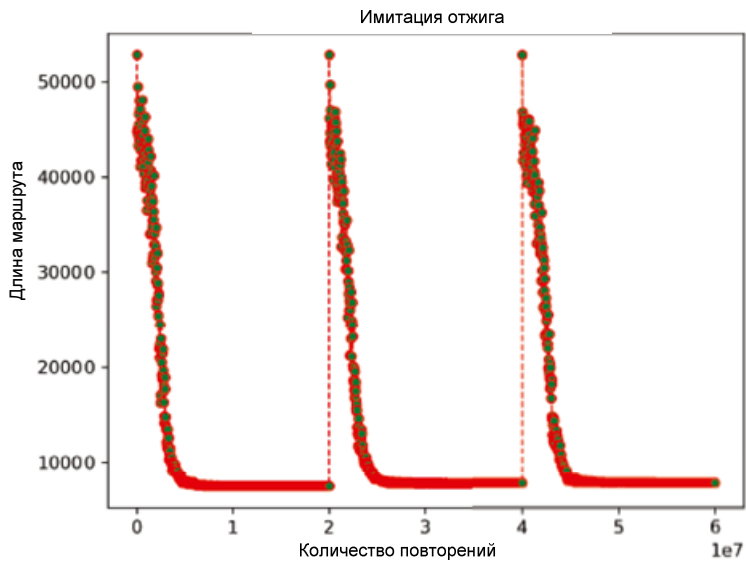


Рис. 12.10. Соотношения между временем поиска решений и их качеством для решения задачи коммивояжера методом имитации отжига

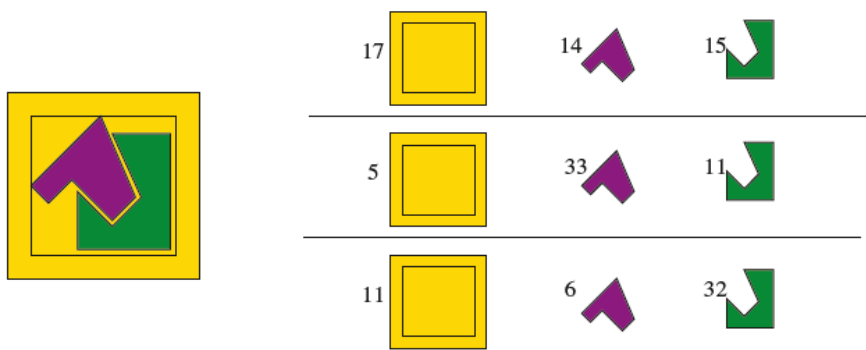


Рис. 12.11. Распределение деталей между тремя устройствами с тем, чтобы сумма отклонений для каждого не превышала 50

Префикс	Суффикс			
	AA	AG	GA	GG
AA	AAAA	AAAG	AAGA	AAGG
AG	AGAA	AGAG	AGGA	AGGG
GA	GAAA	GAAG	GAGA	GAGG
GG	GGAA	GGAG	GGGA	GGGG

Рис. 12.12. Массив всех пуриновых 4-меров

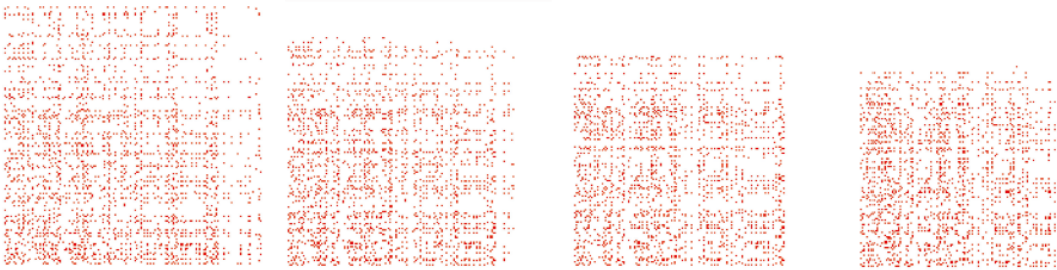


Рис. 12.13. Сжатие массива ВИЧ методом имитации отжига после 0, 500, 1 000 и 5 750 итераций

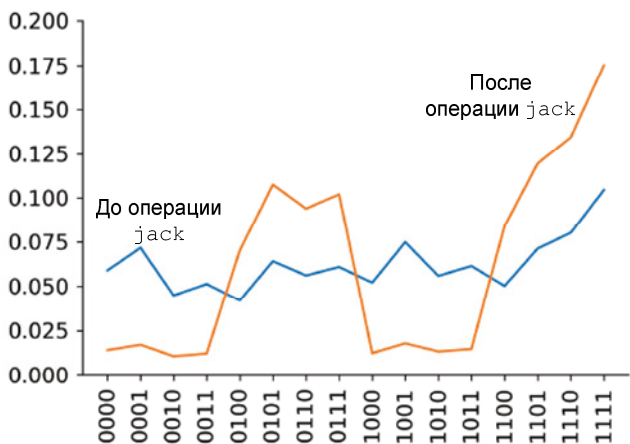


Рис. 12.14. Повышение вероятности всех состояний, где кубит $q_2 = 1$

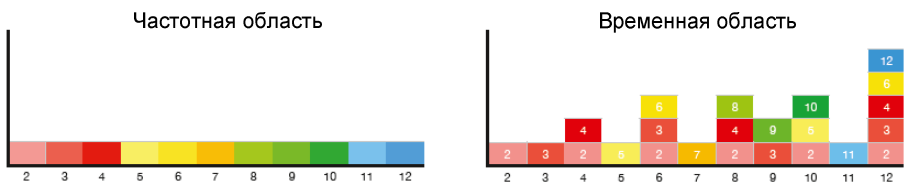


Рис. 12.15. Преобразование целых чисел из частотной области во временную
вычисляет количество множителей для каждого целого числа

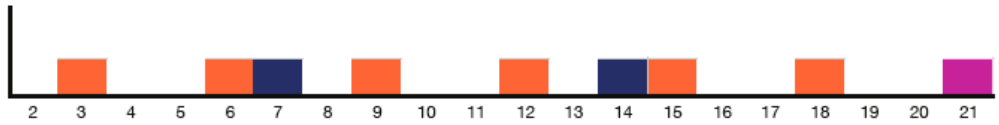


Рис. 12.16. Маловероятно, что множитель можно получить напрямую выборкой из чисел, кратных множителям значения M (здесь 21), но он почти наверняка будет получен из наибольшего общего делителя

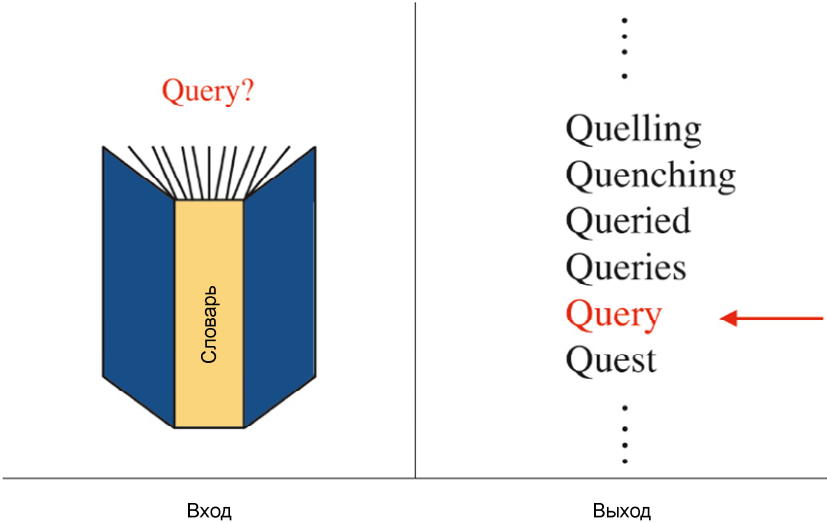


Рис. 15.1. Поиск в словаре

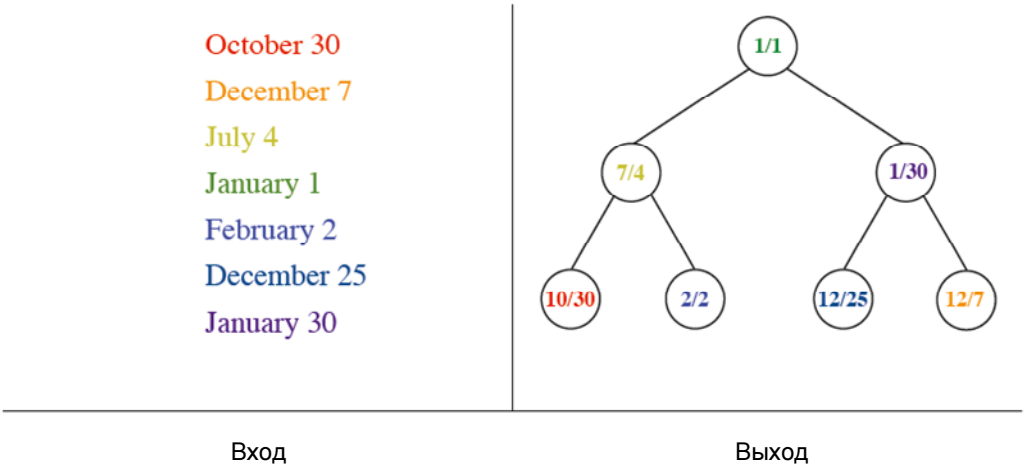
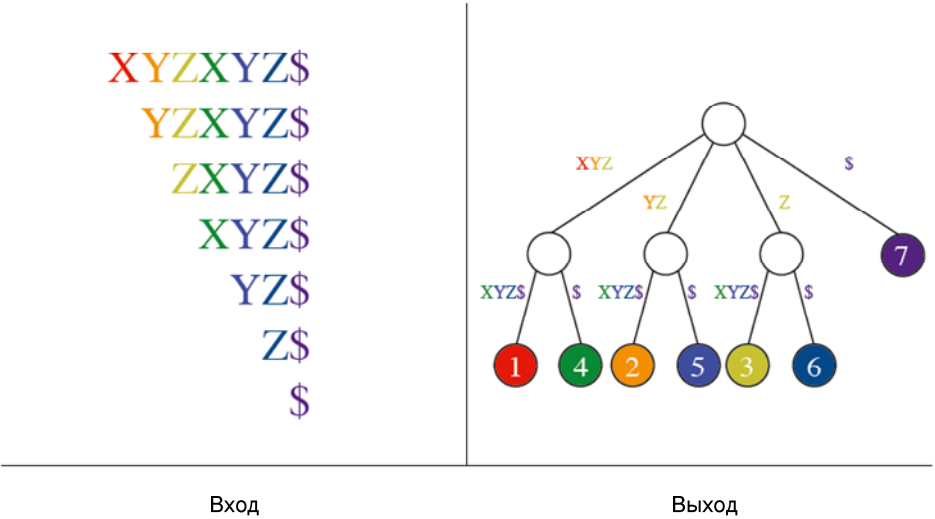
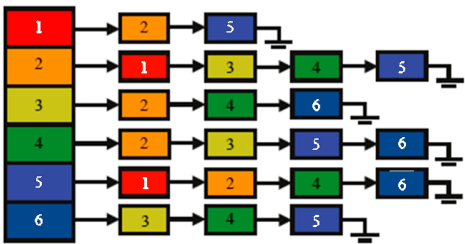
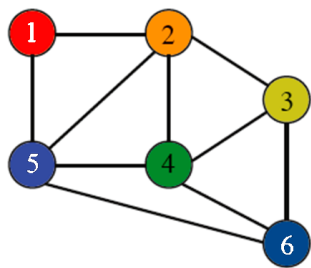


Рис. 15.2. Очередь с приоритетами

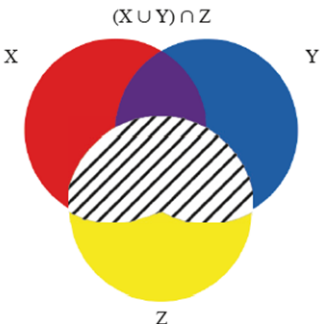
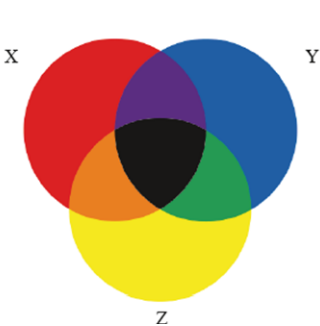




Вход

Выход

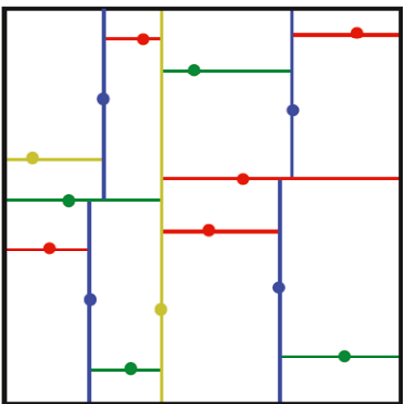
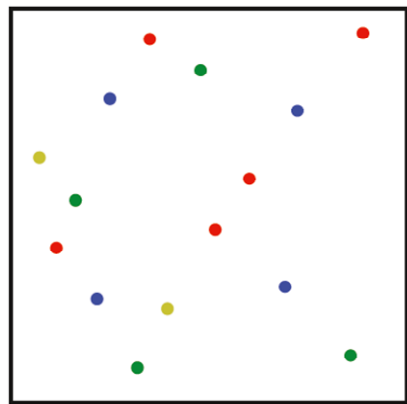
Рис. 15.5. Варианты представления графа



Вход

Выход

Рис. 15.6. Множества X, Y, Z



Вход

Выход

Рис. 15.7. Создание kd-дерева

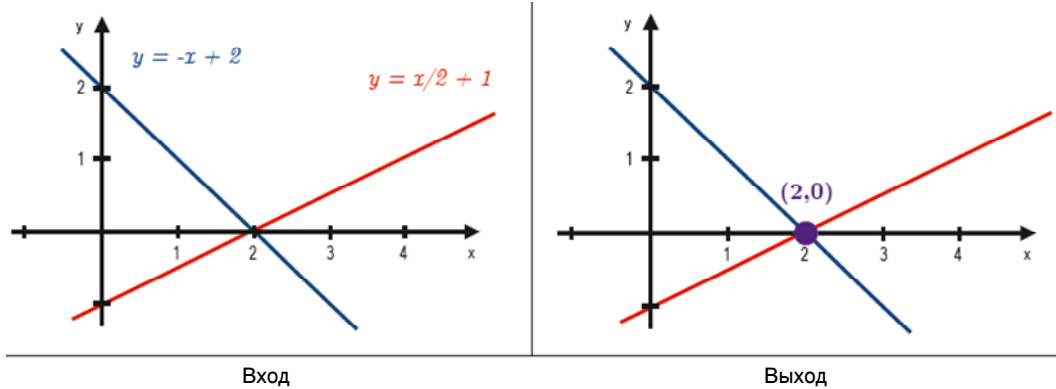


Рис. 16.1. Решение системы линейных уравнений

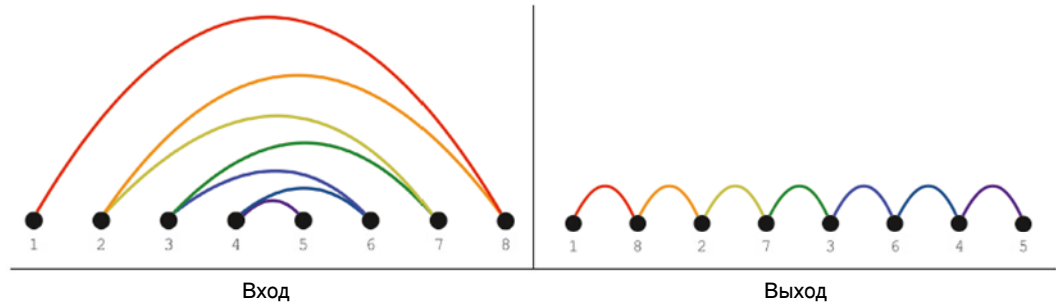


Рис. 16.2. Уменьшение ширины ленты матрицы

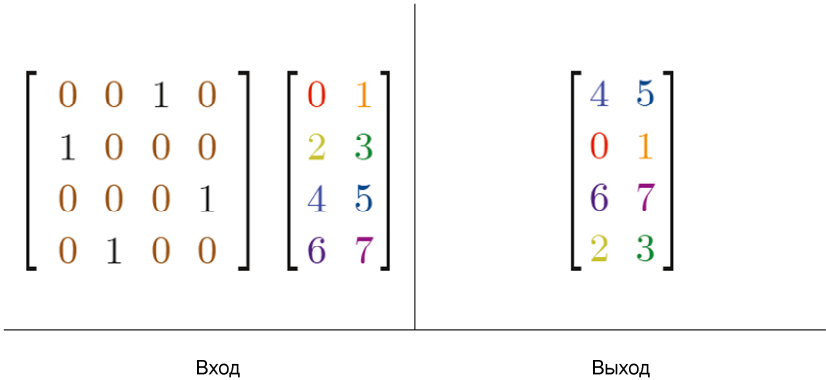


Рис. 16.3. Умножение матриц

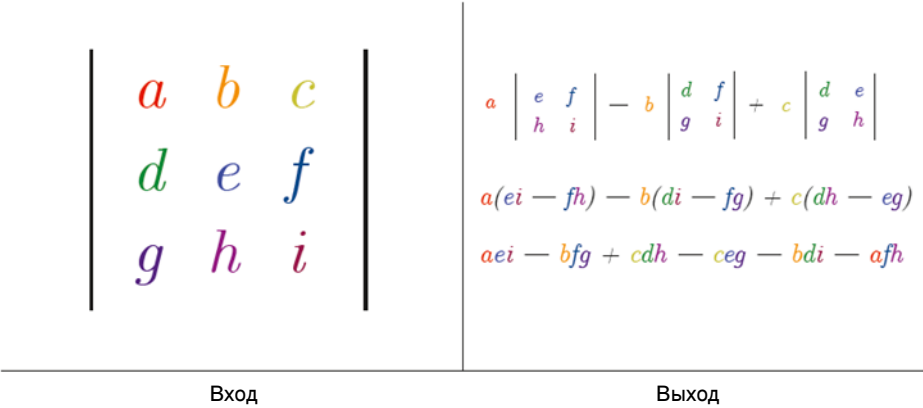


Рис. 16.4. Вычисление определителя матрицы

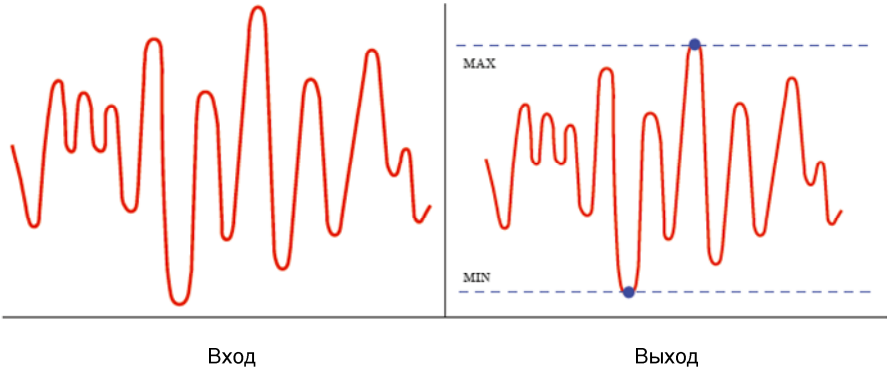


Рис. 16.5. Поиск минимума и максимума функции

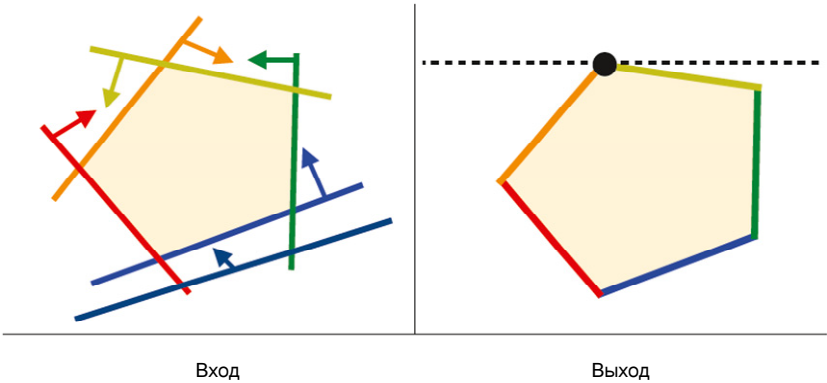


Рис. 16.6. Графическое представление задачи линейного программирования

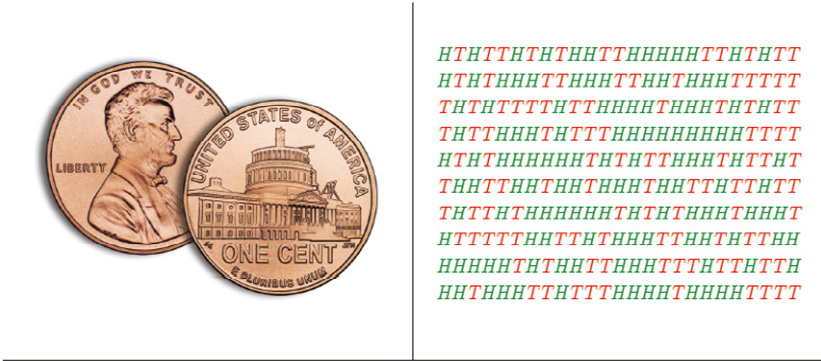


Рис. 16.7. Последовательность случайных символов

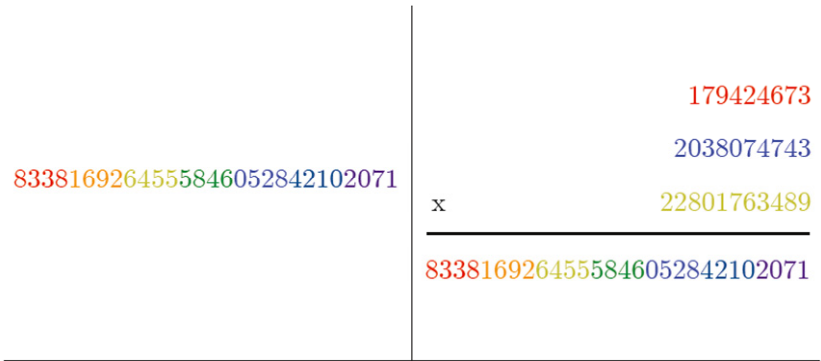


Рис. 16.8. Разложение на множители



Рис. 16.9. Деление очень больших целых чисел

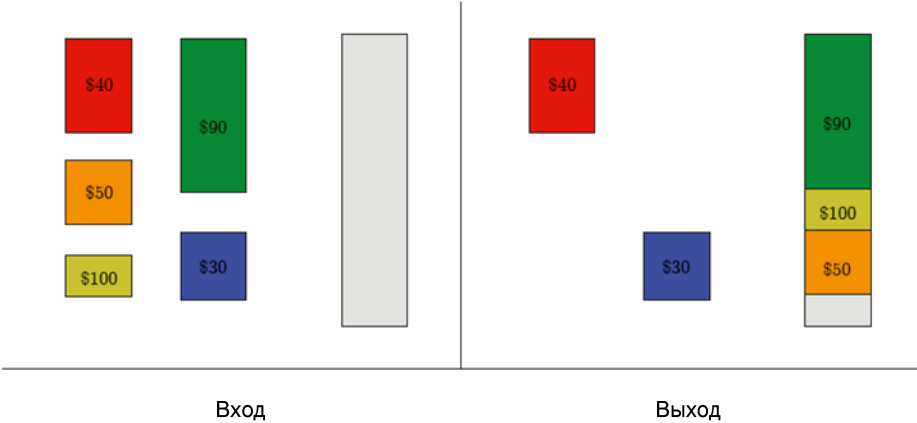


Рис. 16.10. Задача о рюкзаке

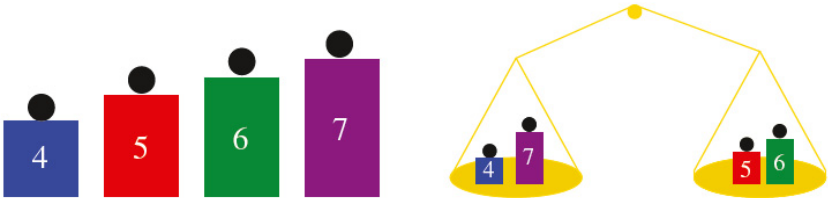


Рис. 16.11. Задача разбиения множества целых чисел — частный случай задачи о рюкзаке

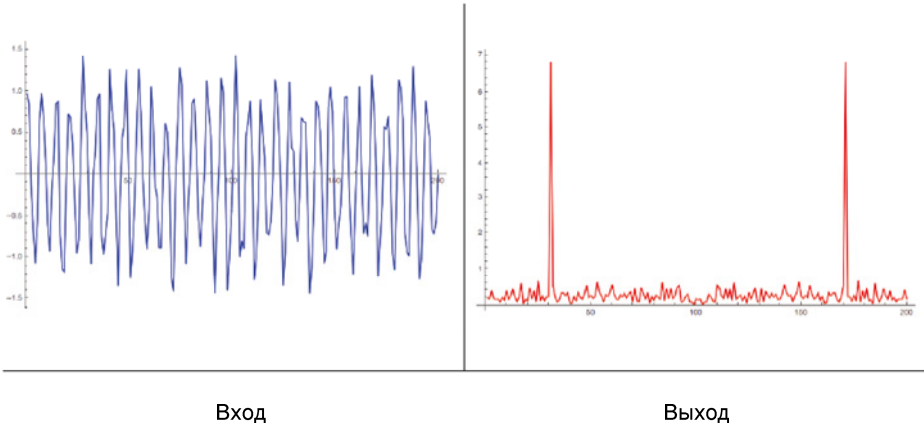


Рис. 16.12. Пример преобразования Фурье

ГЛАВА 17

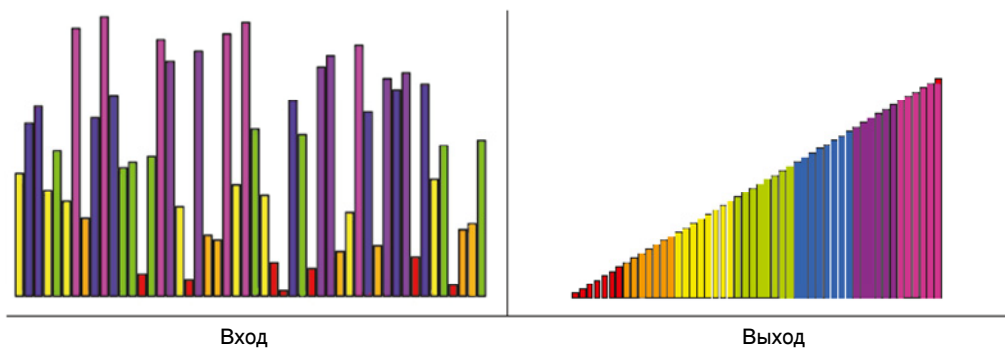


Рис. 17.1. Сортировка

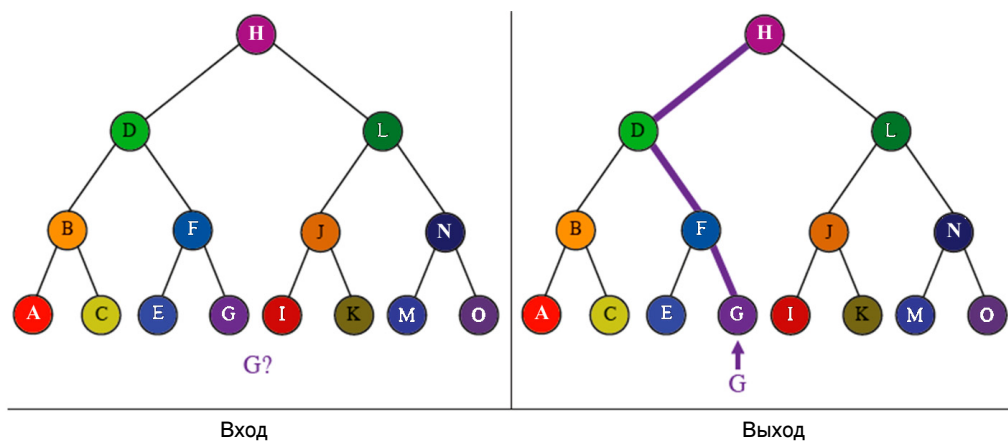


Рис. 17.2. Поиск в наборе ключей

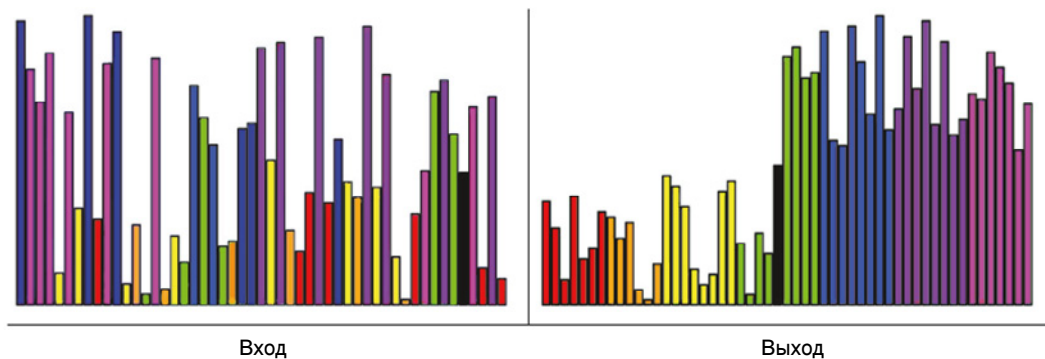


Рис. 17.3. Сортировка по среднему значению

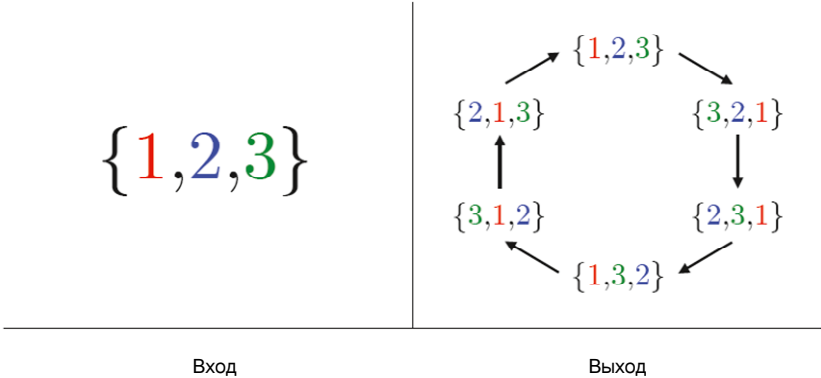


Рис. 17.4. Перестановки

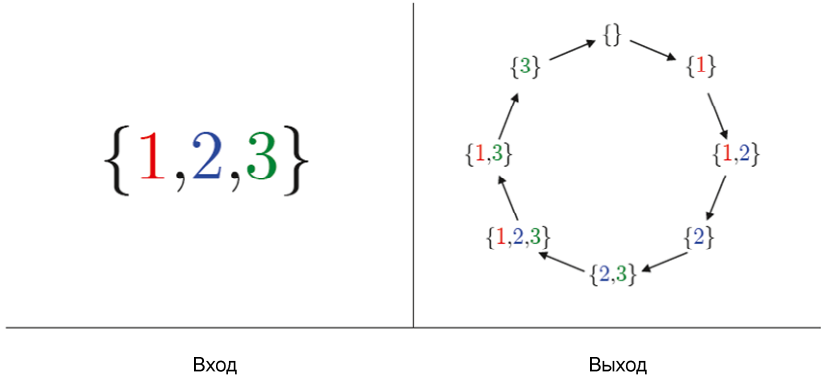


Рис. 17.5. Подмножества

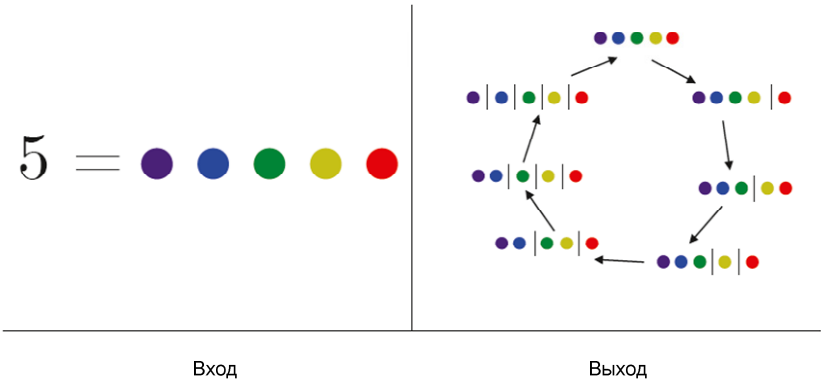


Рис. 17.6. Разбиения

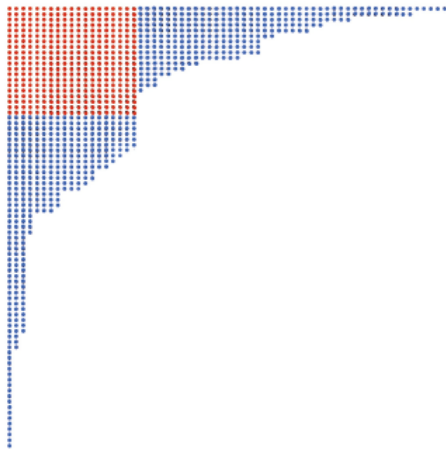


Рис. 17.7. Диаграмма Феррерса для случайного разбиения числа $n = 1000$

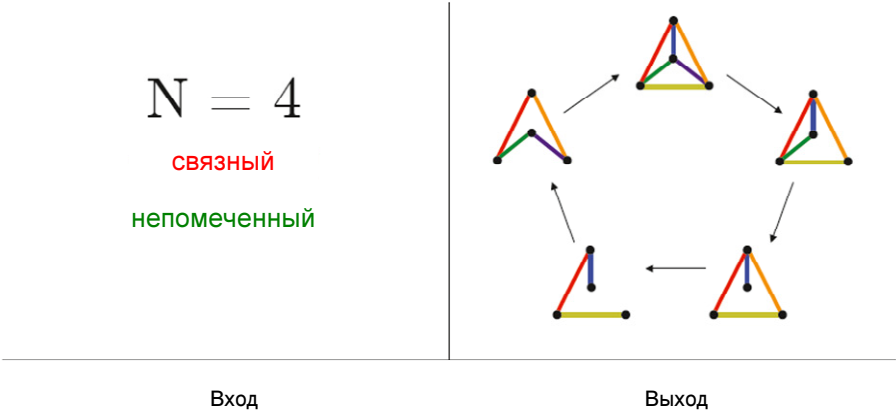


Рис. 17.8. Генерирование графов

December 21, 2012? (Gregorian)	5773 Teveth 8 (Hebrew) 1434 Safar 7 (Islamic) 1934 Agrahayana 30 (Indian Civil) 13.0.0.0 (Mayan Long Count)
Вход	Выход

Рис. 17.9. Календарные вычисления

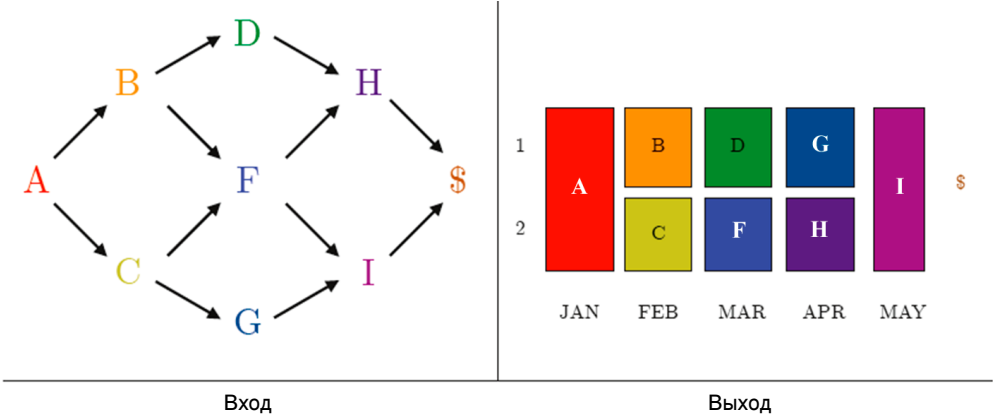


Рис. 17.10. Задача календарного планирования

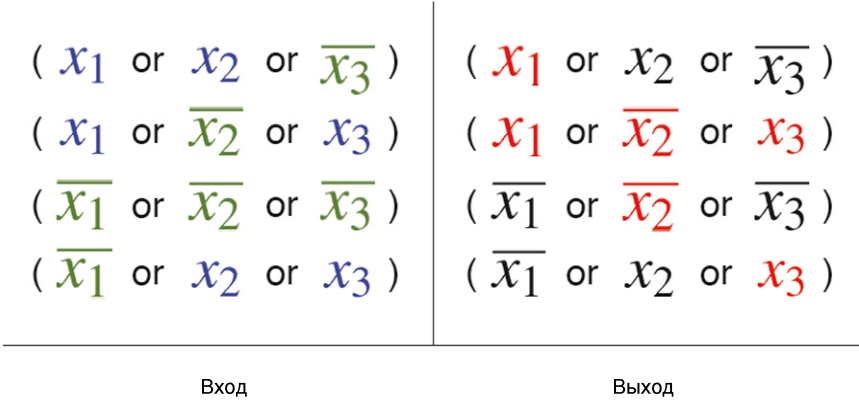


Рис. ЦВ-17.11. Задача выполнимости

ГЛАВА 18

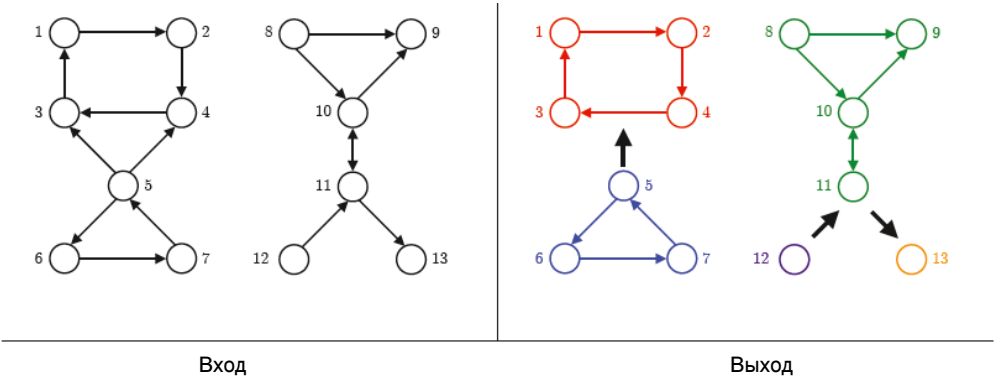


Рис. 18.1. Компоненты связности

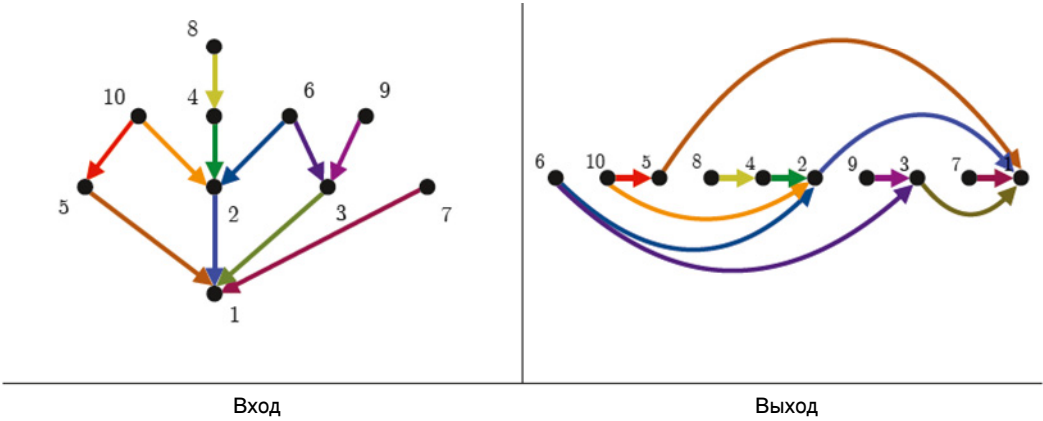


Рис. 18.2. Топологическая сортировка

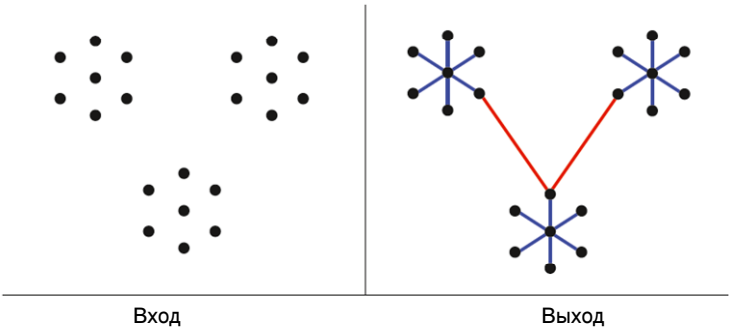


Рис. 18.3. Поиск минимального остовного дерева

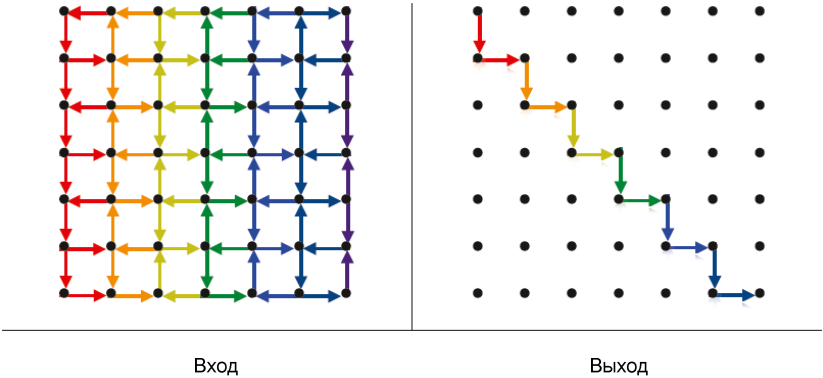


Рис. 18.4. Поиск кратчайшего пути

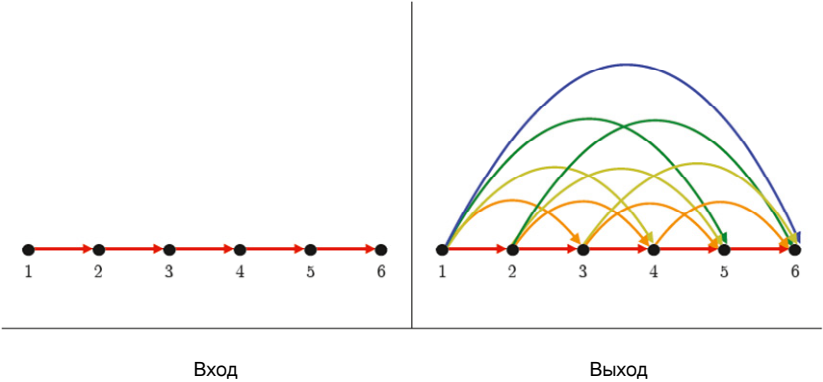


Рис. 18.5. Транзитивное замыкание

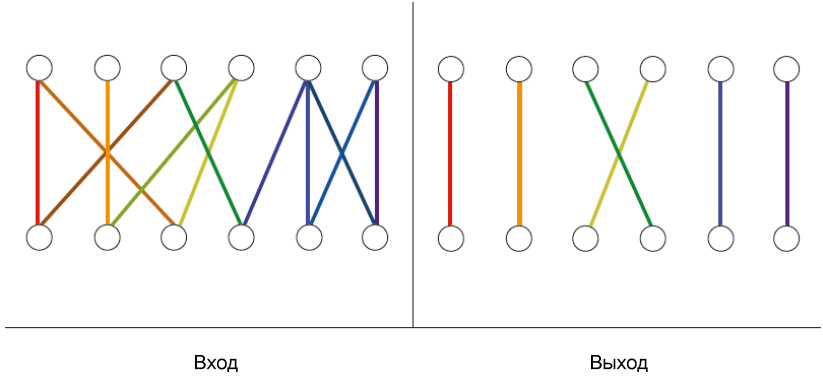


Рис. 18.6. Поиск паросочетаний

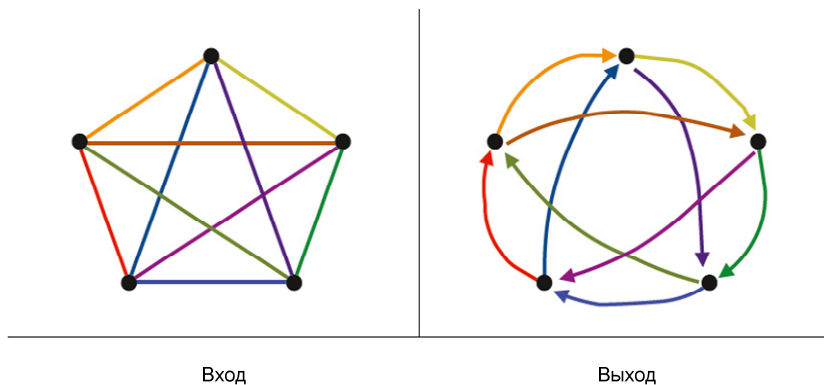


Рис. 18.7. Эйлеров цикл

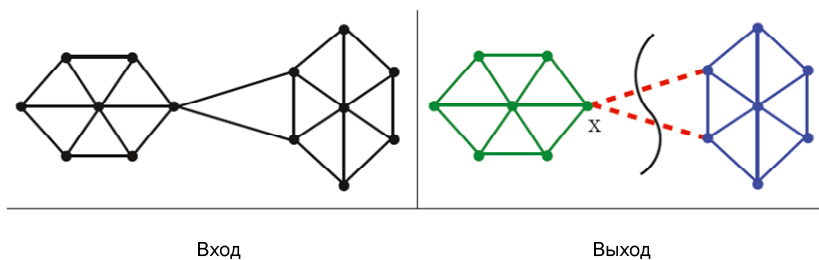


Рис. 18.8. Выяснение числа реберной связности

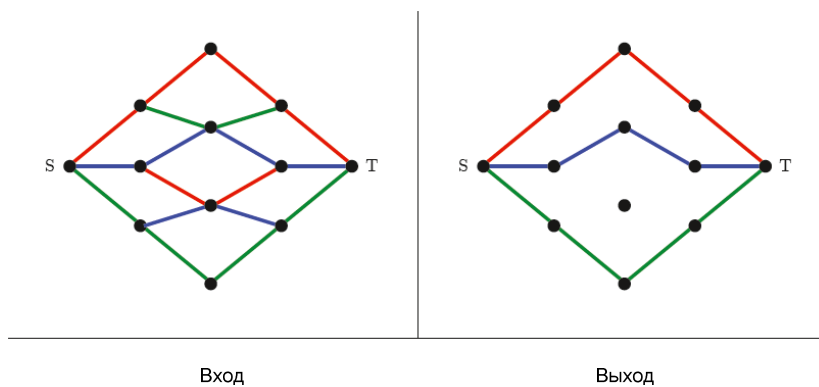


Рис. 18.9. Потoki в сети

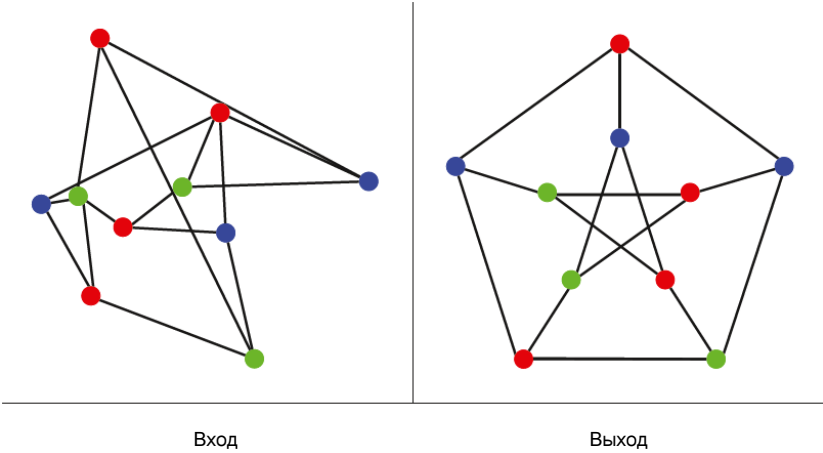


Рис. 18.10. Моделирование графа в виде системы пружин

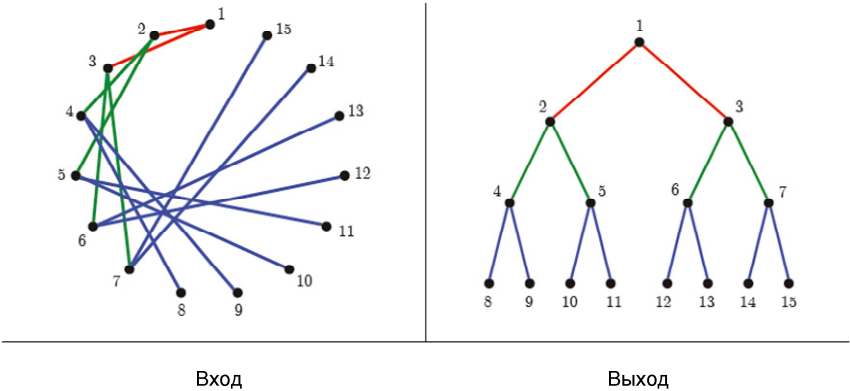


Рис. 18.11. Упорядоченная укладка сбалансированного двоичного дерева

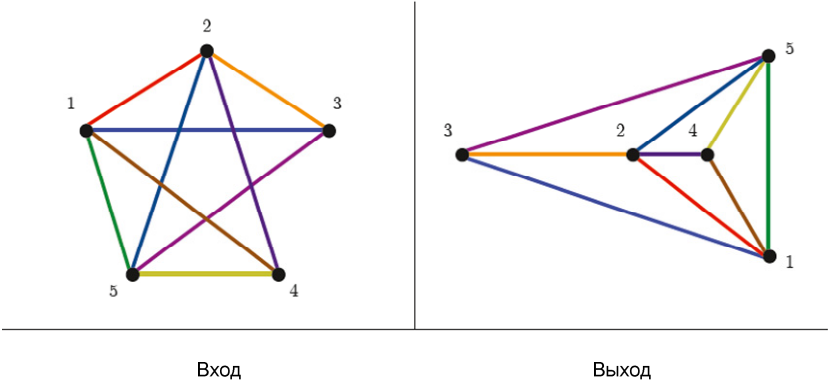


Рис. 18.12. Рисунок графа на плоскости

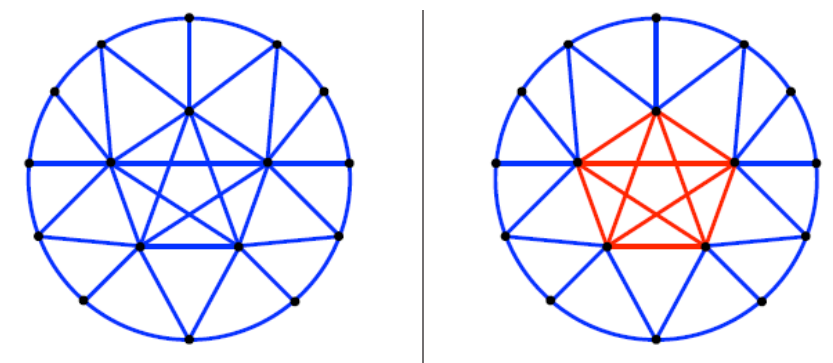
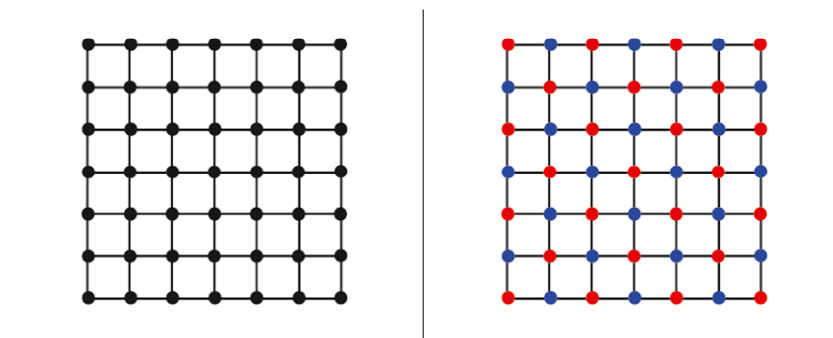


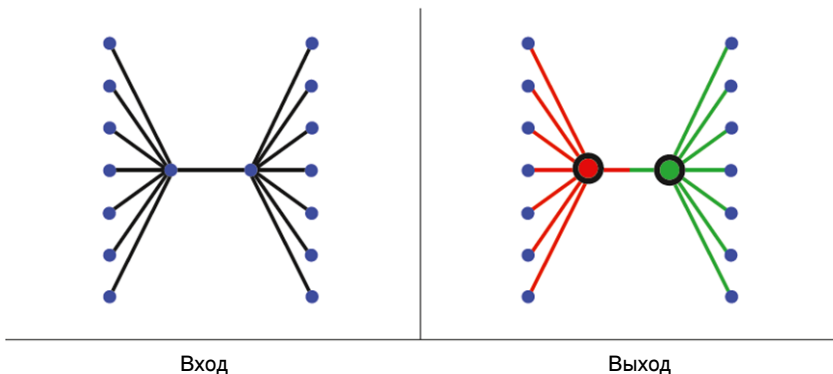
Рис. ЦВ-19.1. Клика



Вход

Выход

Рис. ЦВ-19.2. Независимое множество



Вход

Выход

Рис. 19.3. Вершинное покрытие

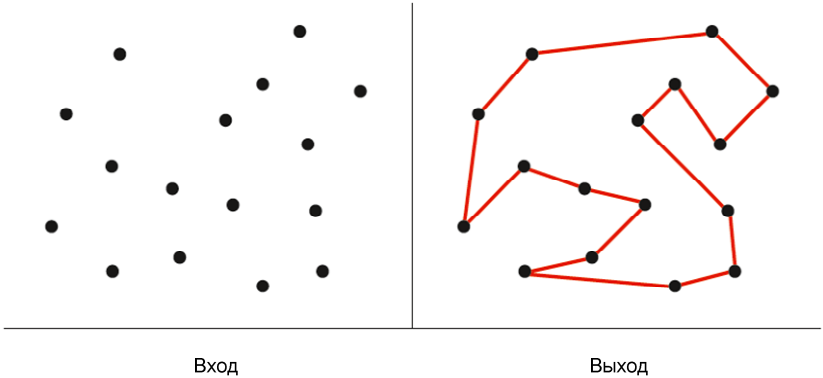


Рис. 19.4. Маршрут коммивояжера

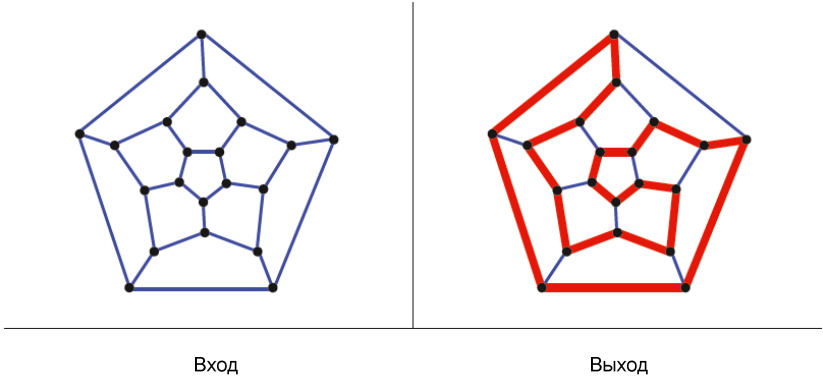


Рис. 19.5. Гамильтонов цикл

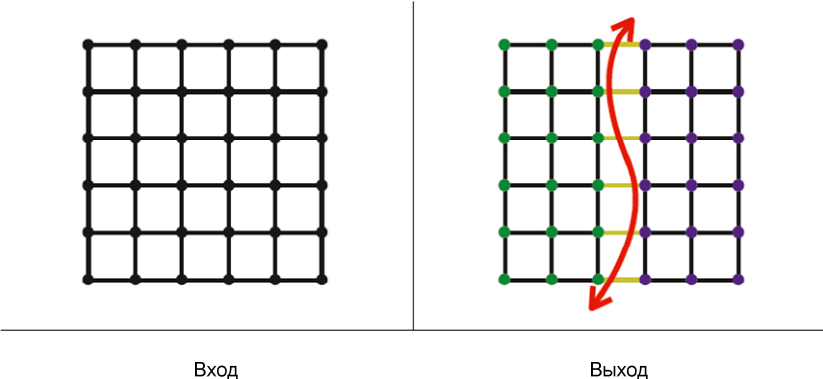


Рис. 19.6. Разбиение графа

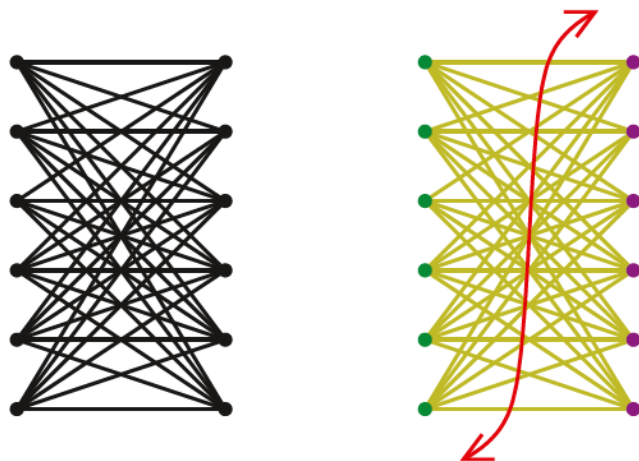


Рис. 19.7. Максимальный разрез двудольного графа разрезает все ребра

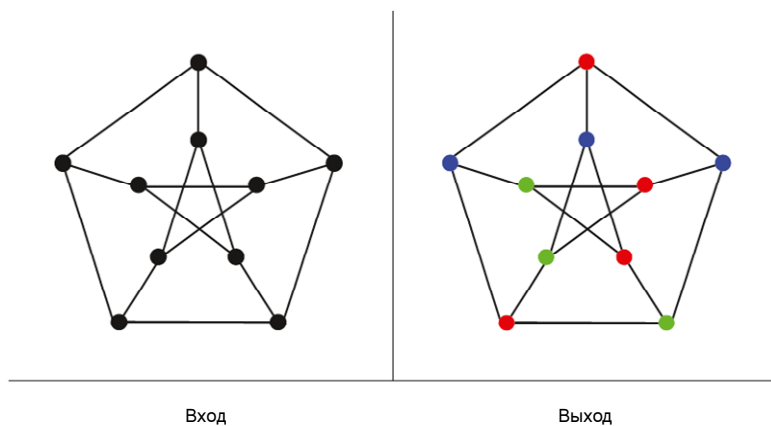


Рис. ЦВ-19.8. Вершинная раскраска

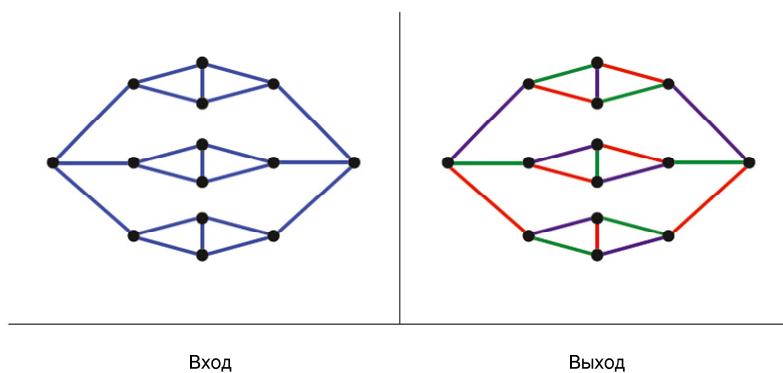


Рис. ЦВ-19.9. Реберная раскраска

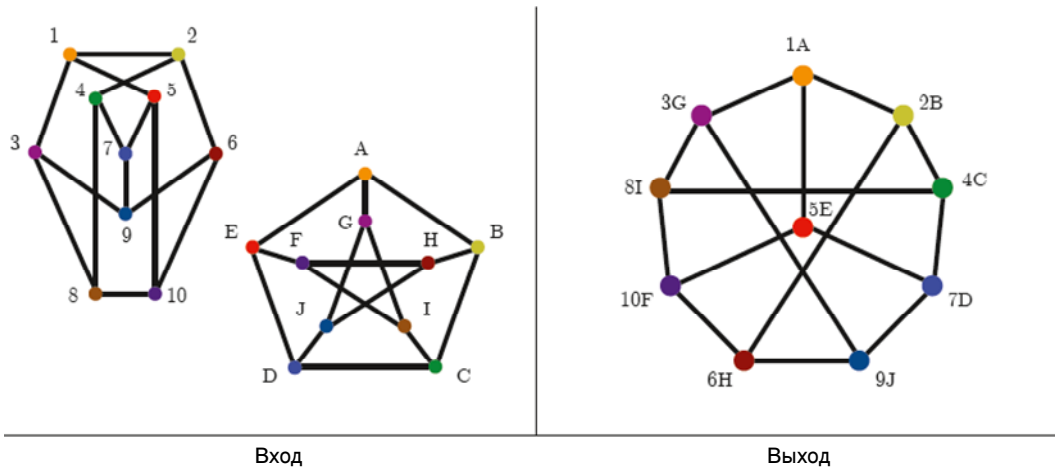


Рис. 19.10. Изоморфизм графов

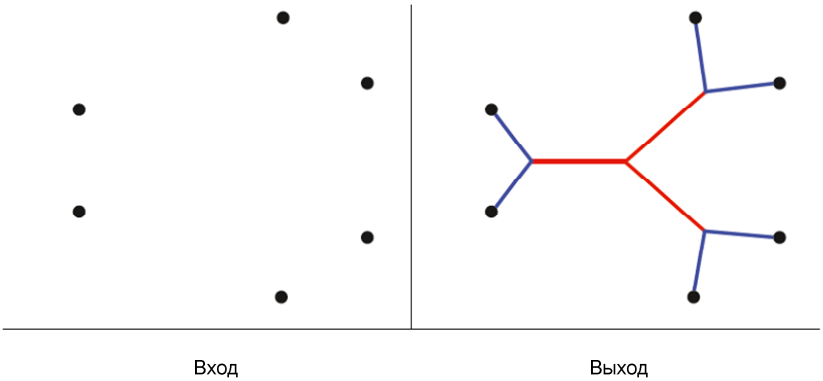


Рис. 19.11. Дерево Штейнера

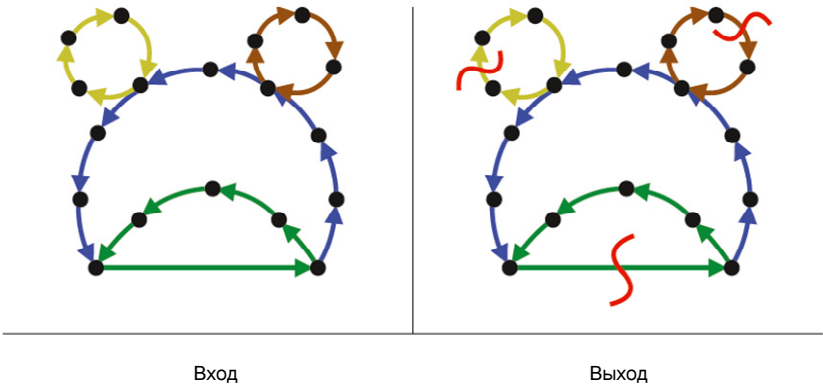


Рис. 19.12. Разрывающее множество ребер

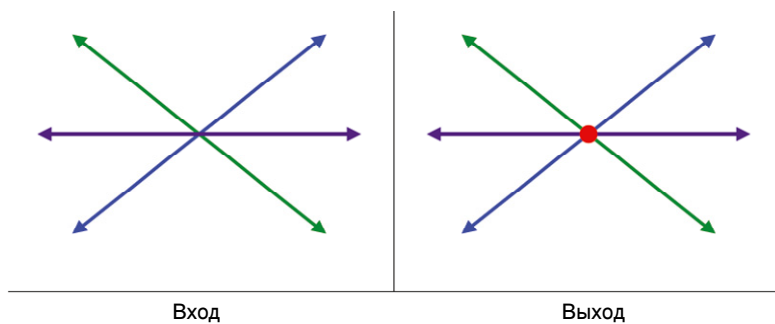


Рис. 20.1. Пересечение прямых

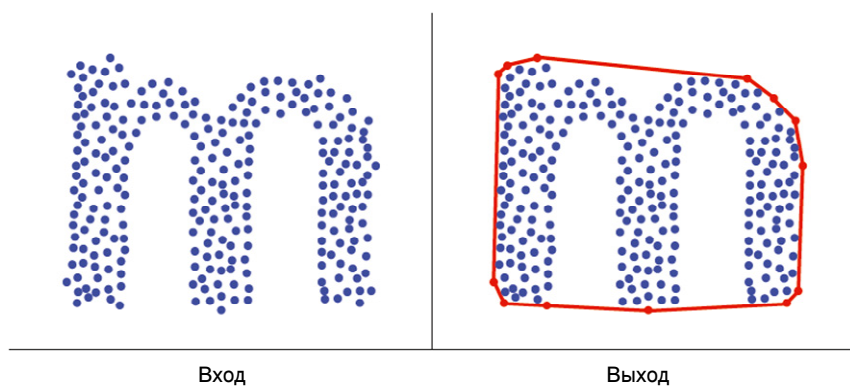


Рис. 20.2. Выпуклая оболочка

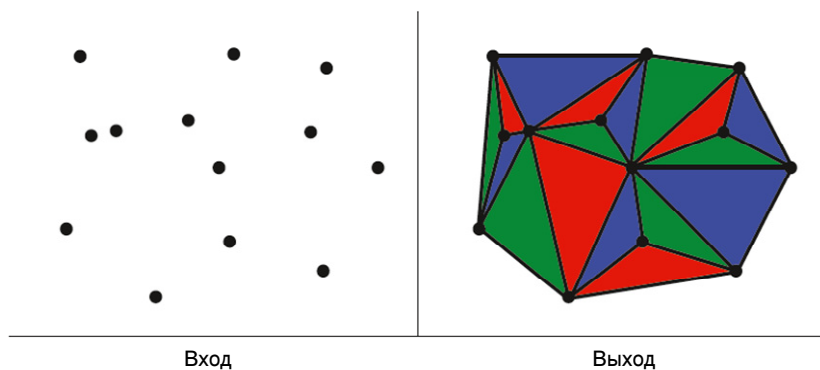


Рис. 20.3. Триангуляция

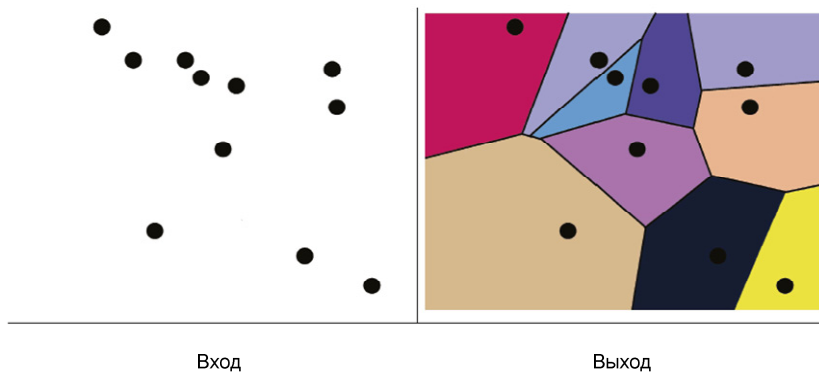


Рис. 20.4. Диаграмма Вороного

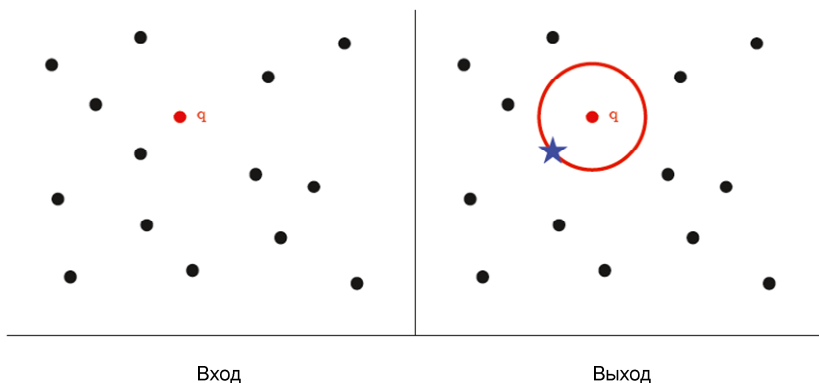


Рис. 20.5. Поиск ближайшей точки

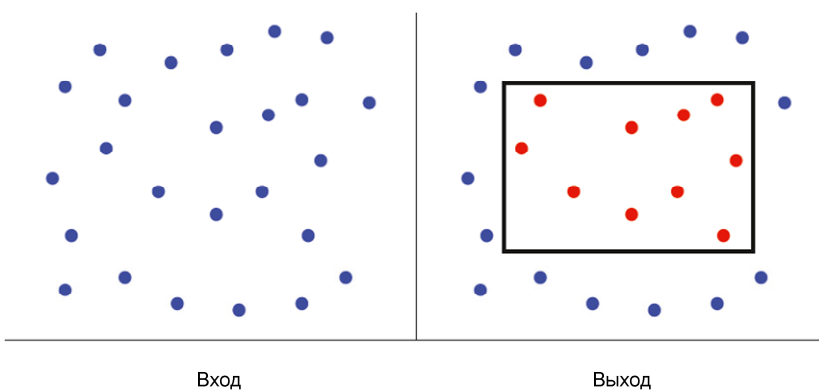
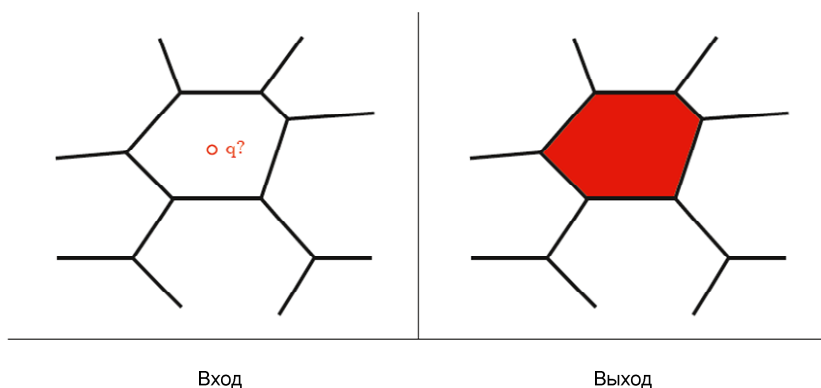
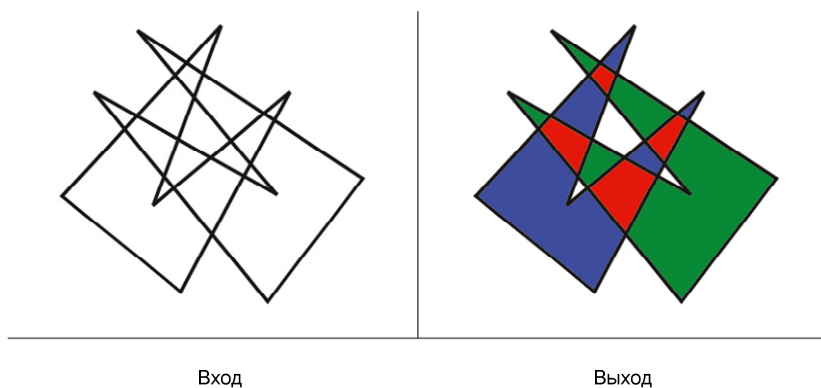
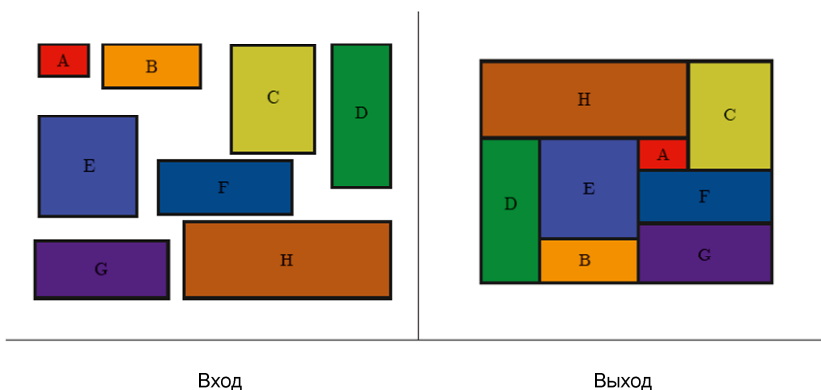


Рис. 20.6. Поиск в области

**Рис. 20.7.** Выяснение местоположения точки**Рис. 20.8.** Выявление пересечений**Рис. 20.9.** Разложение по контейнерам

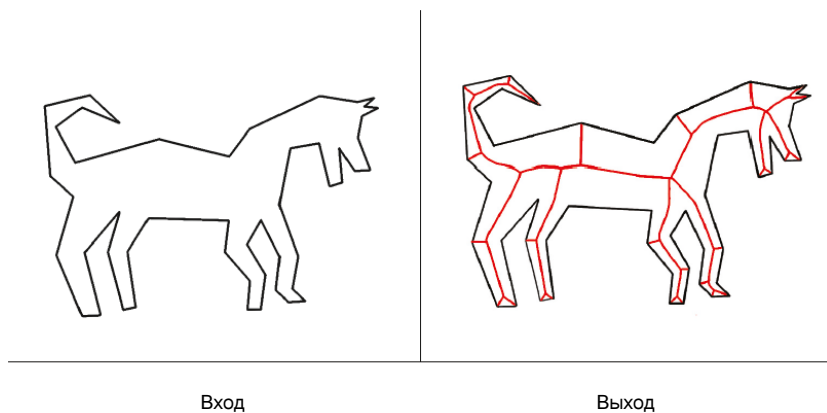


Рис. 20.10. Преобразование к срединной оси

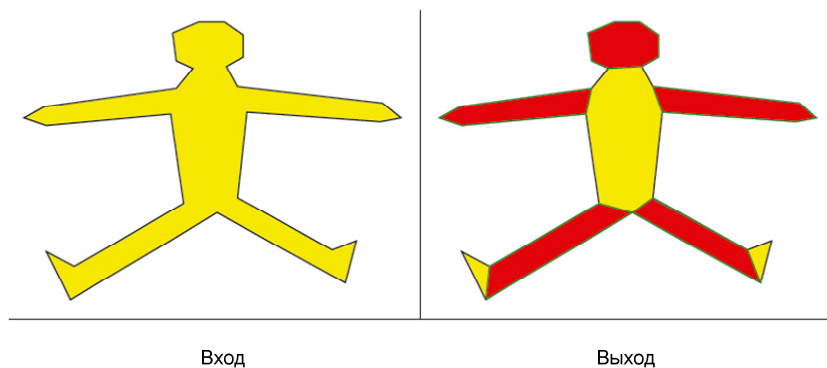


Рис. 20.11. Разбиение многоугольника на части

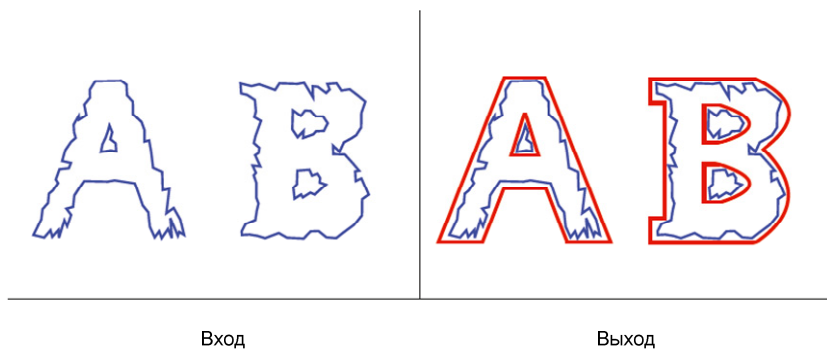


Рис. 20.12. Упрощение многоугольников

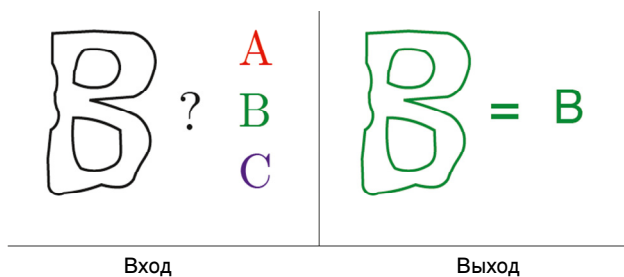


Рис. 20.13. Выявление сходства фигур

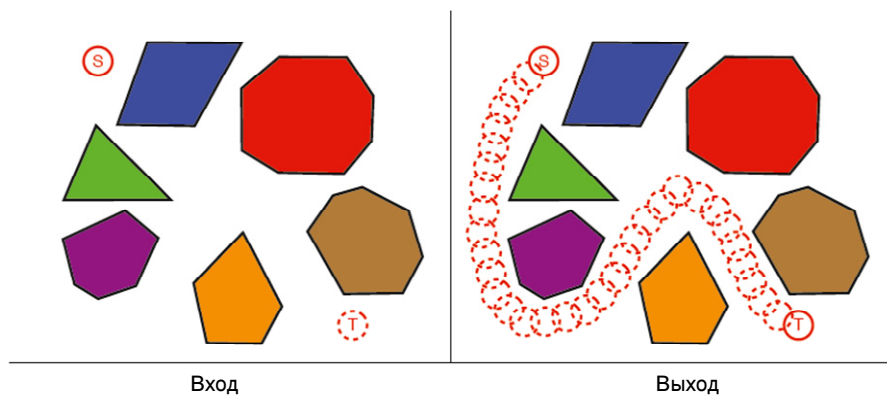


Рис. 20.14. Планирование перемещений

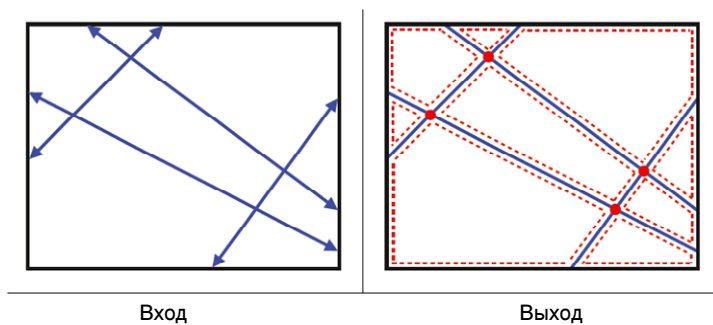


Рис. 20.15. Разбиение плоскости

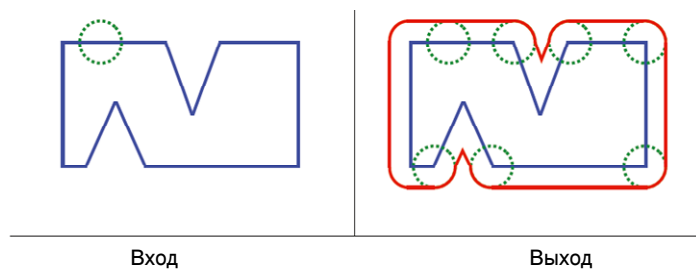


Рис. 20.16. Сумма Минковского

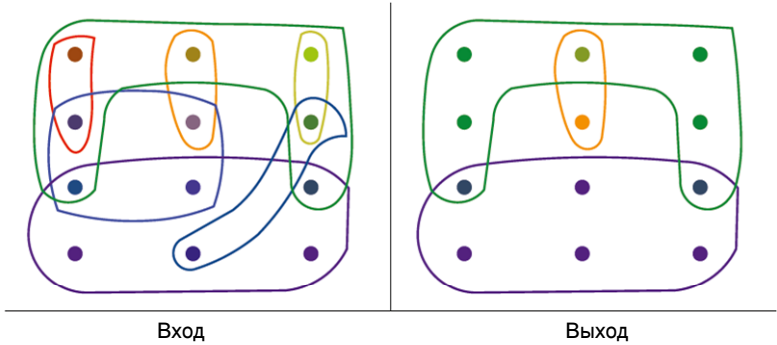


Рис. 21.1. Покрытие множества

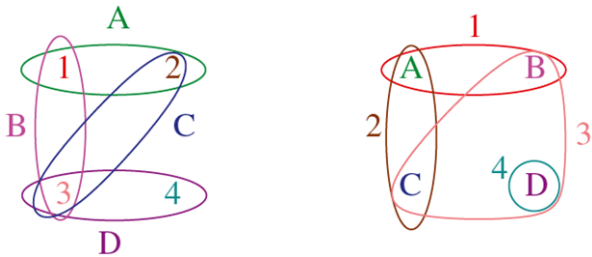


Рис. 21.2. Примеры оптимальных решений: экземпляра задачи о минимальном множестве представителей (слева) и экземпляра двойственной задачи о покрытии множества (справа)

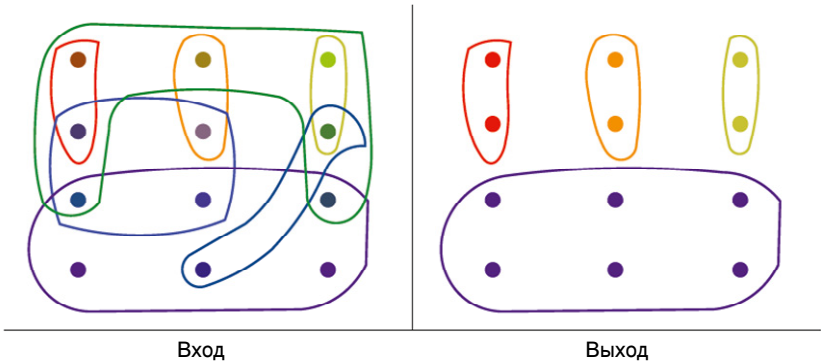


Рис. 21.3. Пример укладки множества



Рис. ЦВ-21.4. Поиск в строке по образцу

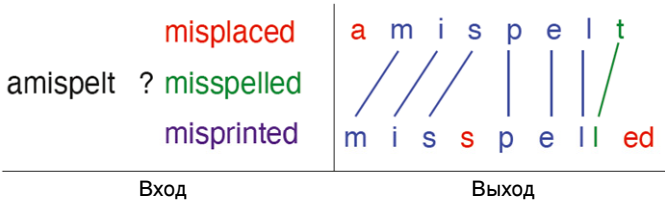


Рис. 21.5. Нечеткое сравнение строк

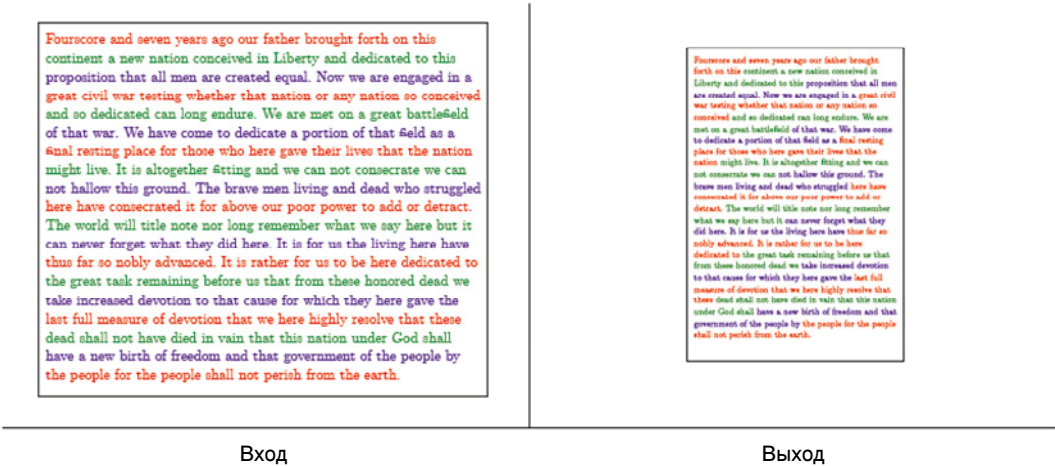


Рис. 21.6. Сжатие текста

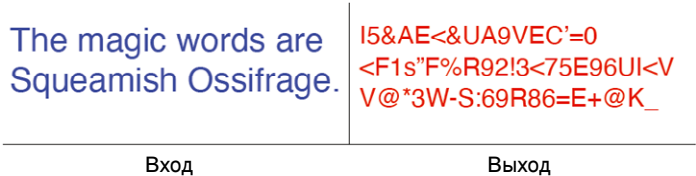


Рис. 21.7. Пример шифрования текста

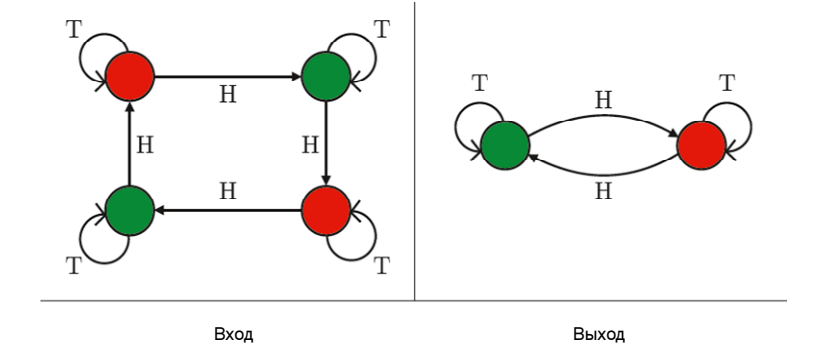


Рис. 21.8. Пример минимизации конечного автомата

АНААІGDDЕТNWORDTS	АНААІGDDЕТNWORDTS
HGITGDDEANWTOSRDSG	HGITGDDEANWTOSRDSG
GTASHIDDENWORDTGAG	GTASHIDDENWORDTGAG
HIGSDDEGNWORGADSTA	HIGSDDEGNWORGADSTA
HAIDAGDENSWORSADTS	HAIDAGDENSWORSADTS
Вход	Выход

Рис. ЦВ-21.9. Поиск максимальной общей подстроки

А В R A C A	<u>А В R A C A D A B R A</u>
R A C A D A	А B R A C A
A C A D A B	R A C A D A
C A D A B R	A C A D A B
A D A B R A	C A D A B R
Вход	Выход

Рис. 21.10. Поиск минимальной общей надстроки