

Глава 3. Обходы графов

Существует ряд задач на графах, в которых требуется найти маршрут, который содержит все вершины или ребра графа – **обход**.

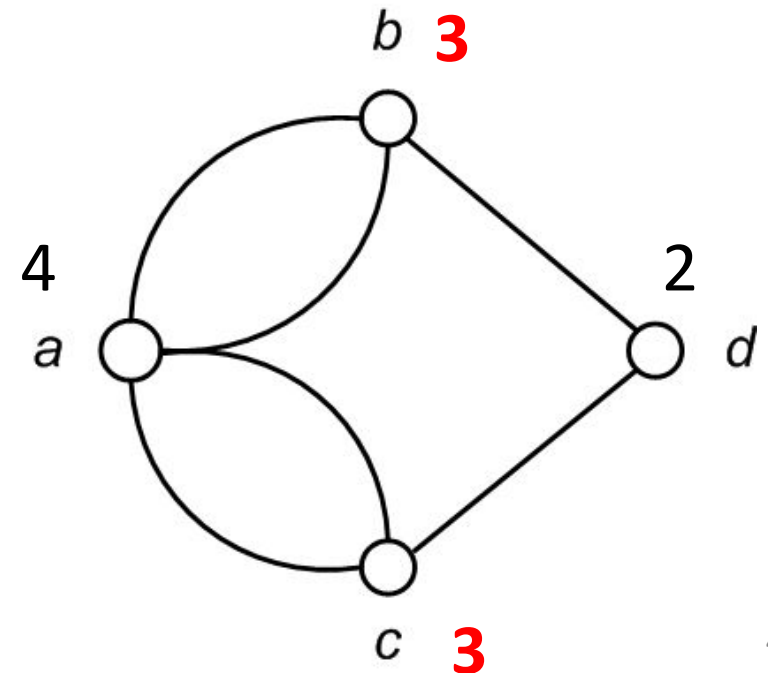
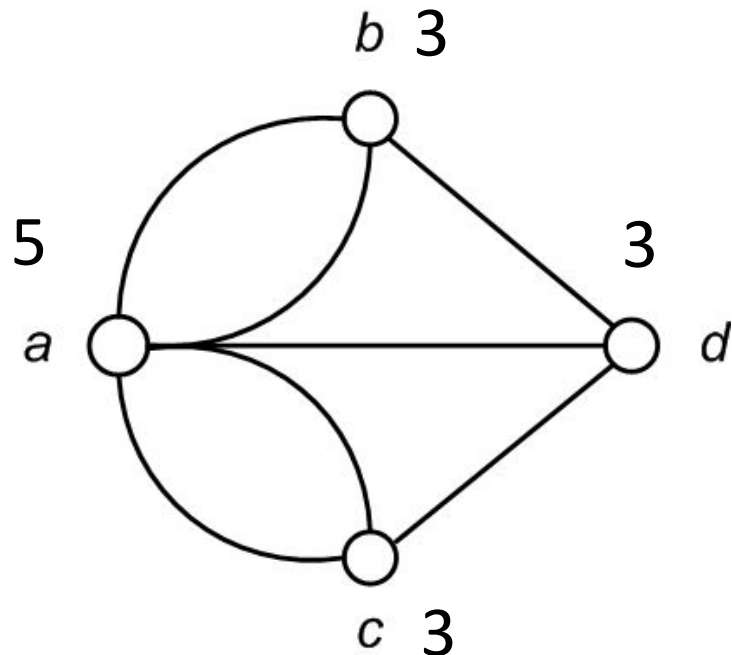
Часто требуется, чтобы длина этого маршрута была минимальной (для взвешенных графов), или ограничивается число проходов по одному и тому же элементу графа.

3.1. Эйлеровы цепи и циклы

Постановка задачи

Определение. *Эйлеровой цепью (циклом)* называется цепь (цикл), содержащая (содержащий) все ребра графа. Если в графе существует эйлерова цепь (цикл), то граф называется *эйлеровым*.

Эйлеров граф можно нарисовать одним росчерком пера.



Кенигсбергские мосты

Теорема Эйлера (1736 г.)

Теорема. Эйлерова цепь (цикл) существует тогда и только тогда, когда число вершин с нечетной валентностью равно 2 (0).

Доказательство.

1. Необходимость.

Пусть эйлерова цепь существует и проходит из вершины x в вершину y . Если эта цепь – цикл ($x = y$), то в каждую вершину цепь должна зайти столько же раз, сколько и выйти (т.е. валентности всех вершин – четные). Если $x \neq y$, то из x цепь должна выйти на один раз больше, чем зайти; в y цепь должна зайти на один раз больше, чем выйти (т.е. валентности вершин x и y должны быть нечетными).

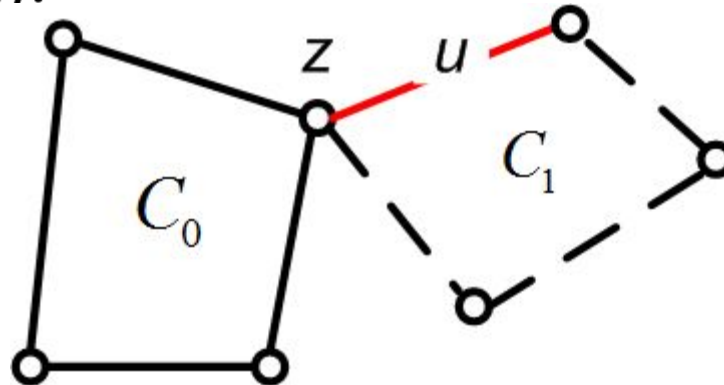
2. Достаточность (конструктивное доказательство).

Пусть валентности всех вершин, за исключением может быть вершин x и y , четные. Докажем существование эйлерова цикла, построив его.

Если все валентности четные: начнем цепь из некоторой произвольной вершины x и пойдём по некоторому еще не пройденному ребру к следующей вершине и так до тех пор, пока не вернемся в вершину x и не замкнем цикл. Если пройдены все ребра, то искомым эйлеров цикл C_0 построен.

Достаточность (продолжение).

Если в C_0 входят не все ребра графа, то цикл C_0 должен проходить через некоторую вершину z , инцидентную некоторому ребру u , не вошедшему в C_0 (т.к. граф связен).



Если удалить все ребра цикла C_0 , то в оставшемся суграфе вершины будут по-прежнему иметь четные валентности. Начиная теперь с вершины z построим цикл C_1 , начинающийся и кончающийся в z .

Достаточность (продолжение).

Если в C_1 пройдены все оставшиеся ребра, то процесс закончен. Нужный нам эйлеров цикл будет образован частью цикла C_0 до z , затем циклом C_1 и, наконец, частью цикла C_0 от z .

Если циклы C_1 и C_1 содержат не все ребра графа, то строится следующий цикл, и так далее, до тех пор, пока не будет найден эйлеров цикл.

Достаточность (окончание).

Если вершины x и y имеют нечетные валентности, то сначала находим цепь, соединяющую x и y ; удалив ребра этой цепи, построим эйлеров цикл, начиная с вершины x . Эйлерова цепь выходит теперь из вершины x , проходит эйлеров цикл, а затем снова из вершины x по найденной цепи в вершину y .



На этом доказательстве основан алгоритм нахождения эйлеровой цепи.

Алгоритм

Шаг 1. Находим простую цепь, соединяющую вершины с нечетной валентностью (если они есть) x и y .

Если $x \neq y$, то все ребра этой цепи пометим меткой $L = 0$.

Шаг 2. Разметка циклов. Пусть наибольшая из меток ребер равна L . Среди непомеченных ребер выбираем такое u , которое смежно хотя бы с одним помеченным. Из непомеченных ребер построим простой цикл, содержащий ребро u (например, с помощью алгоритма нахождения цепей) и все ребра этого цикла пометим меткой $L + 1$.

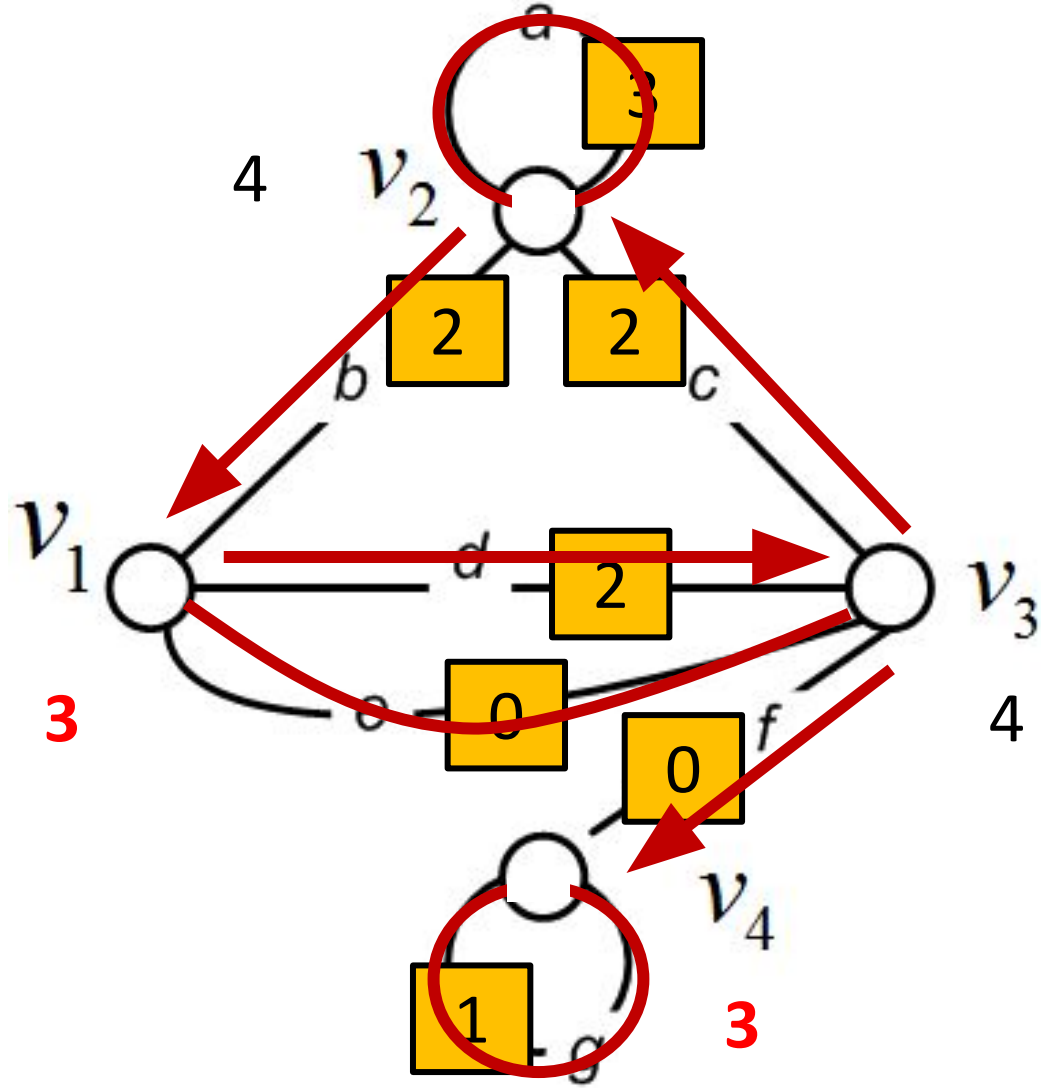
Разметка продолжается до тех пор, пока не будут помечены все ребра графа.

Шаг 3. Маршрут строится следующим образом: за начальную вершину выбираем $x_0 = x$. Если уже построен маршрут $x_0 u_1 x_1 \dots x_{k-1} u_k x_k$, ($k \geq 0$), котором все ребра различны, то

1) в случае $k = m$ (число ребер графа) процесс закончен (маршрут найден);

2) если $k < m$, то среди ребер, инцидентных x_k и отличных от $u_1 \dots u_k$ выбираем такое, которое **помечено наибольшей меткой** (или любое из них) и добавляем его к маршруту как ребро u_{k+1} , а за x_{k+1} берем ту вершину, с которой u_{k+1} соединяет вершину x_k (возможно $x_{k+1} = x$).

Пример



$v_1 d v_3 c v_2 a v_2 b v_1 e v_3 f v_4 g v_4$

Задача китайского почтальона

В графе с неотрицательными весами ребер найти циклический маршрут наименьшей длины, проходящий через каждое ребро графа по крайней мере один раз.

Почтальон называется китайским потому, что первоначально эту задачу изучал китайский математик Мэй-Ку Куан в 1962 году.

Очевидно, если граф эйлеров, то эйлеров цикл и будет оптимальным.

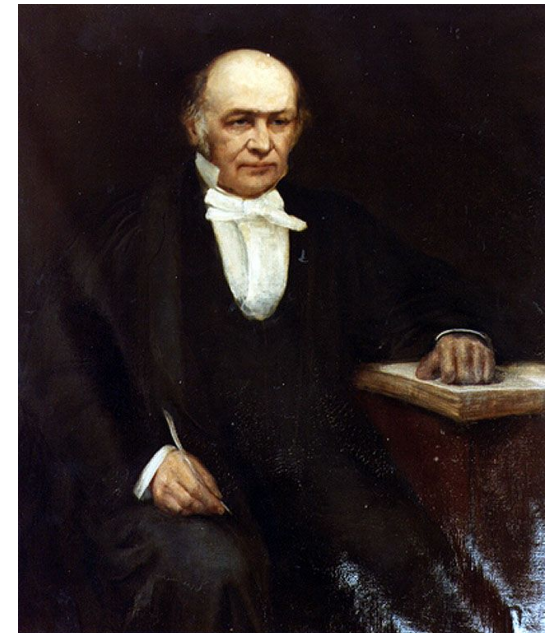
Если граф не эйлеров, то задача становится весьма трудоемкой. Для ее решения имеются специальные алгоритмы, сокращающие число перебираемых вариантов.

3.1. Гамильтоновы цепи и циклы

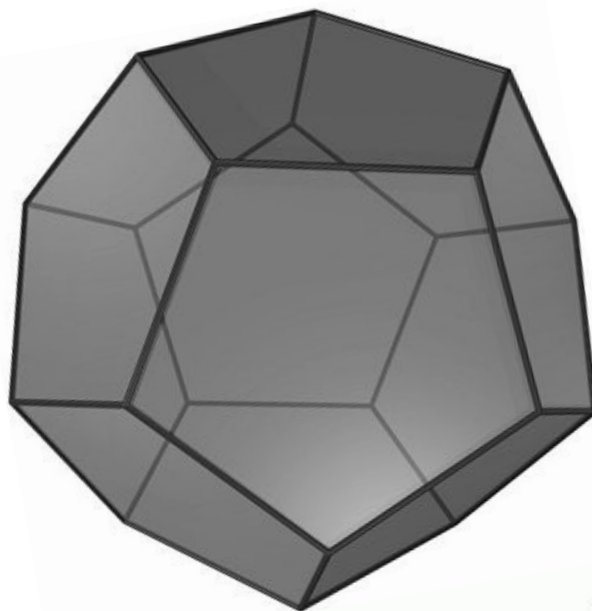
Постановка задачи

Определение. *Гамильтоновой цепью (циклом)* графа называется простая цепь (простой цикл), содержащая (содержащий) все вершины графа.

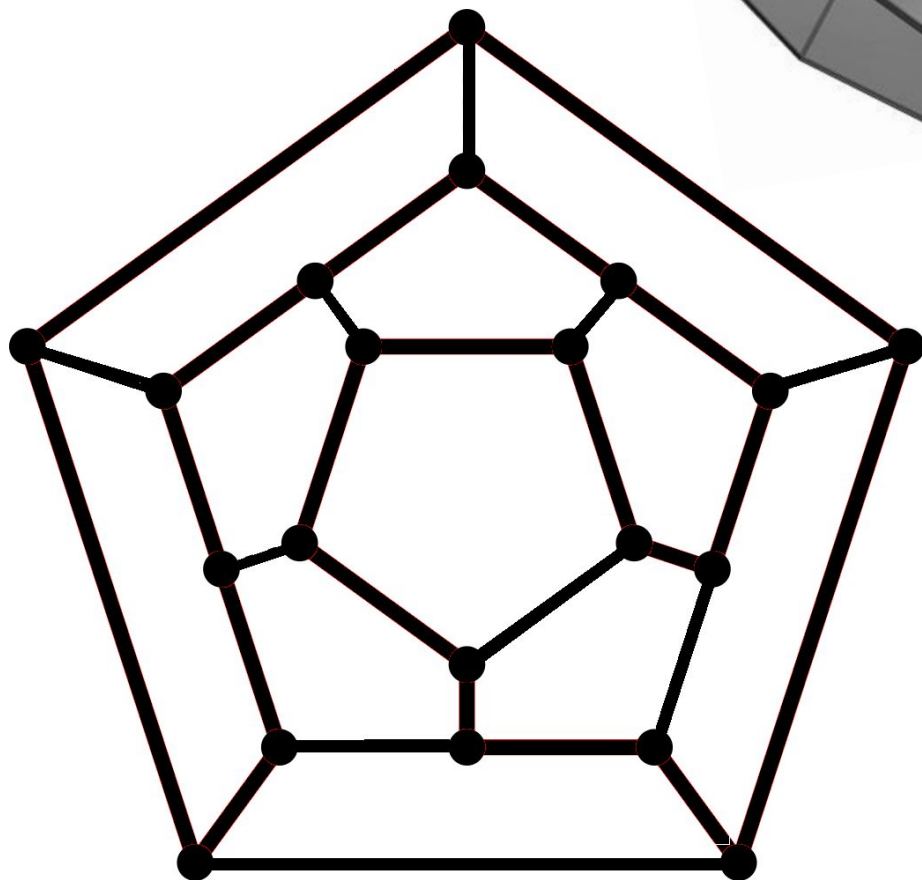
Сэр Уильям Роуэн Гамильтон (1805-1865) – самый известный ирландский математик, один из лучших математиков 19 века. Известен фундаментальными открытиями в математике, механике, физике (векторный анализ, вариационное исчисление, общий принцип наименьшего действия в механике, теория распространения света и др.)



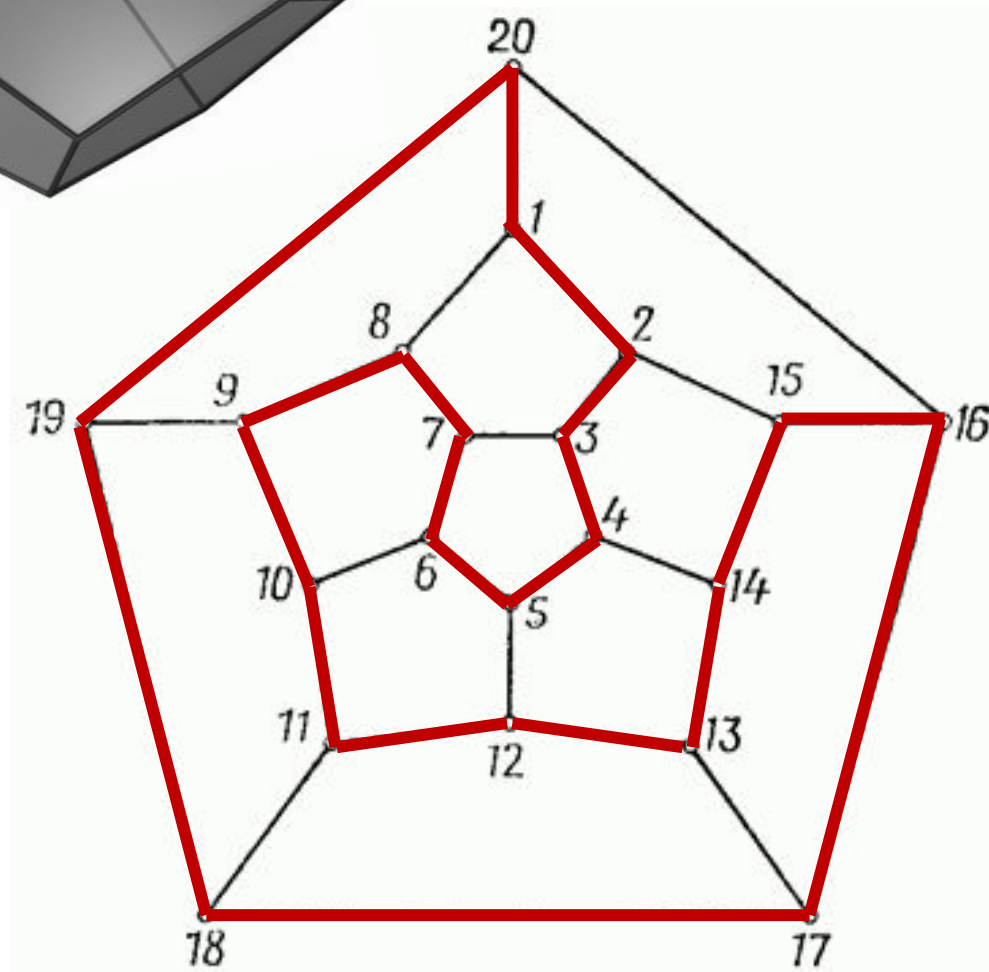
В 1859 г. Гамильтон изобрел головоломку обхода вершин додекаэдра



Додекаэдр:
12 граней,
20 вершин,
30 ребер



Граф Гамильтона



Гамильтонов цикл

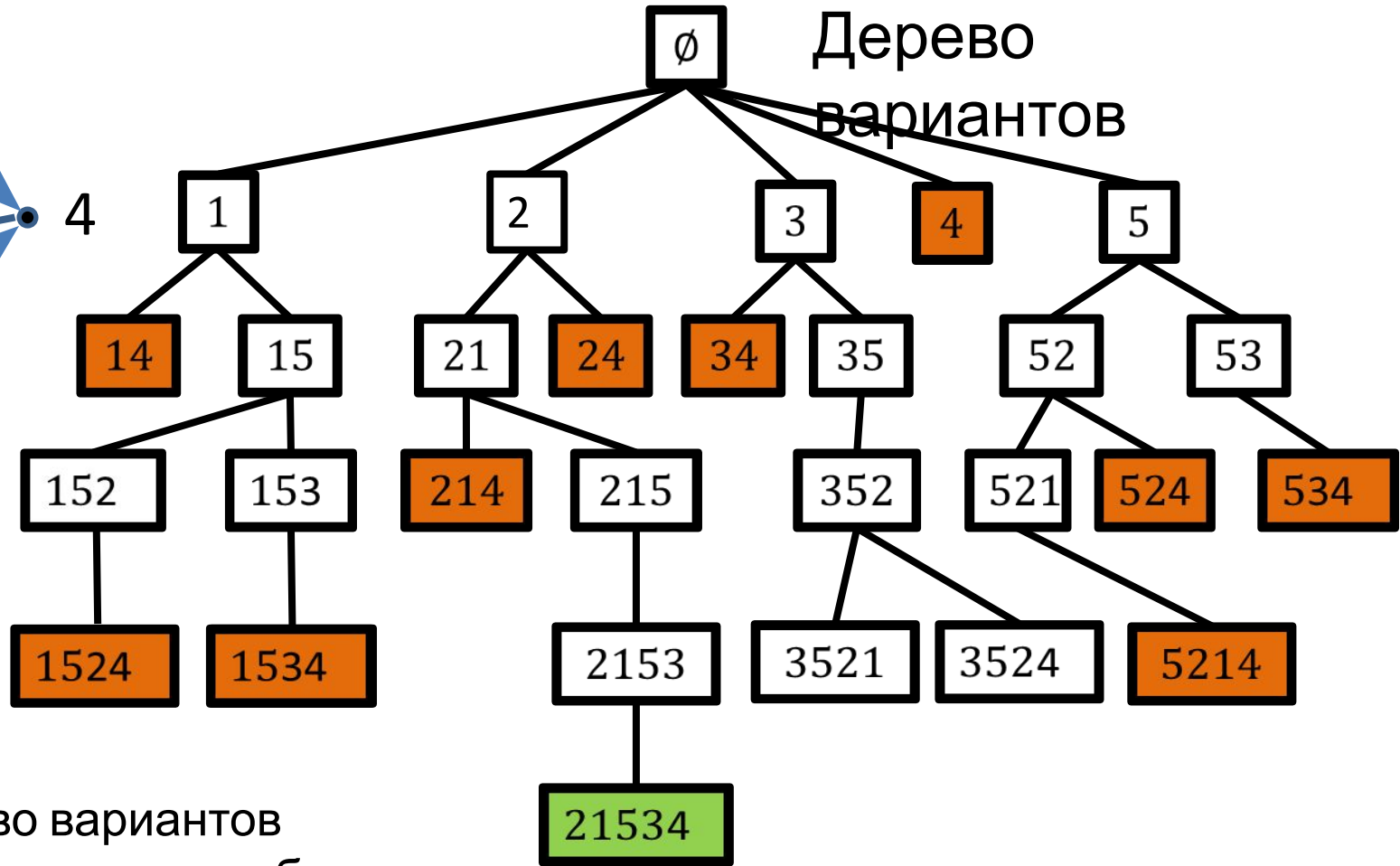
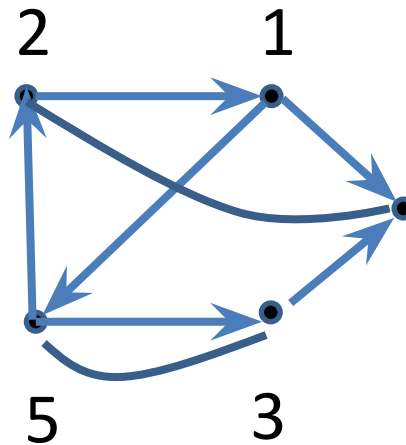
Понятия гамильтоновой цепи и цикла естественным образом обобщаются на случай ориентированных графов: двигаться в очередную вершину можно только по направлениям, указанным стрелками.

Пока неизвестен какой-либо достаточно простой критерий необходимости и достаточности существования гамильтоновой цепи (цикла) для произвольного графа G .

Неизвестен также алгоритм нахождения гамильтоновой цепи (цикла) полиномиальной сложности.

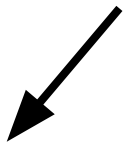
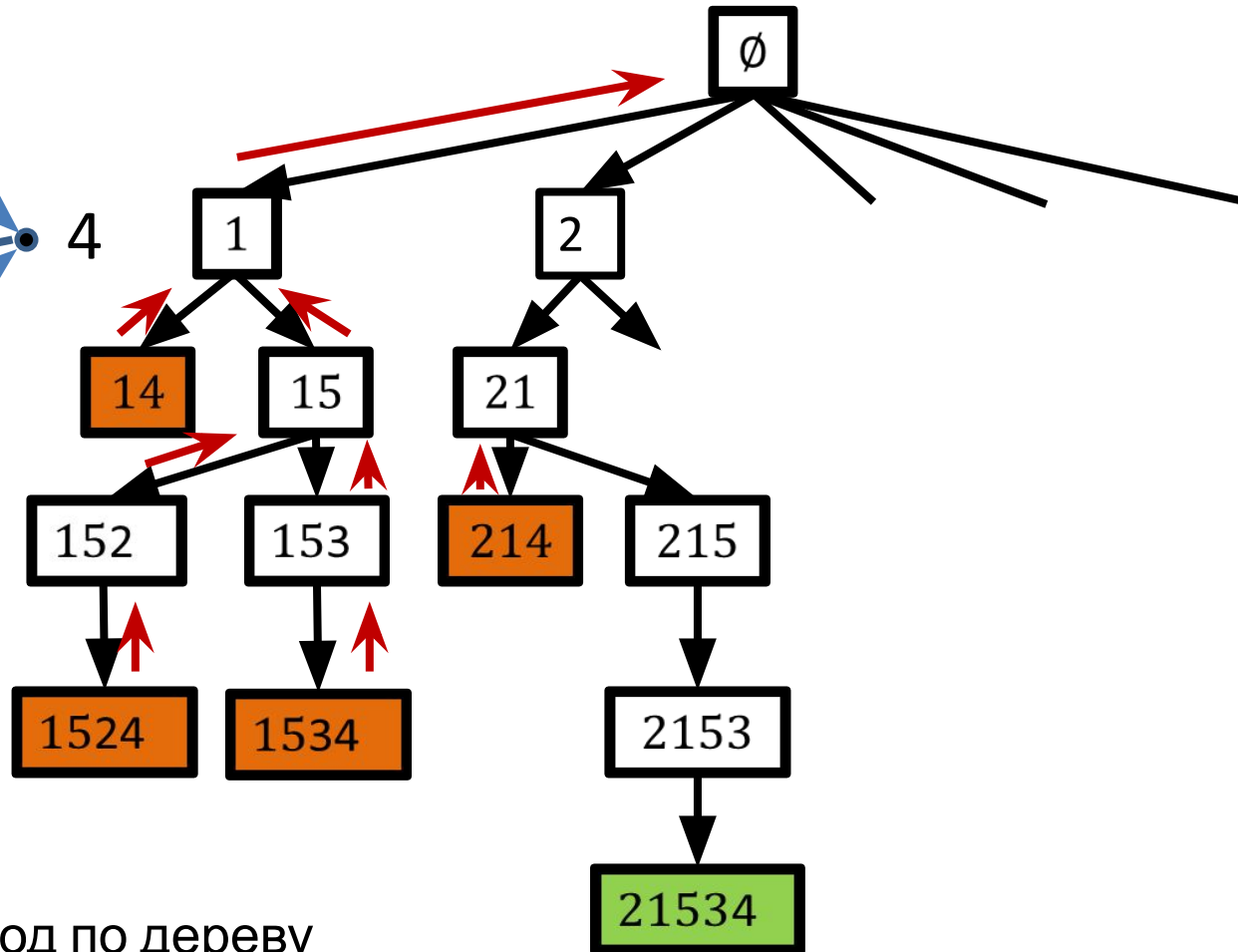
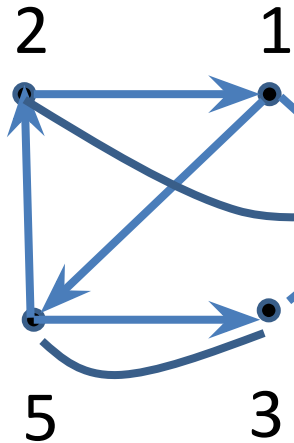
Мы рассмотрим два простейших переборных алгоритма поиска гамильтоновой цепи (цикла), пригодных для графов малой размерности: поиск в ширину и поиск в глубину.

Поиск в ширину



- множество вариантов
стоит исследовать глубже
- бесперспективное (тупиковое)
множество вариантов

Поиск в глубину



- прямой ход по дереву
поиска



- обратный ход (**back tracking**)

Задача коммивояжера

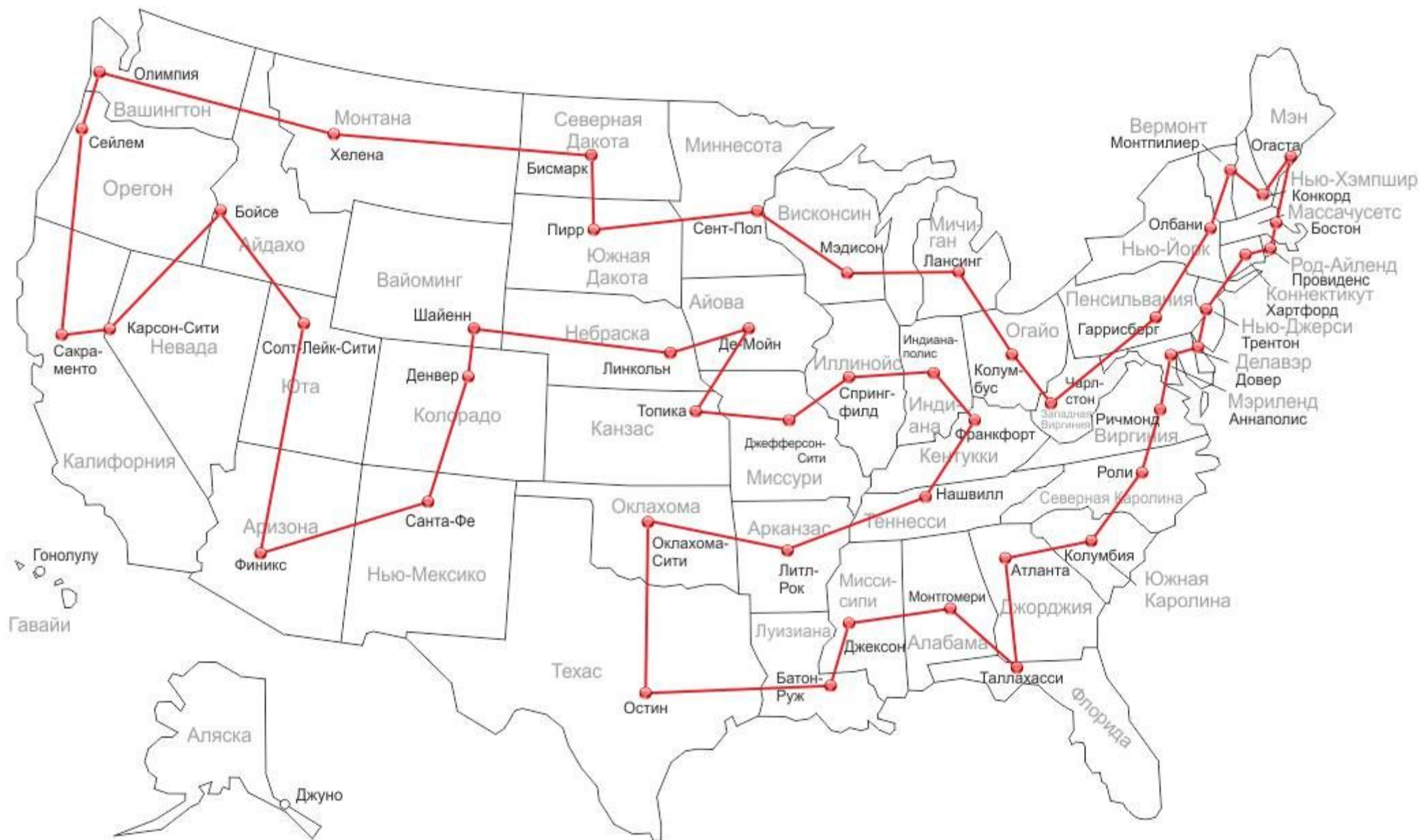
Задача коммивояжера (*travelling seller problem, TSP*) формулируется следующим образом: во взвешенном графе (обычно – полном) найти гамильтонову цепь (цикл) наименьшей длины (с наименьшим суммарным весом ребер).

Название задачи происходит от следующей формулировки: имеется n городов и известны расстояния между каждой парой городов; коммивояжер (бродячий торговец), выходящий из какого-либо города, должен посетить $n - 1$ других городов и вернуться в исходный. В каком порядке ему нужно посещать города, чтобы общее пройденное расстояние было минимальным?

Самый простой способ – перебрать все $(n - 1)!$ гамильтоновых циклов (если граф полный), однако трудоемкость этого перебора имеет порядок n^n . Уже при $n > 60$ перебор невозможен никакими теоретически мыслимыми компьютерами за время, меньшее нескольких миллиардов лет. Но это – единственный способ найти **точное** решение.

Существует множество алгоритмов нахождения **приближенных** решений. Задача коммивояжера из-за простоты постановки и наглядности является прекрасной моделью для разработки новых алгоритмов дискретной оптимизации. К ним относятся алгоритмы, реализующие **метод ветвей и границ, генетические алгоритмы** и др.

Пример. Столицы 48 штатов



Длина пути 16675 миль. Алгоритм LMatrix