Алгоритмы для задачи коммивояжёра

Александр Куликов

Петербургское отделение Математического института им. В. А. Стеклова
Российская академия наук

Computer Science клуб
24 февраля 2012

1. Введение

2. Эвристики
   - Метод ветвей и границ
   - Метод локального поиска

3. Приближённые алгоритмы
   - 1.5-приближение для Metric-TSP
   - Неприближаемость общего случая

4. Точные алгоритмы
   - Динамическое программирование
   - Формула включений-исключений
   - Матрица Татта и перманент

Содержание

Формулировка задачи

- Задача о гамильтоновом цикле: проверить, есть ли в графе цикл, проходящий по каждой вершине ровно один раз.

Формулировка задачи

- Задача о гамильтоновом цикле: проверить, есть ли в графе цикл, проходящий по каждой вершине ровно один раз.
- Задача коммивояжёра: найти в данном *полном* взвешенном графе гамильтонов цикл минимального веса.

Формулировка задачи

- Задача о гамильтоновом цикле: проверить, есть ли в графе цикл, проходящий по каждой вершине ровно один раз.
- Задача коммивояжёра: найти в данном *полном* взвешенном графе гамильтонов цикл минимального веса.
- Периодически мы будем искать не цикл, а путь.

Формулировка задачи

- Задача о гамильтоновом цикле: проверить, есть ли в графе цикл, проходящий по каждой вершине ровно один раз.
- Задача коммивояжёра: найти в данном *полном* взвешенном графе гамильтонов цикл минимального веса.
- Периодически мы будем искать не цикл, а путь.
- Применения: проектирование схем, планирование, сборка генома.

Формулировка задачи

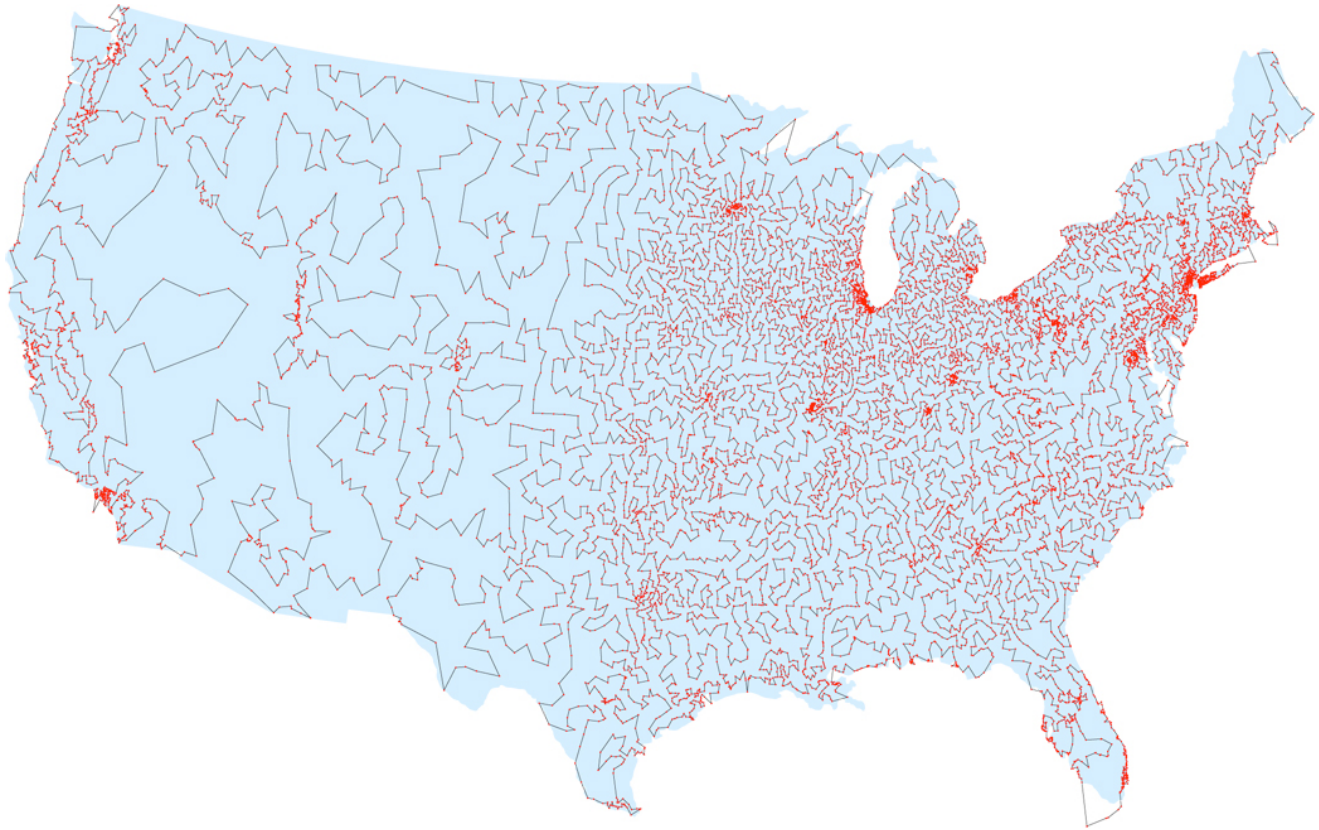
- Задача о гамильтоновом цикле: проверить, есть ли в графе цикл, проходящий по каждой вершине ровно один раз.
- Задача коммивояжёра: найти в данном *полном* взвешенном графе гамильтонов цикл минимального веса.
- Периодически мы будем искать не цикл, а путь.
- Применения: проектирование схем, планирование, сборка генома.
- Сложность полного перебора: $O(n!)$.

Цикл по 15 городам Германии

Оптимальный маршрут коммивояжёра через 15 крупнейших городов Германии. Указанный маршрут является самым коротким из всех возможных 43 589 145 600.

http://en.wikipedia.org/wiki/Travelling_salesman_problem

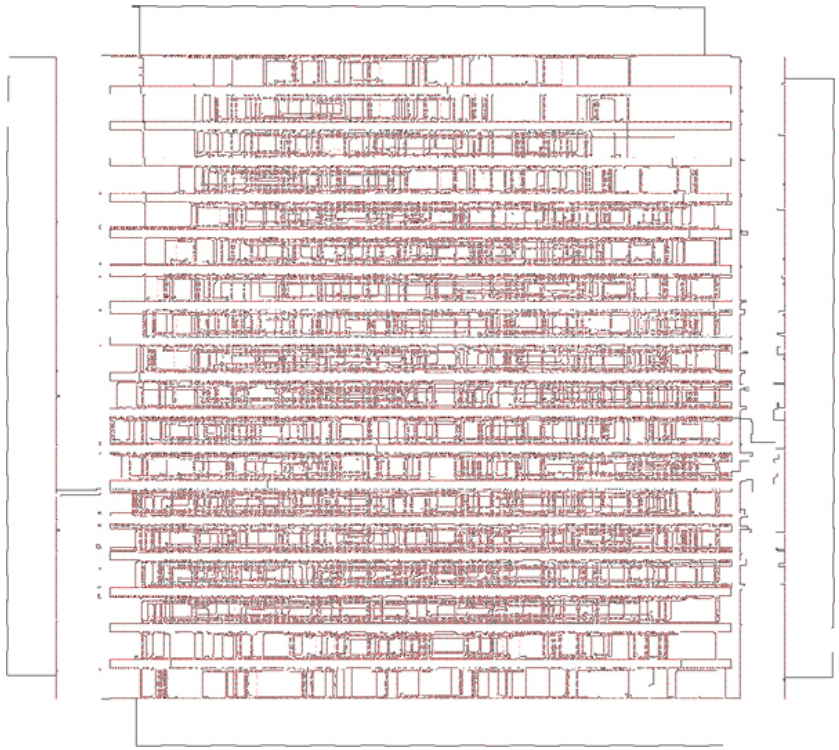
Цикл по 13 509 городам США



David Applegate, Robert Bixby, Vasek Chvatal and William Cook.
The Traveling Salesman Problem: A Computational Study.

Оптимальный путь лазера

85 900 «городов»



David Applegate, Robert Bixby, Vasek Chvatal and William Cook.
The Traveling Salesman Problem: A Computational Study.

Ещё интересное

`http://www.tsp.gatech.edu/`

- две книги
- мировые рекорды
- датасеты
- программы
- игры
- триллер

Содержание

Содержание

Метод ветвей и границ

1 Начать с некоторой задачи P_0
2 $S = \{P_0\} \leftarrow$ множество активных подзадач
3 лучшийрезультат $= \infty$
4 **while** S не пусто
5 **do** выбрать подзадачу (частичное решение) $P \in S$
и удалить её из S
6 разбить P на меньшие подзадачи $P_1, P_2, \cdots, P_k$
7 **for** каждой P_i
8 **do if** P_i является полным решением
9 **then** обновить лучшийрезультат
10 **elseif** нижняяграница$(P_i) <$ лучшийрезультат
11 **then** добавить P_i в S
12 **return** лучшийрезультат

Подзадачи и нижняя граница

- подзадача: $[a, S, b]$ — достроение простого пути из a в b, проходящего по всем вершинам из $S \ni a, b$ (то есть кратчайший путь из b в a, проходящий по $V \setminus S$)

Подзадачи и нижняя граница

- подзадача: $[a, S, b]$ — достроение простого пути из a в b, проходящего по всем вершинам из $S \ni a, b$ (то есть кратчайший путь из b в a, проходящий по $V \setminus S$)
- начальная задача: $[a, \{a\}, a]$

Подзадачи и нижняя граница

- подзадача: $[a, S, b]$ — достроение простого пути из a в b, проходящего по всем вершинам из $S \ni a, b$ (то есть кратчайший путь из b в a, проходящий по $V \setminus S$)
- начальная задача: $[a, \{a\}, a]$
- нижняя граница — сумма из

Подзадачи и нижняя граница

- подзадача: $[a, S, b]$ — достроение простого пути из a в b, проходящего по всем вершинам из $S \ni a, b$ (то есть кратчайший путь из b в a, проходящий по $V \setminus S$)
- начальная задача: $[a, \{a\}, a]$
- нижняя граница — сумма из
  - самого лёгкого ребра из a в $V \setminus S$,

Подзадачи и нижняя граница

- подзадача: $[a, S, b]$ — достроение простого пути из a в b, проходящего по всем вершинам из $S \ni a, b$ (то есть кратчайший путь из b в a, проходящий по $V \setminus S$)
- начальная задача: $[a, \{a\}, a]$
- нижняя граница — сумма из
  - самого лёгкого ребра из a в $V \setminus S$,
  - самого лёгкого ребра из b в $V \setminus S$ и

Подзадачи и нижняя граница

- подзадача: $[a, S, b]$ — достроение простого пути из a в b, проходящего по всем вершинам из $S \ni a, b$ (то есть кратчайший путь из b в a, проходящий по $V \setminus S$)
- начальная задача: $[a, \{a\}, a]$
- нижняя граница — сумма из
  - самого лёгкого ребра из a в $V \setminus S$,
  - самого лёгкого ребра из b в $V \setminus S$ и
  - минимального покрывающего дерева графа на вершинах $V \setminus S$.

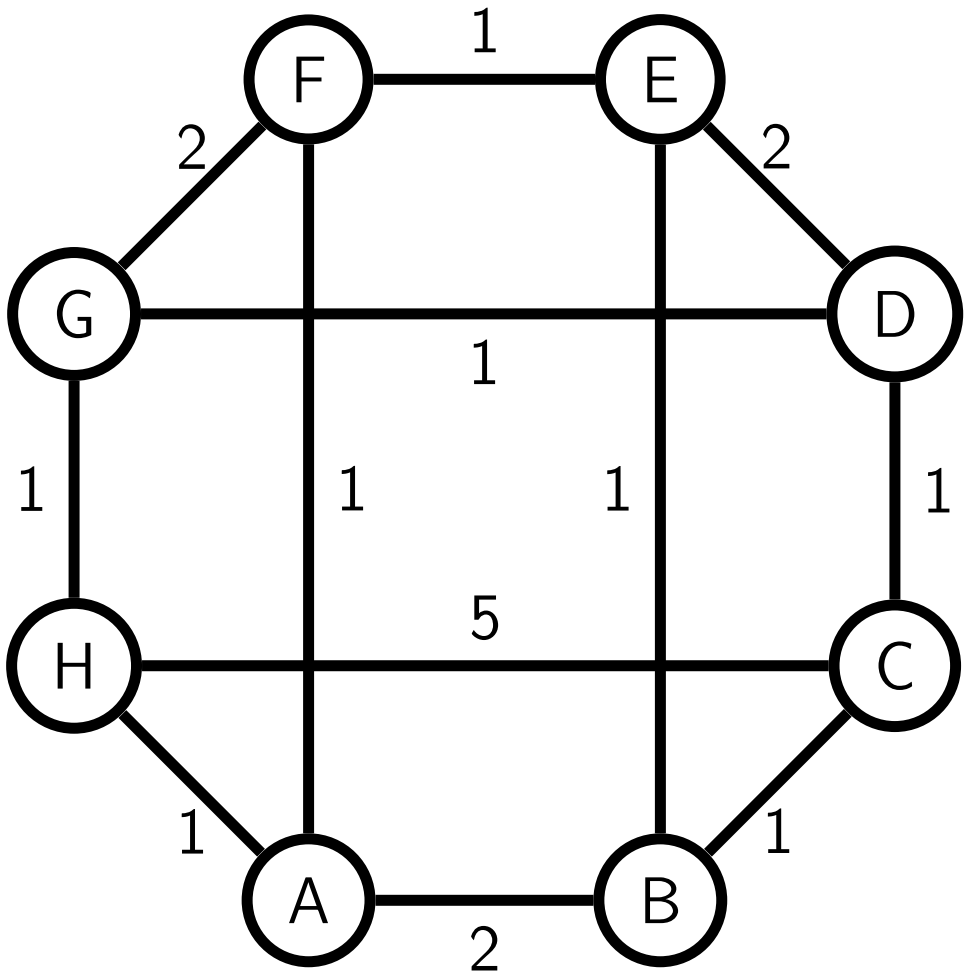
Кстати, о минимальных покрывающих деревьях

- Задача о минимальном покрывающем дереве — оставить в графе $(n-1)$ ребро, так чтобы граф остался связным и чтобы суммарный вес был минимальным. Решается почти за линейное время.

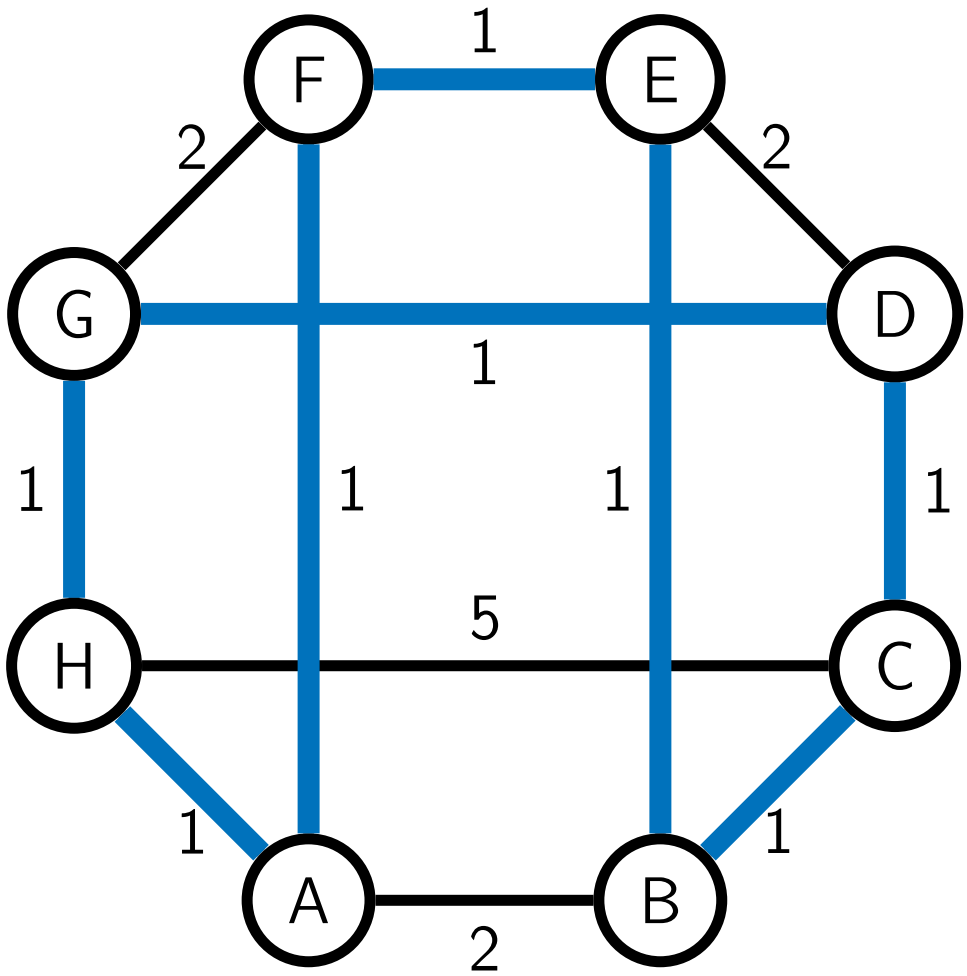
Кстати, о минимальных покрывающих деревьях

- Задача о минимальном покрывающем дереве — оставить в графе $(n-1)$ ребро, так чтобы граф остался связным и чтобы суммарный вес был минимальным. Решается почти за линейное время.
- Задача о минимальном пути коммивояжёра — то же самое, но запрещаем вершины степени больше двух. До сих пор не умеем решать быстрее 2^n.

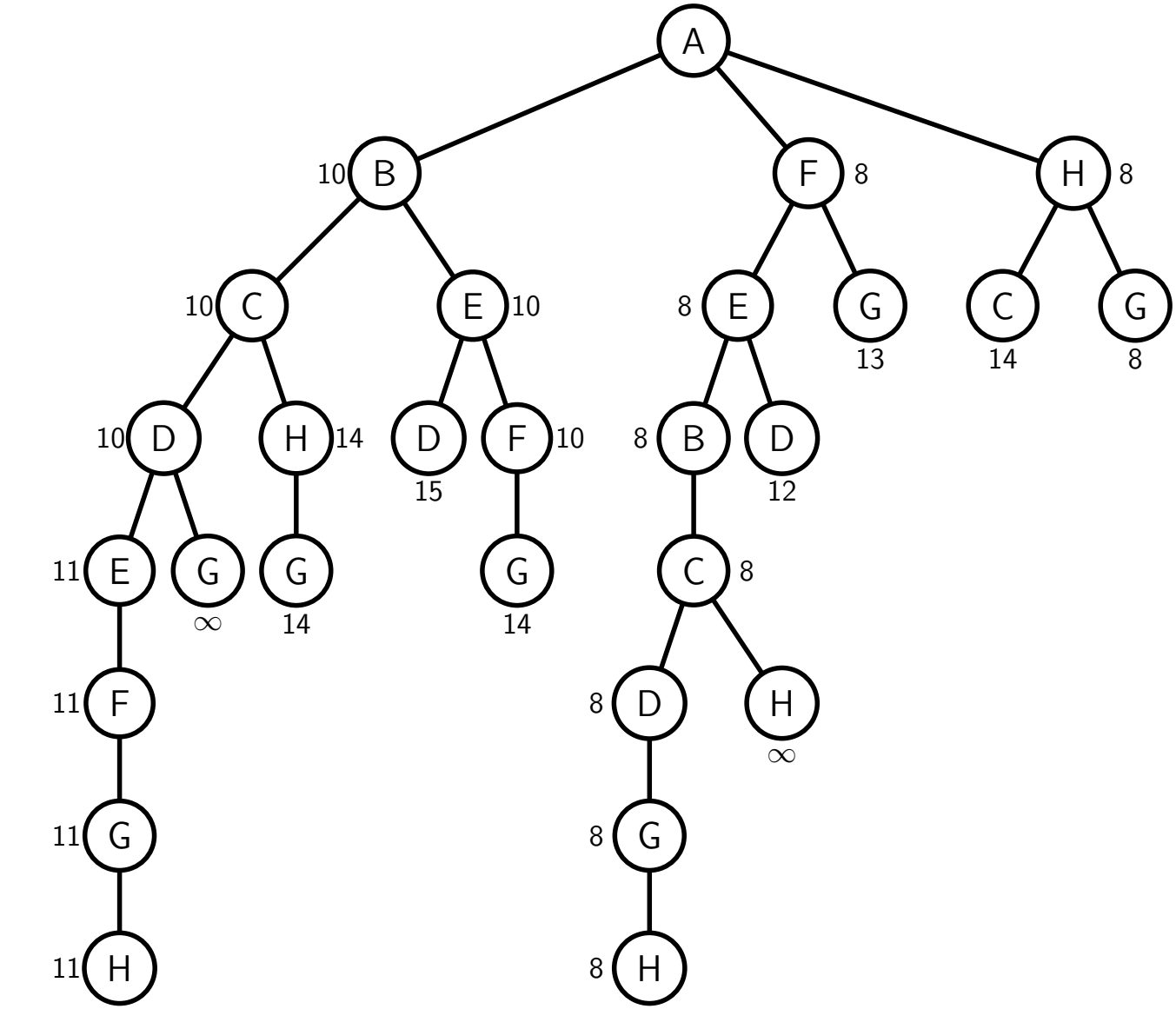
Пример графа



Пример графа



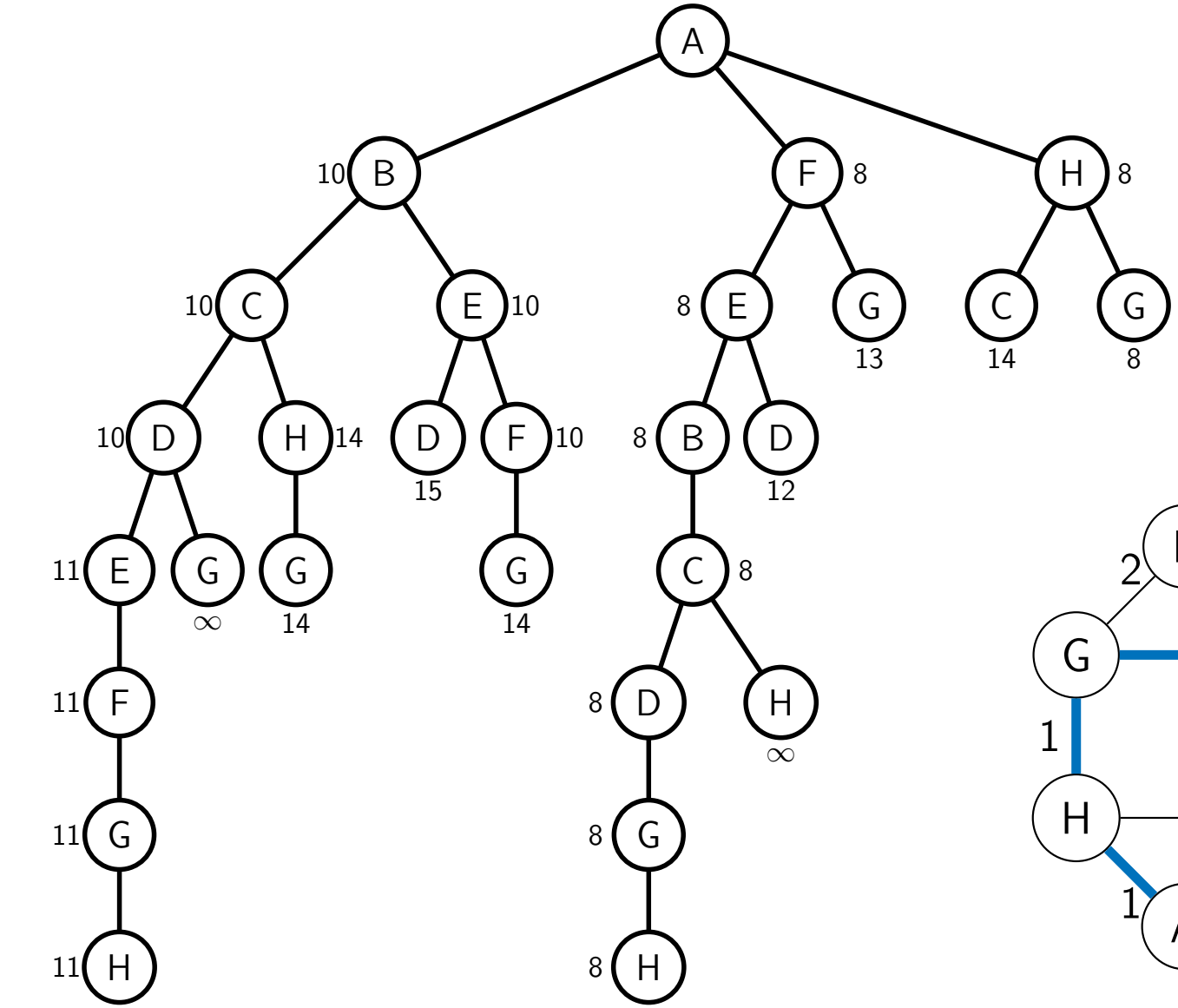
Дерево поиска



Стоимость: 11

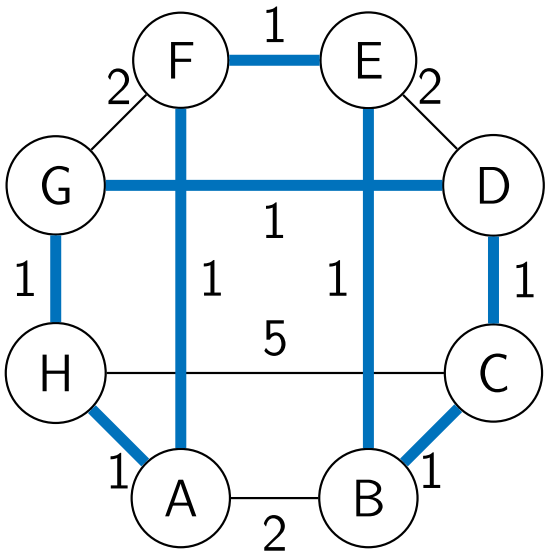
Стоимость: 8

Дерево поиска



Стоимость: 11

Стоимость: 8

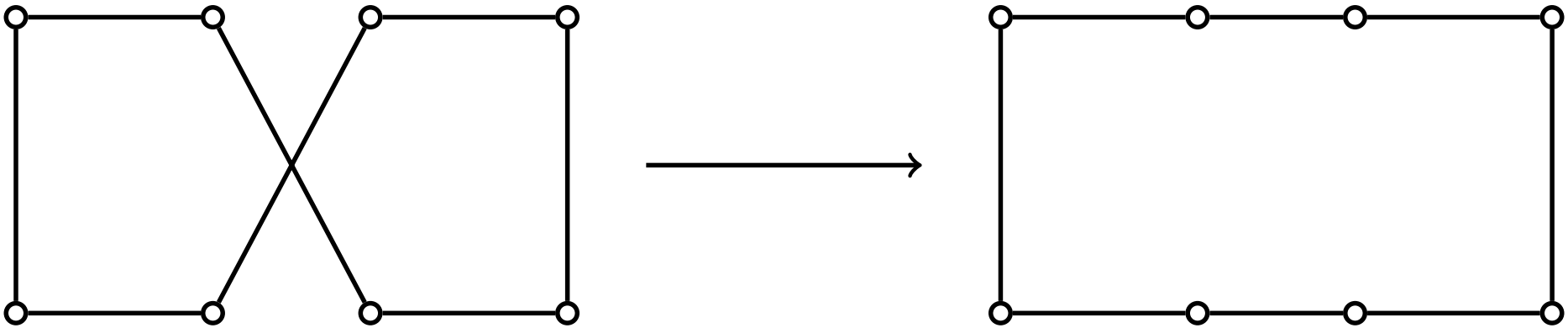


Содержание

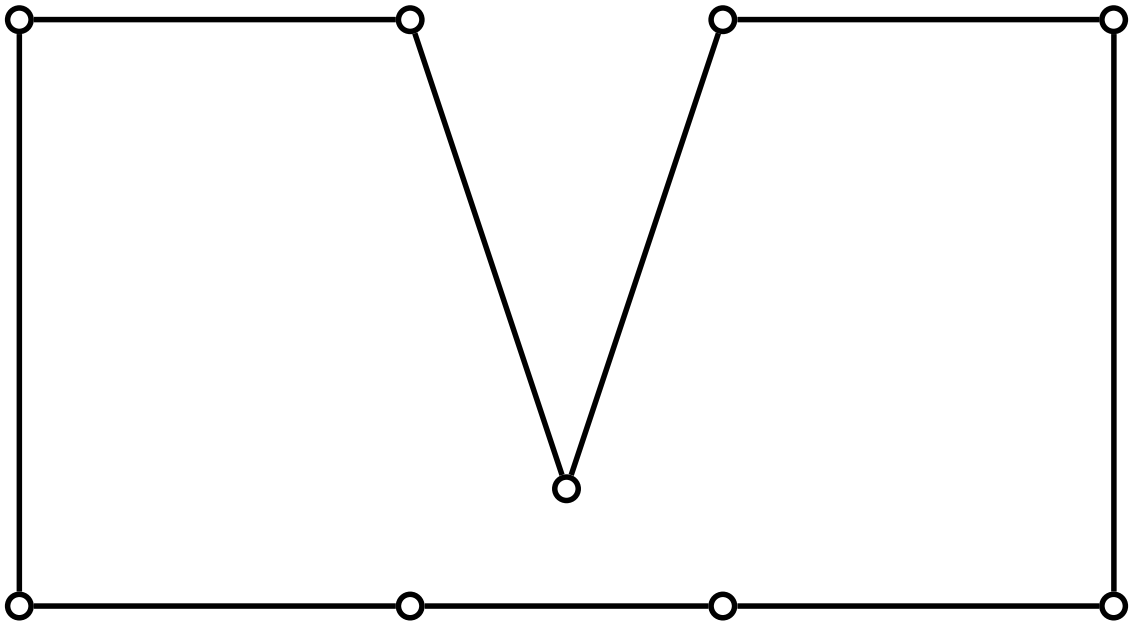
Локальный поиск

1 $s \leftarrow$ какое-нибудь начальное решение
2 **while** в окрестности s есть решение s' большей стоимости
3 **do** заменить s на s'
4 **return** s

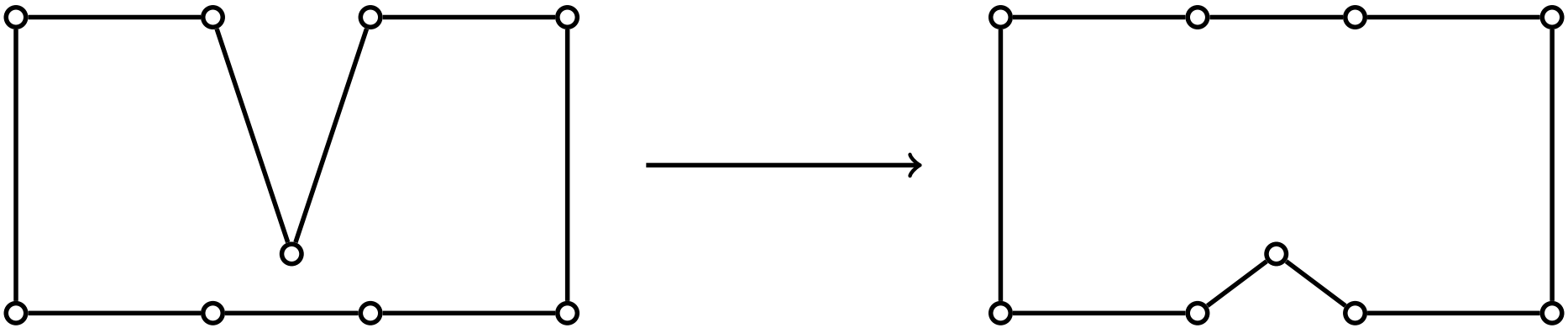
2-окружение



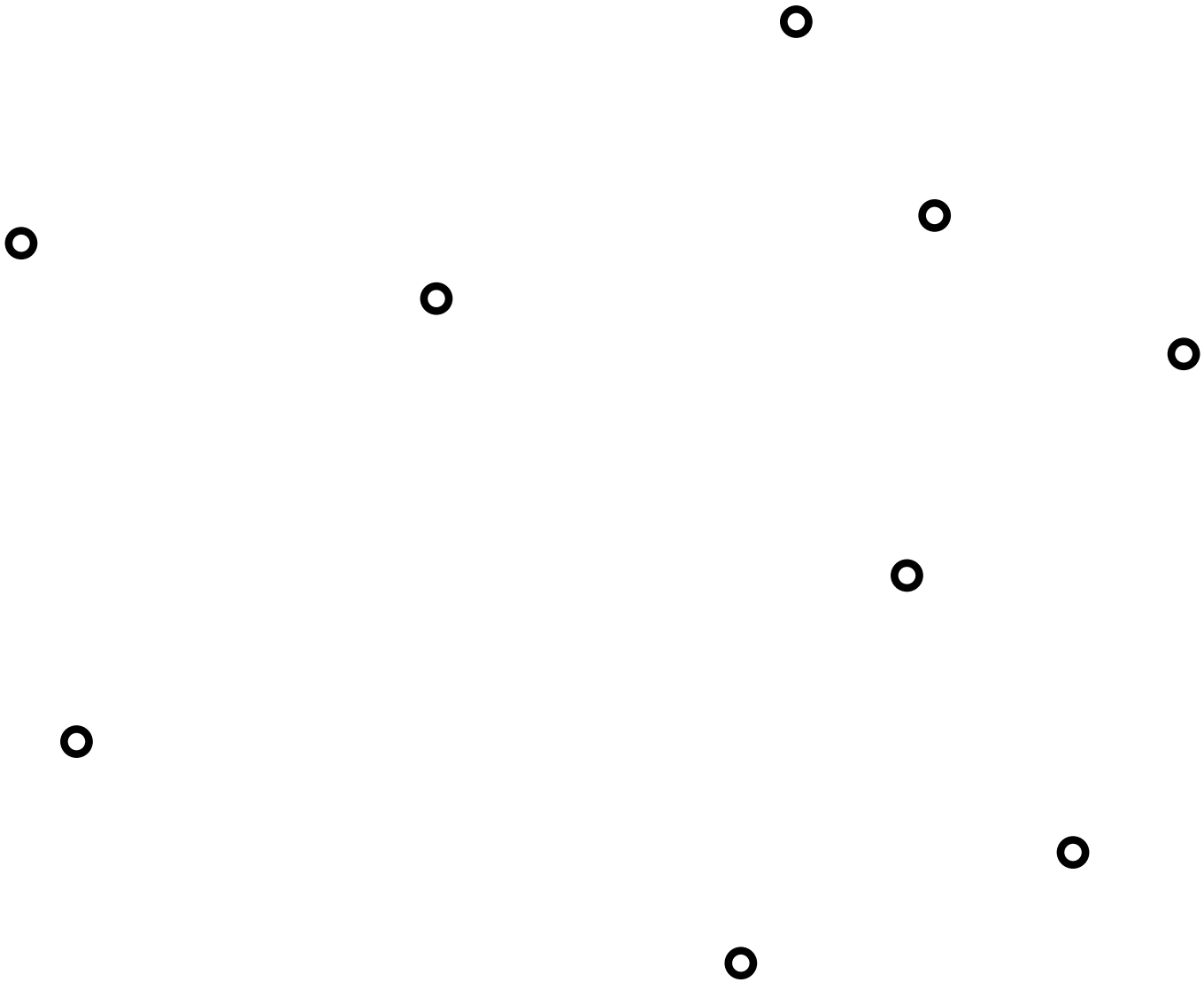
Узкое место



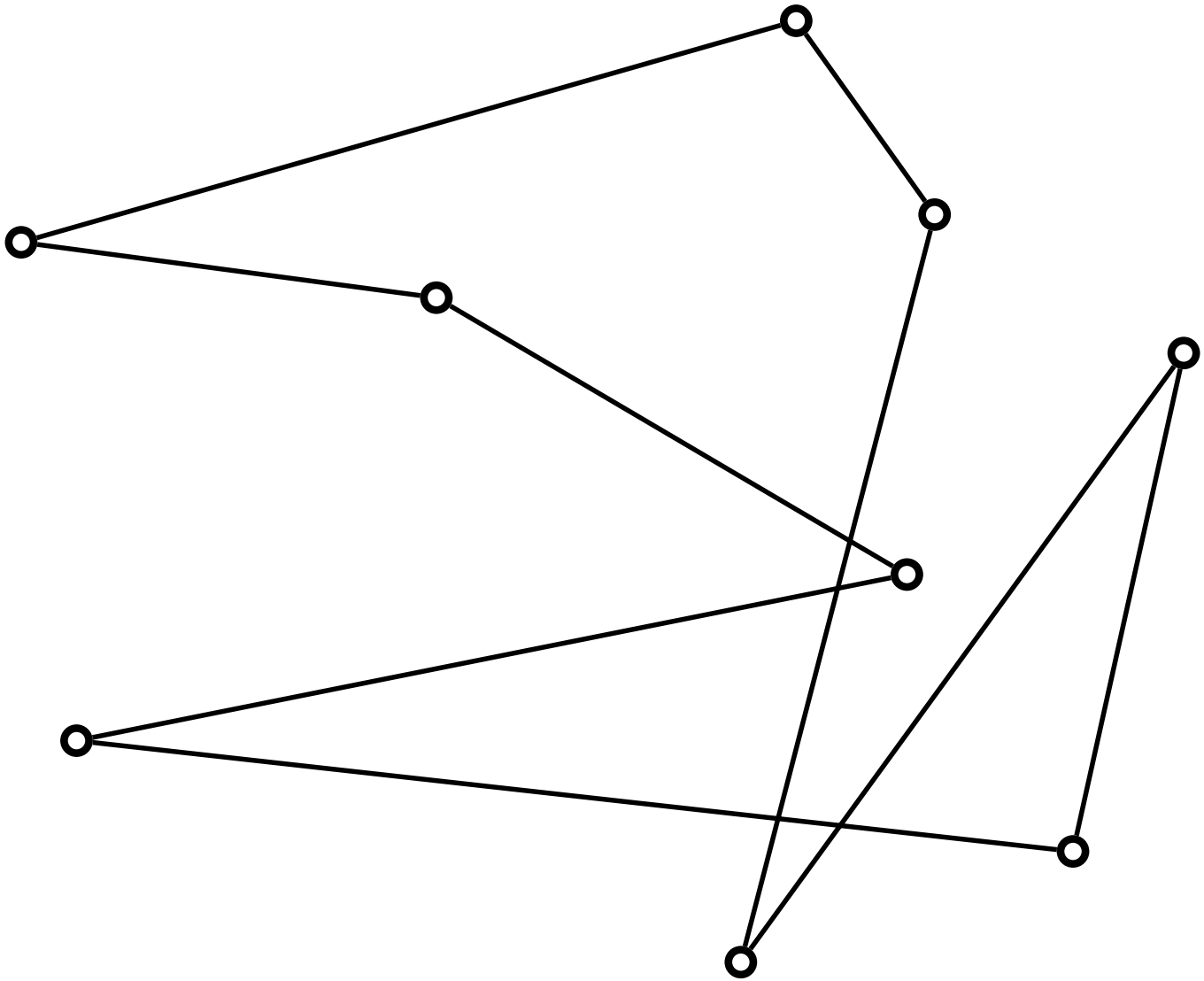
3-окружение



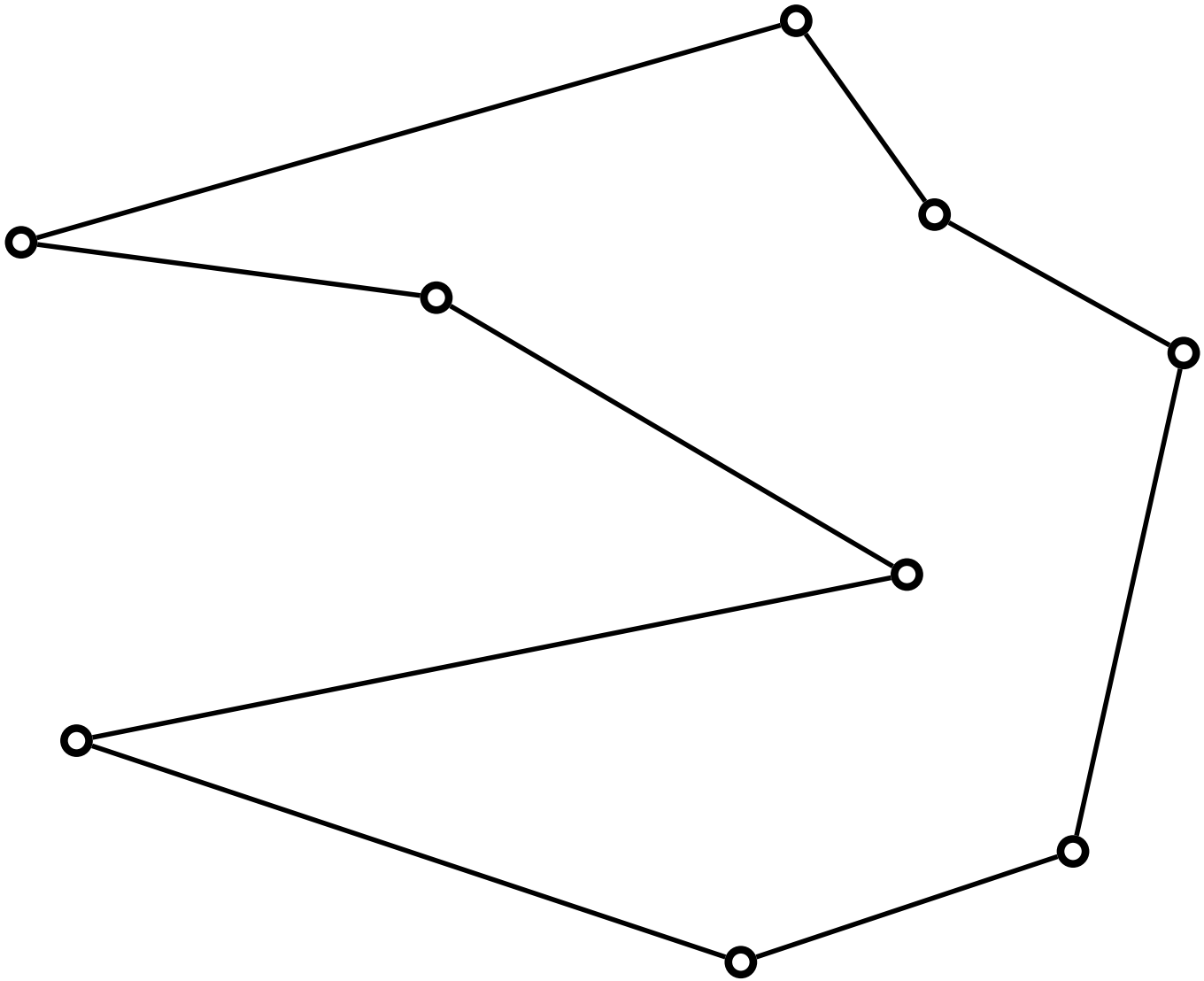
Пример локального поиска (с 3-окружением)



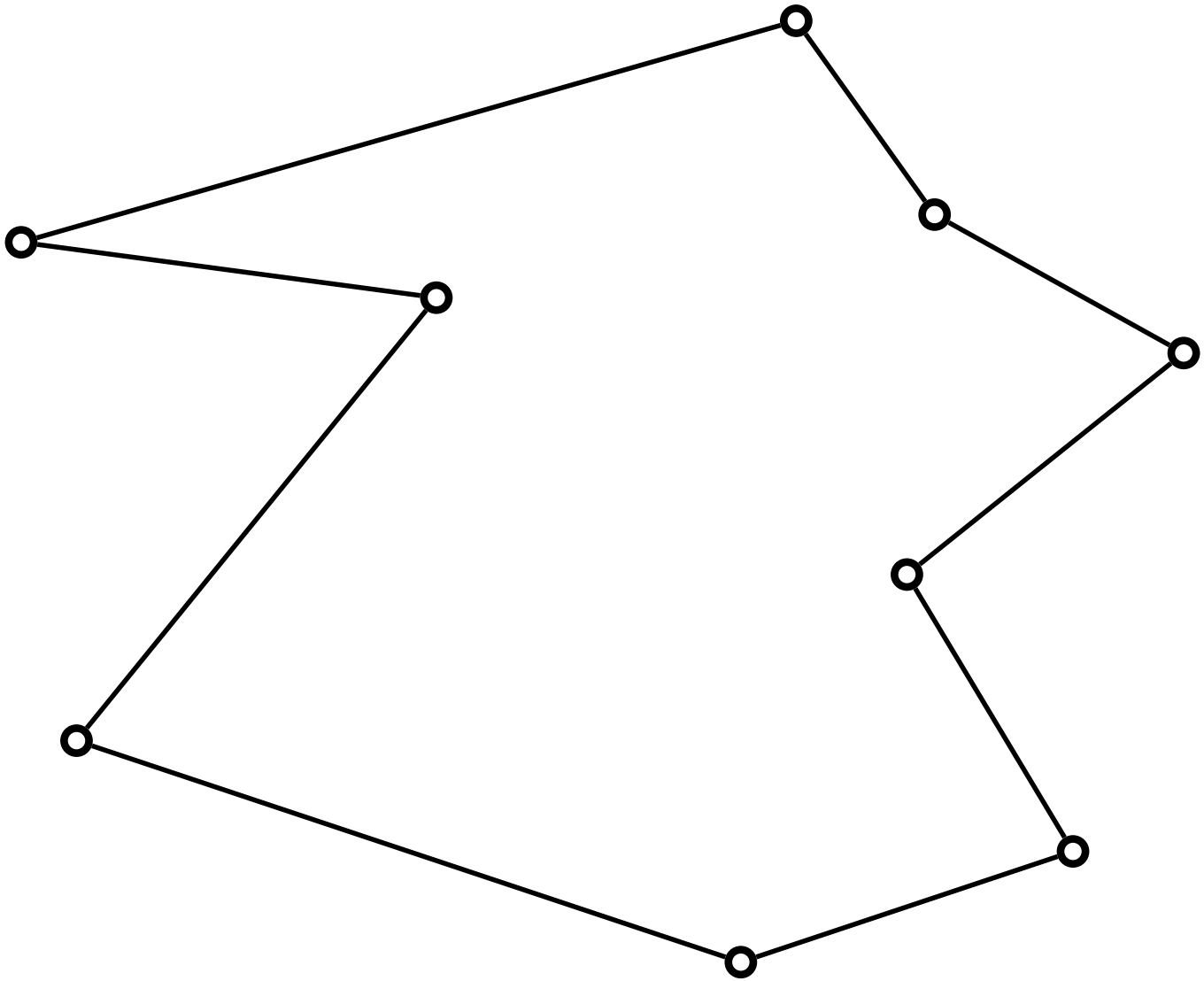
Пример локального поиска (с 3-окружением)



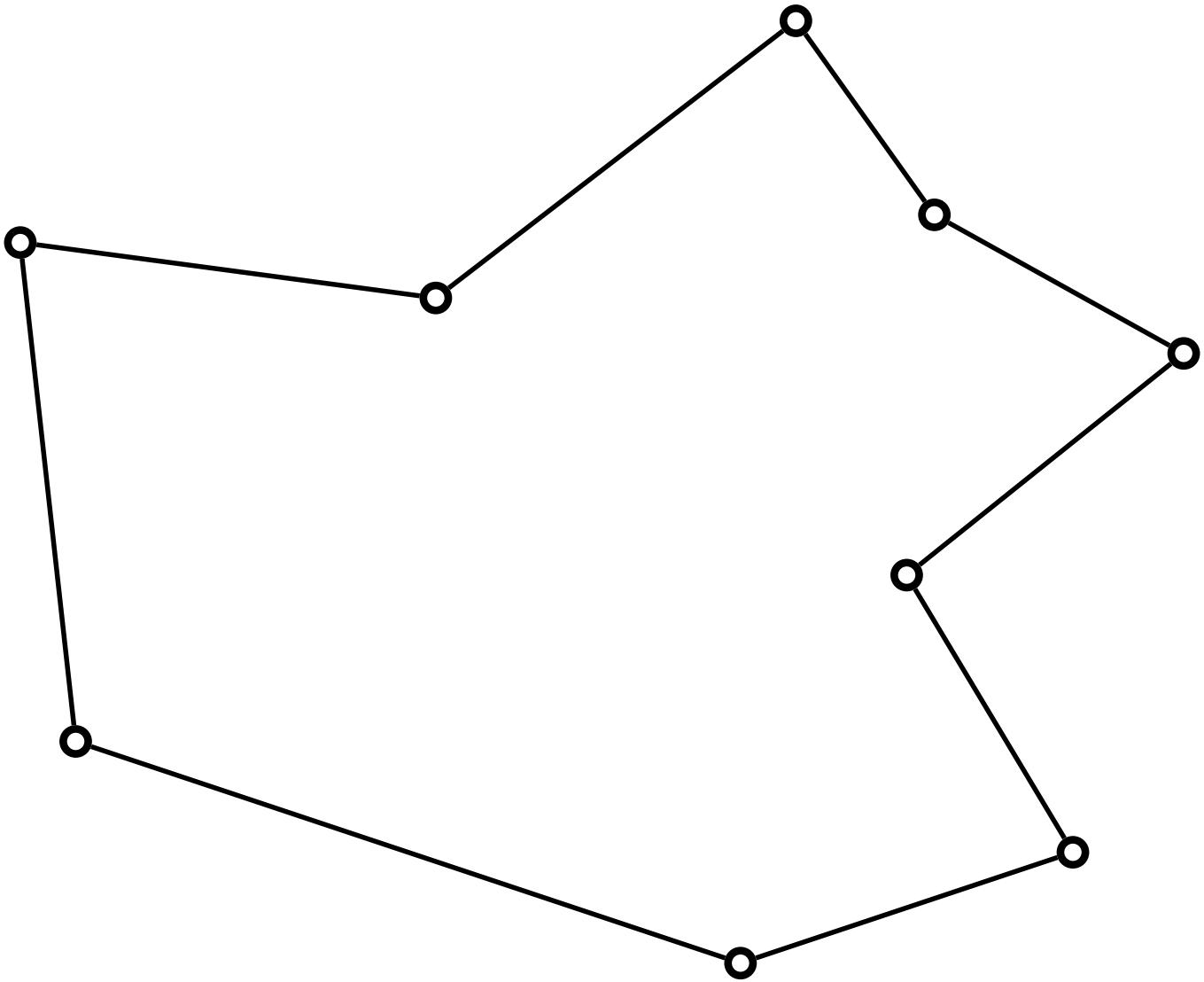
Пример локального поиска (с 3-окружением)



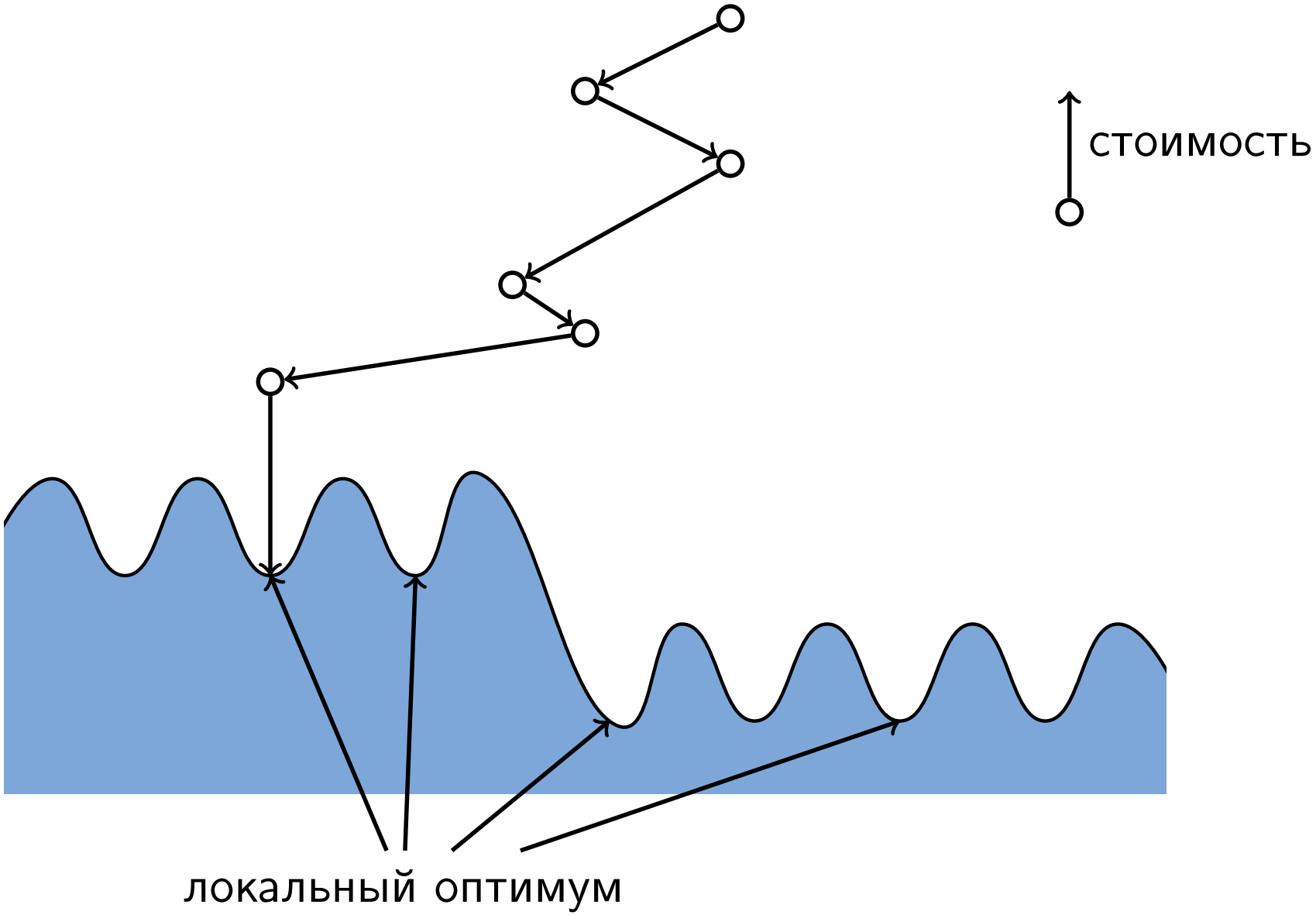
Пример локального поиска (с 3-окружением)



Пример локального поиска (с 3-окружением)



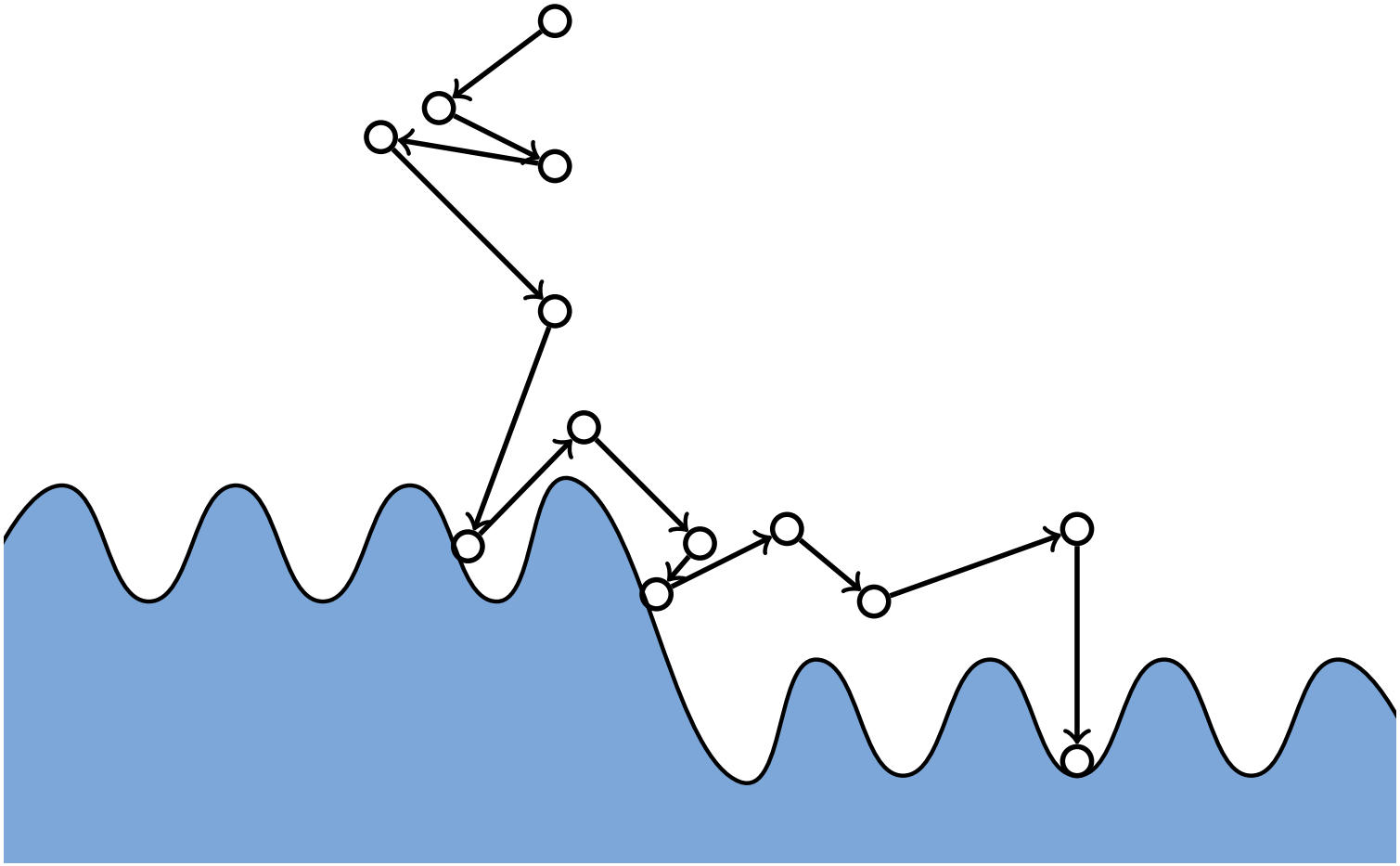
Локальный поиск абстрактно



Метод имитации отжига

1 $s \leftarrow$ какое-нибудь начальное решение
2 **repeat**
3 выбрать случайное решение s' из окружения s
4 $\Delta \leftarrow \text{cost}(s') - cost(s)$
5 **if** $\Delta < 0$
6 **then** заменить s на s'
7 **else** заменить s на s' с вероятностью $e^{-\Delta/T}$

Метод имитации отжига абстрактно



Содержание

Содержание

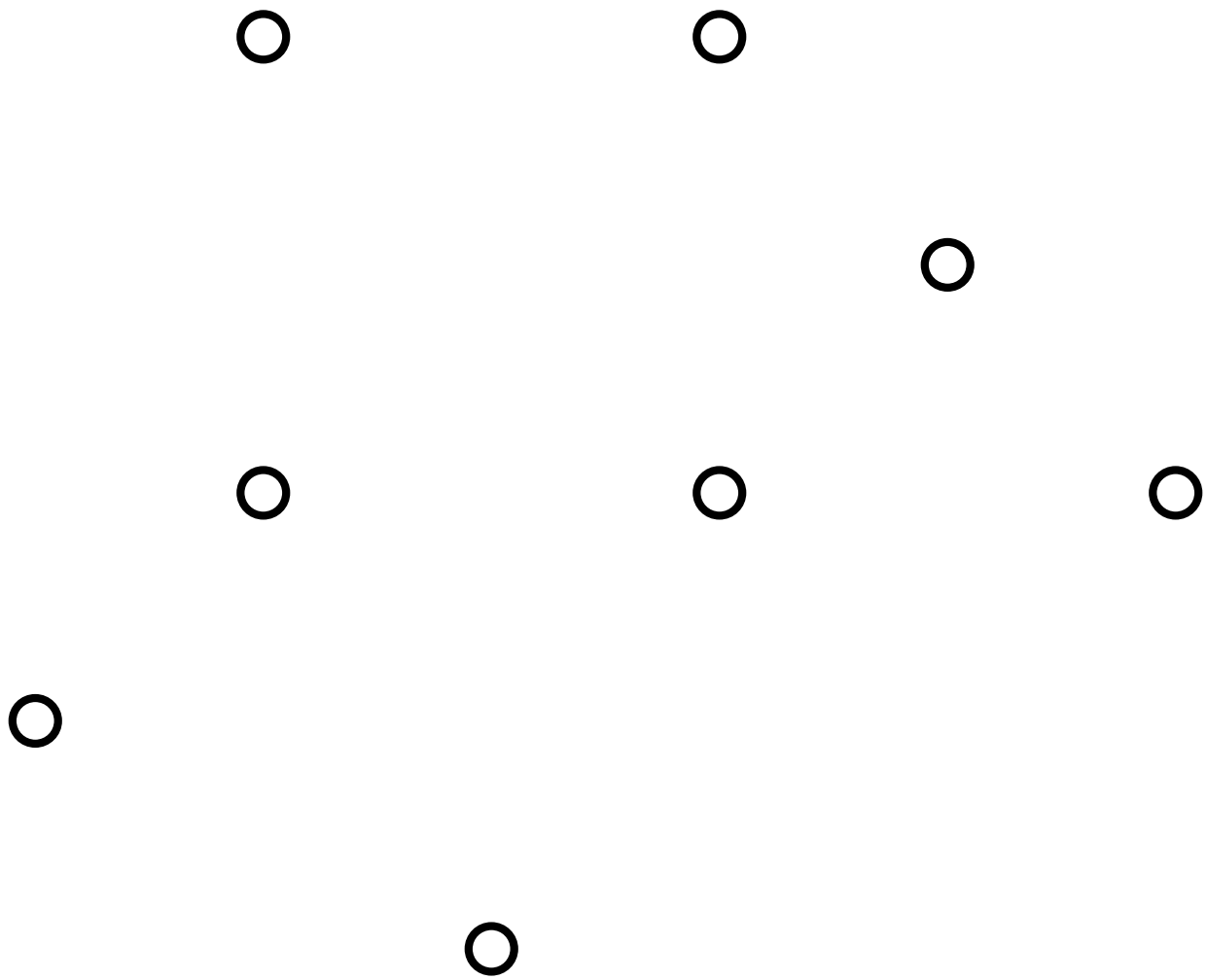
Задача коммивояжёра в метрическом пространстве

Задача коммивояжёра в метрическом пространстве (Metric TSP): частный случай для графов, веса рёбер которых удовлетворяют неравенству треугольника ($w(i,j) \leq w(i,k) + w(k,j)$).

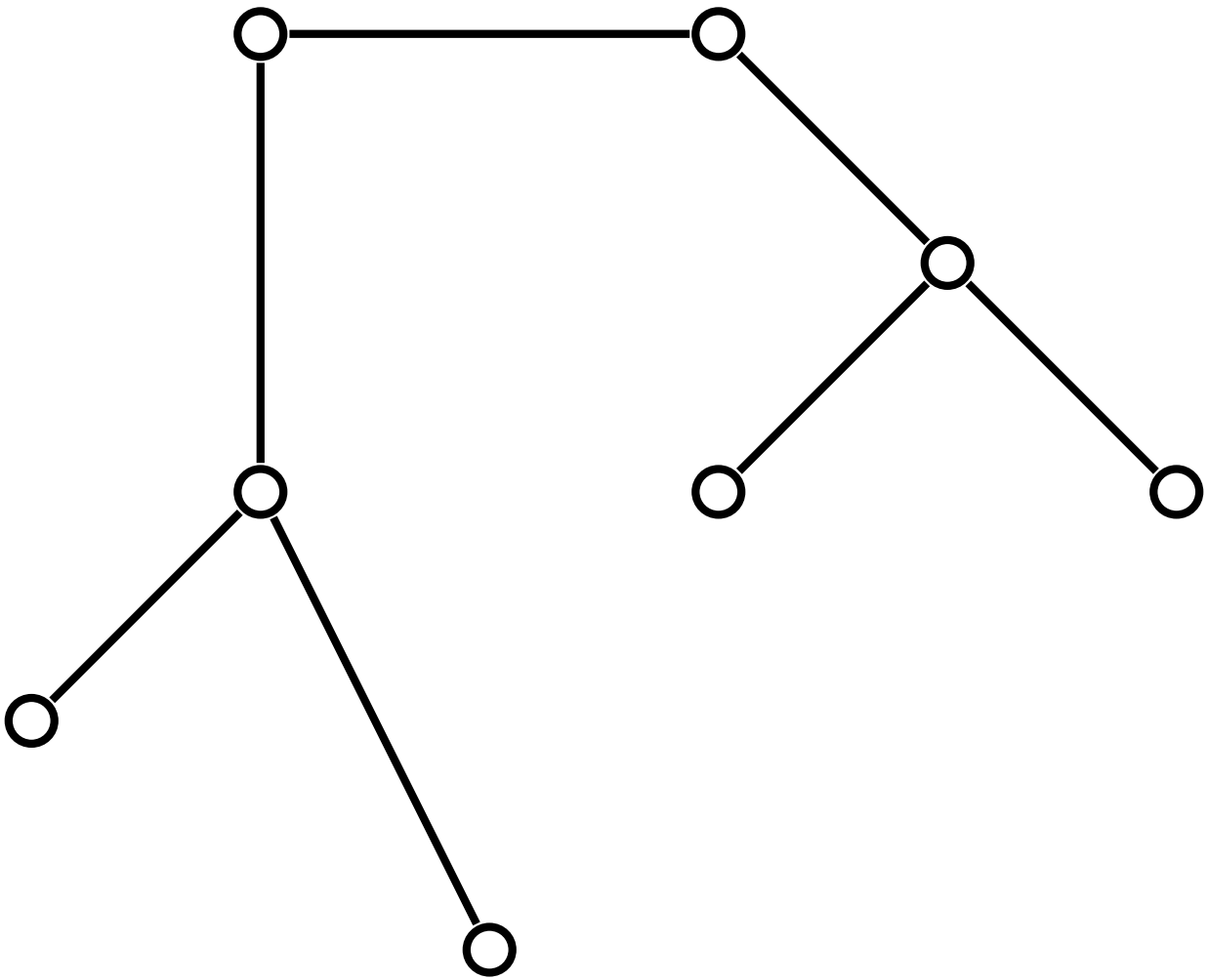
2-приближённый алгоритм

1. построить минимальное покрывающее дерево T
2. продублировать каждое ребро дерева T и
   в полученном графе найти эйлеров цикл
3. выкинуть из этого цикла все повторения вершин и
   вернуть полученный цикл

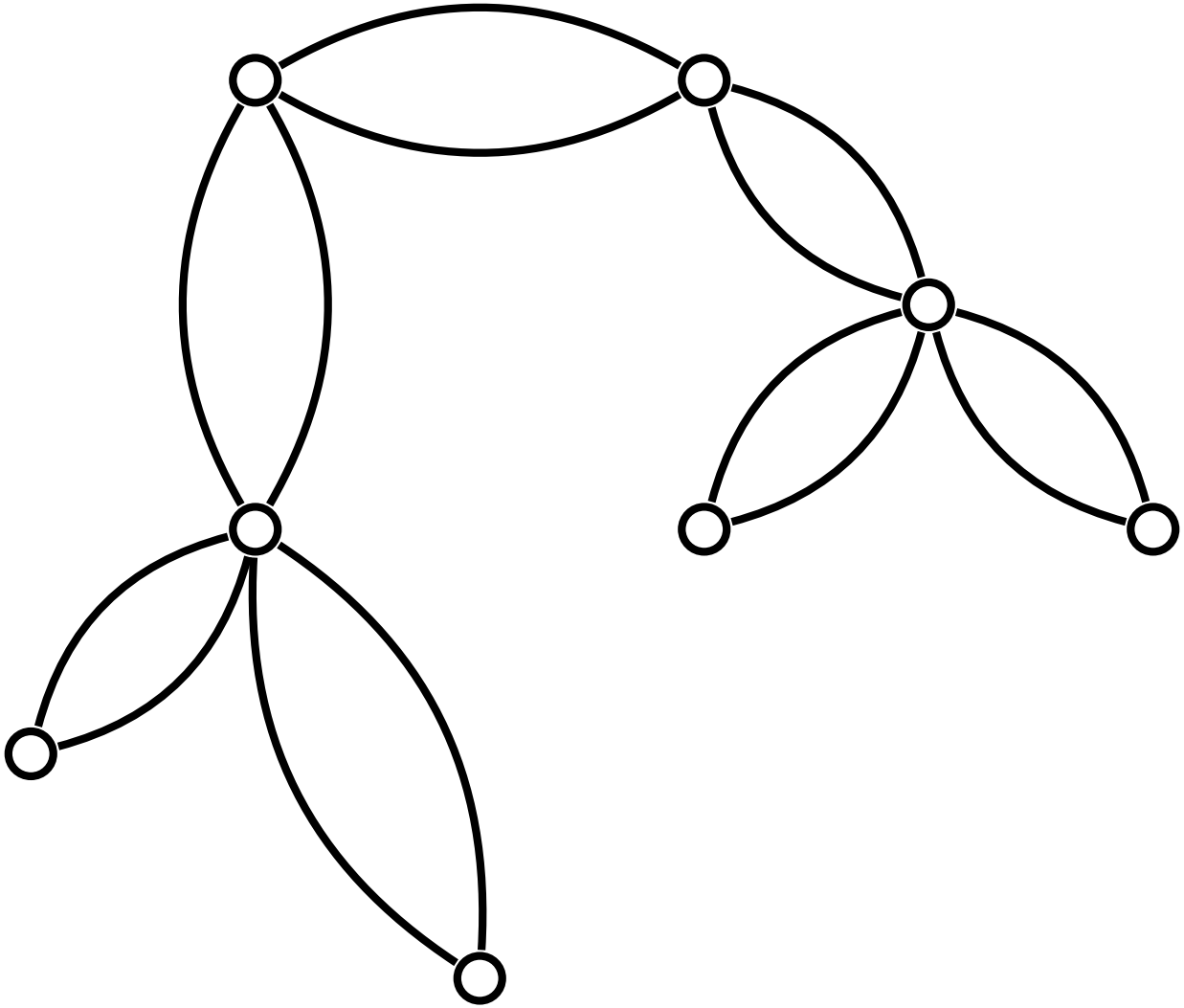
Пример



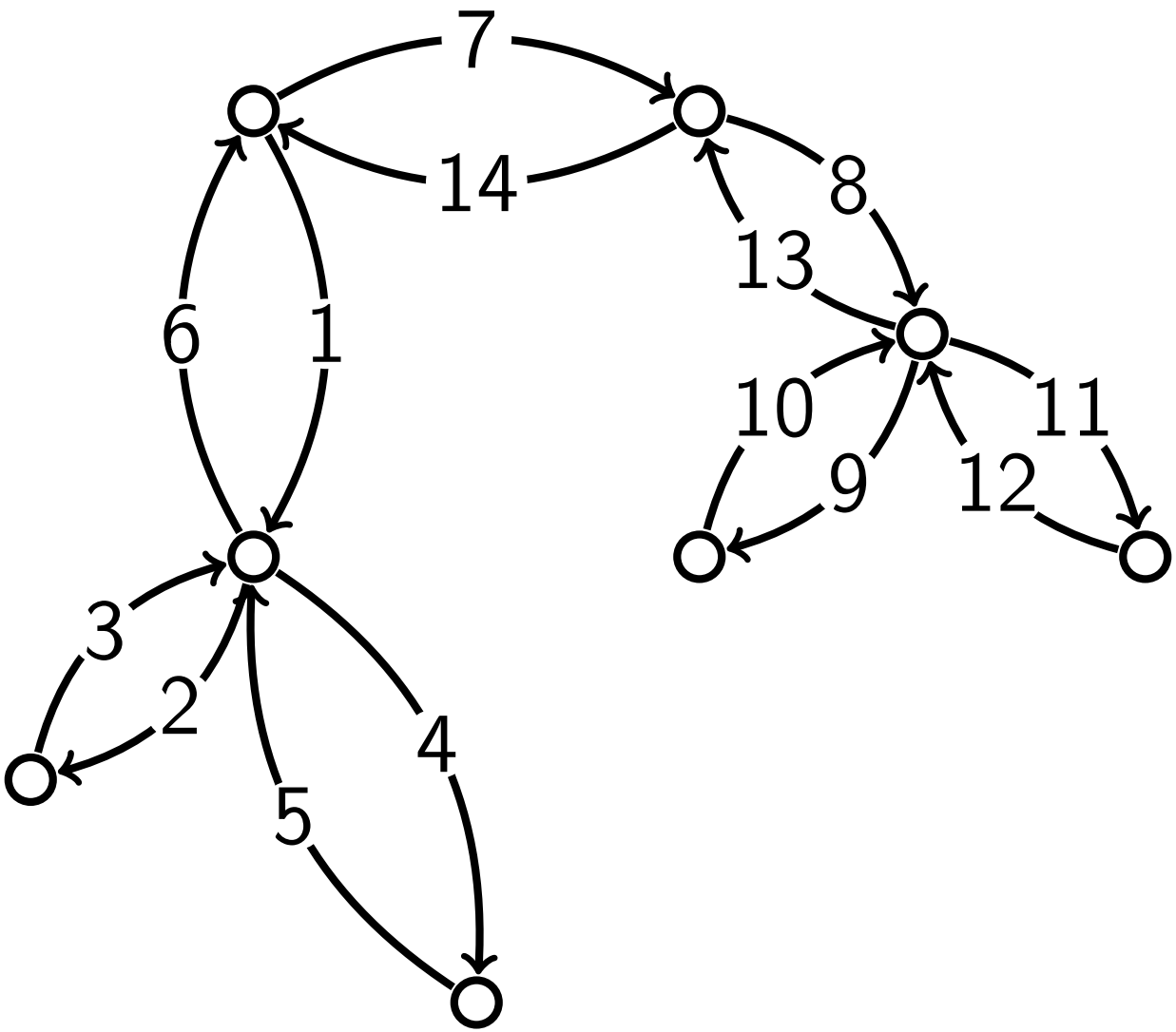
Пример



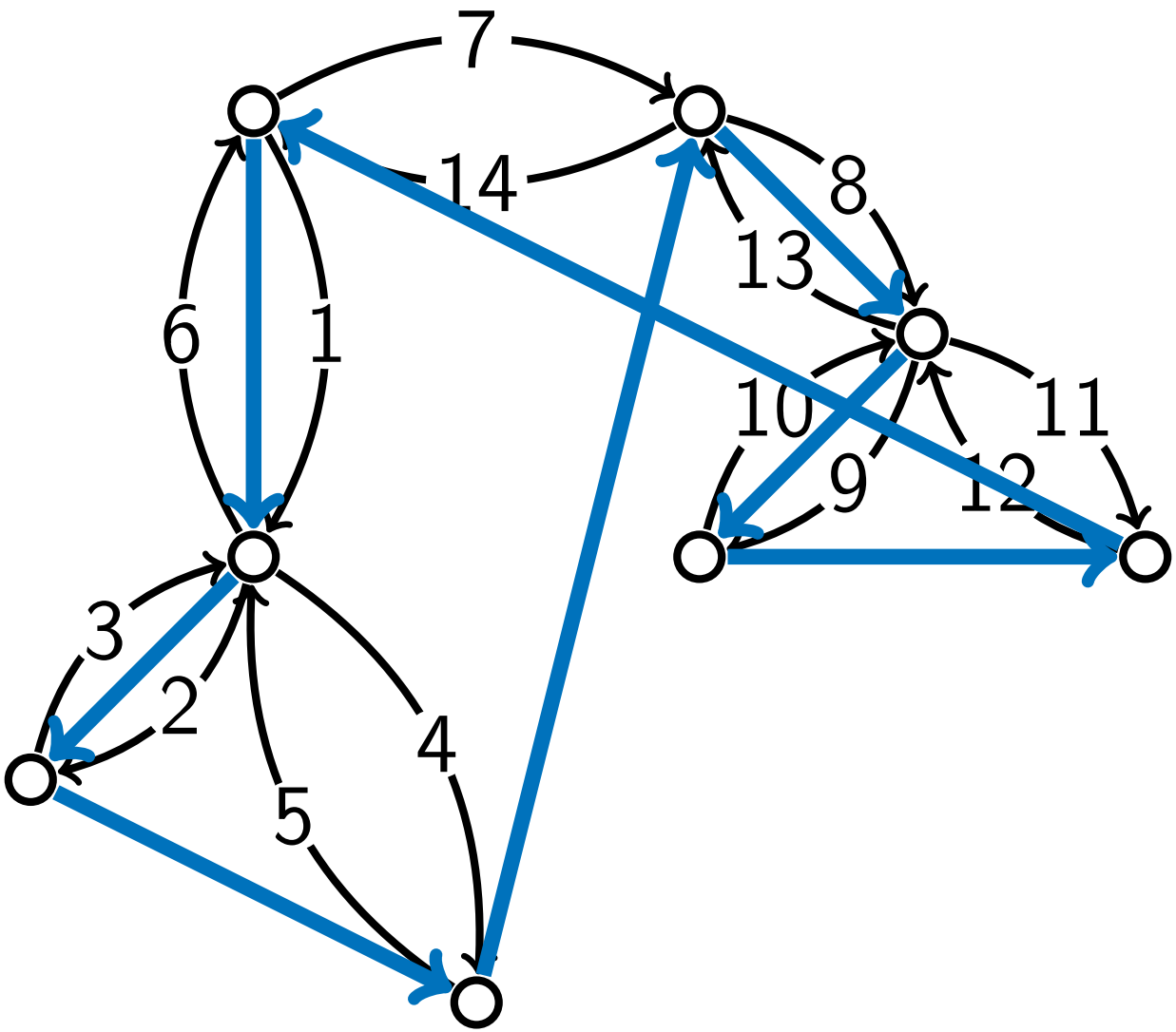
Пример



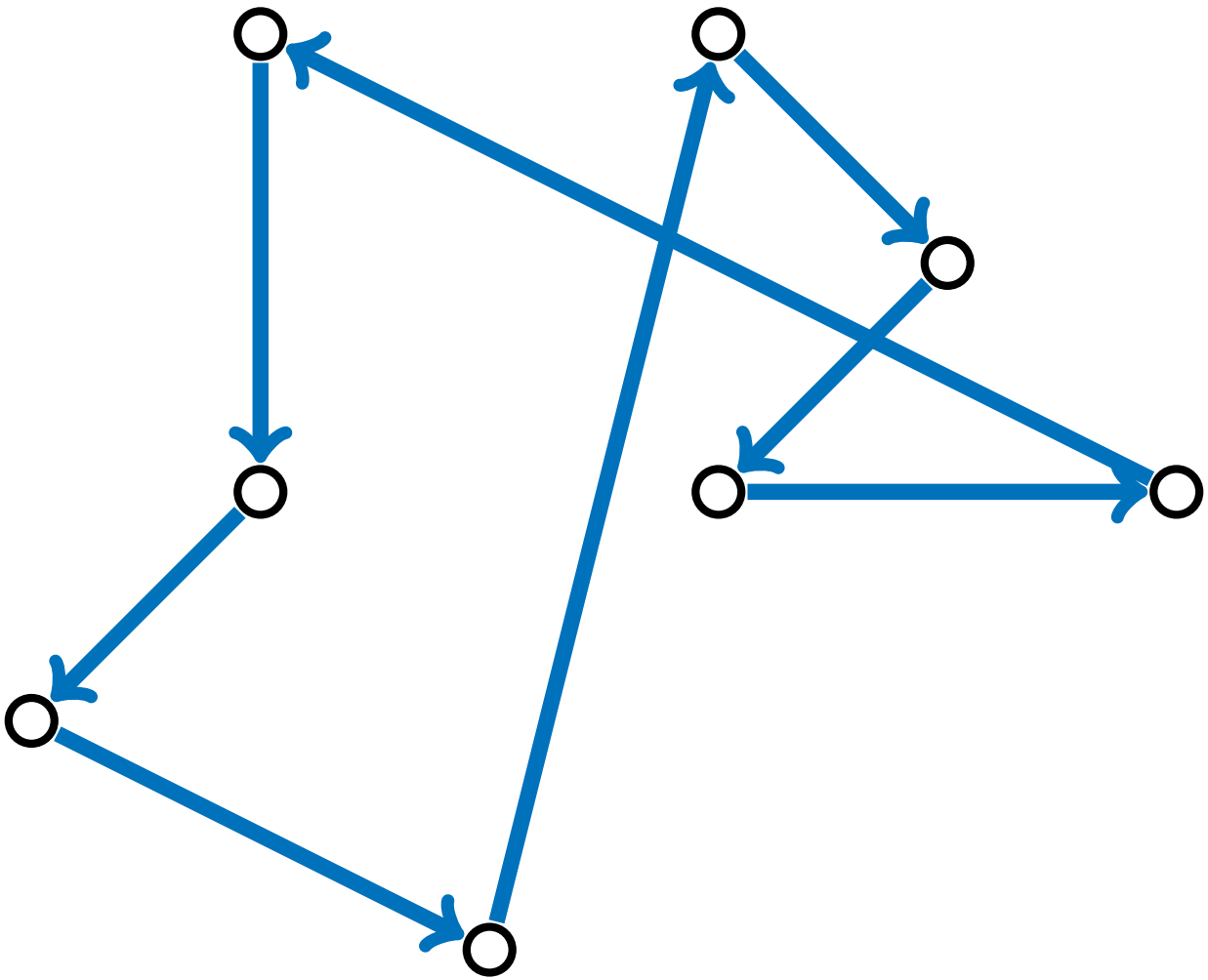
Пример



Пример



Пример



Доказательство

- пусть W_T — вес минимального остовного дерева, а W_{opt} — вес оптимального гамильтонова цикла

Доказательство

- пусть W_T — вес минимального остовного дерева, а W_{opt} — вес оптимального гамильтонова цикла
- $W_T \leq W_{\text{opt}}$, поскольку при выкидывании ребра из гамильтонва цикла получается остовное дерево

Доказательство

- пусть W_T — вес минимального остовного дерева, а W_{opt} — вес оптимального гамильтонова цикла
- $W_T \leq W_{\text{opt}}$, поскольку при выкидывании ребра из гамильтонва цикла получается остовное дерево
- каждое ребро построенного гамильтонова цикла заменяет какой-то путь эйлерова цикла, длина которого по неравенству треугольника не менее длины этого ребра

Доказательство

- пусть W_T — вес минимального остовного дерева, а W_{opt} — вес оптимального гамильтонова цикла
- $W_T \leq W_{\text{opt}}$, поскольку при выкидывании ребра из гамильтонва цикла получается остовное дерево
- каждое ребро построенного гамильтонова цикла заменяет какой-то путь эйлерова цикла, длина которого по неравенству треугольника не менее длины этого ребра
- значит, длина найденного пути не превосходит $2W_T$, а следовательно, и $2W_{\text{opt}}$ □

1.5-приближённый алгоритм

1. построить минимальное покрывающее дерево T
2. найти минимальное полное паросочетание всех вершин дерева T нечетной степени
3. добавить найденные рёбра в дерево T и найти в полученном графе эйлеров цикл
4. выкинуть из этого цикла все повторения вершин и вернуть полученный цикл

Доказательство

- как и в предыдущем доказательстве, вес построенного цикла не превосходит $W_T + W_P$, где W_P — вес минимального паросочетания вершин нечетной степени дерева T

Доказательство

- как и в предыдущем доказательстве, вес построенного цикла не превосходит $W_T + W_P$, где W_P — вес минимального паросочетания вершин нечетной степени дерева T
- нужно показать, что $W_P \leq W_{\mathrm{opt}}/2$

Доказательство

- как и в предыдущем доказательстве, вес построенного цикла не превосходит $W_T + W_P$, где W_P — вес минимального паросочетания вершин нечетной степени дерева T
- нужно показать, что $W_P \leq W_{\mathrm{opt}}/2$
- обозначим через A множество всех вершин нечётной степени дерева T

Доказательство

- как и в предыдущем доказательстве, вес построенного цикла не превосходит $W_T + W_P$, где W_P — вес минимального паросочетания вершин нечетной степени дерева T
- нужно показать, что $W_P \leq W_{\text{opt}}/2$
- обозначим через A множество всех вершин нечётной степени дерева T
- рассмотрим такой гамильтонов цикл на вершинах множества A: вершины множества A в нём будут встречаться в такой последовательности, в какой они идут в оптимальном гамильтоновом цикле графа G

Доказательство (продолжение)

- важно отметить, что нам не нужно строить такой цикл; нам важен лишь факт его существования

Доказательство (продолжение)

- важно отметить, что нам не нужно строить такой цикл; нам важен лишь факт его существования
- разбив вершины только что построенного цикла на чётные и нечётные, мы получим два паросочетания

Доказательство (продолжение)

- важно отметить, что нам не нужно строить такой цикл; нам важен лишь факт его существования
- разбив вершины только что построенного цикла на чётные и нечётные, мы получим два паросочетания
- вес хотя бы одного из них будет не более $W_{\mathrm{opt}}/2$

Доказательство (продолжение)

- важно отметить, что нам не нужно строить такой цикл; нам важен лишь факт его существования
- разбив вершины только что построенного цикла на чётные и нечётные, мы получим два паросочетания
- вес хотя бы одного из них будет не более $W_{\mathrm{opt}}/2$
- значит, и вес минимального паросочетания не превосходит $W_{\mathrm{opt}}/2$ $\square$

Содержание

Неприближаемость

- Предположим, что существует α-приближённый алгоритм для задачи коммивояжёра.

Неприближаемость

- Предположим, что существует α-приближённый алгоритм для задачи коммивояжёра.
- Возьмём тогда произвольный (невзвешенный и необязательно полный) граф и присвоим всем его рёбрам вес 1.

Неприближаемость

- Предположим, что существует α-приближённый алгоритм для задачи коммивояжёра.
- Возьмём тогда произвольный (невзвешенный и необязательно полный) граф и присвоим всем его рёбрам вес 1.
- Между любыми двумя не соединёнными ребром вершинами добавим ребро веса $\alpha n + 1$.

Неприближаемость

- Предположим, что существует α-приближённый алгоритм для задачи коммивояжёра.
- Возьмём тогда произвольный (невзвешенный и необязательно полный) граф и присвоим всем его рёбрам вес 1.
- Между любыми двумя не соединёнными ребром вершинами добавим ребро веса $\alpha n + 1$.
- Заметим теперь, что если в исходном графе существует гамильтонов цикл, то в новом графе существует гамильтонов цикл веса n.

Неприближаемость (продолжение)

- Если же такого цикла в исходном графе нет, то самый лёгкий цикл в новом графе имеет вес хотя бы $(\alpha n + 1) + (n - 1) > \alpha n$.

Неприближаемость (продолжение)

- Если же такого цикла в исходном графе нет, то самый лёгкий цикл в новом графе имеет вес хотя бы $(\alpha n + 1) + (n - 1) > \alpha n$.
- Таким образом, с помощью α-приближенного алгоритма для задачи о коммивояжёре мы можем понять, стоимость оптимального цикла в построенном графе превосходит n или нет.

Неприближаемость (продолжение)

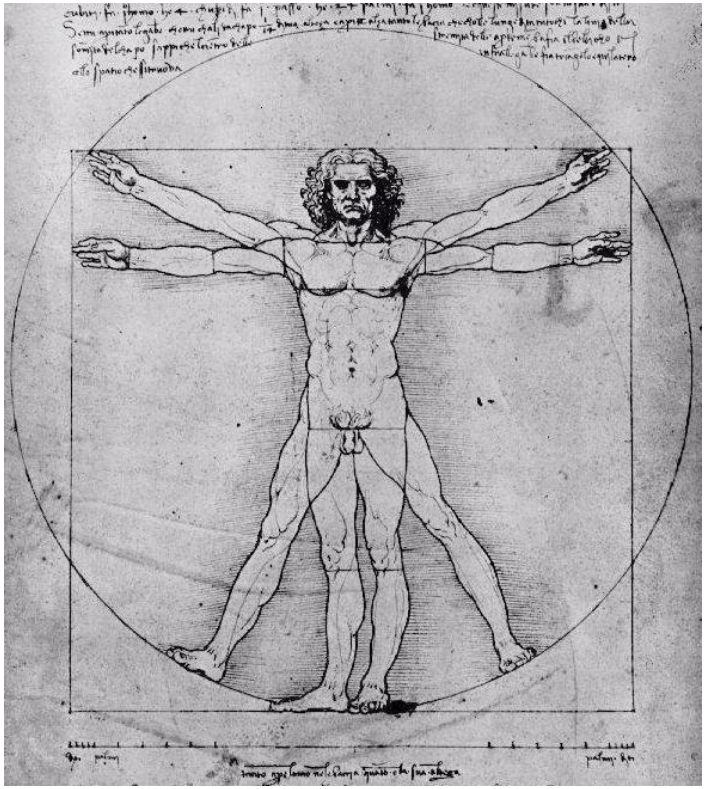
- Если же такого цикла в исходном графе нет, то самый лёгкий цикл в новом графе имеет вес хотя бы $(\alpha n + 1) + (n - 1) > \alpha n$.
- Таким образом, с помощью α-приближенного алгоритма для задачи о коммивояжёре мы можем понять, стоимость оптимального цикла в построенном графе превосходит n или нет.
- А это позволит нам понять (за полиномиальное время!), есть в исходном графе гамильтонов цикл или нет.

Неприближаемость (продолжение)

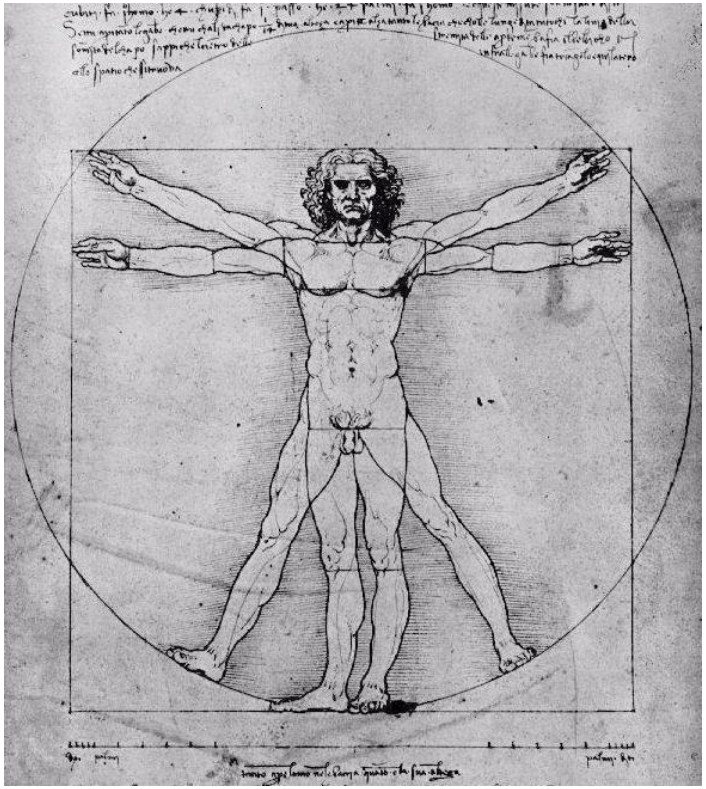
- Если же такого цикла в исходном графе нет, то самый лёгкий цикл в новом графе имеет вес хотя бы $(\alpha n + 1) + (n - 1) > \alpha n$.
- Таким образом, с помощью α-приближенного алгоритма для задачи о коммивояжёре мы можем понять, стоимость оптимального цикла в построенном графе превосходит n или нет.
- А это позволит нам понять (за полиномиальное время!), есть в исходном графе гамильтонов цикл или нет.
- Но тогда $\mathrm{P} = \mathrm{NP}$. □

Содержание

Теория и практика



Теория и практика



Camil Demetrescu. Engineering shortest path algorithms

Содержание

Динамическое программирование

- Подзадачи: для подмножества городов $S \subseteq \{1, 2, \ldots, n\}$, включающего 1, и $j \in S$, обозначим через $C[S, j]$ длину кратчайшего пути, начинающегося в 1 и заканчивающегося в j, проходящего через каждый город из множества S ровно один раз.

Динамическое программирование

- Подзадачи: для подмножества городов $S \subseteq \{1, 2, \ldots, n\}$, включающего 1, и $j \in S$, обозначим через $C[S, j]$ длину кратчайшего пути, начинающегося в 1 и заканчивающегося в j, проходящего через каждый город из множества S ровно один раз.
- Пересчёт: $C[S, j] = \min_{i \in S, i \neq j} \{C[S \setminus \{j\}, i] + d_{ij}\}$.

Псевдокод

1 $C[\{1\}, 1] \leftarrow 0$
2 **for** $s \leftarrow 2$ **to** n
3 **do for** всех $S \subseteq \{1, 2, \ldots, n\}$ размера s, содержащих 1
4 **do** $C[S, 1] \leftarrow \infty$
5 **for** всех $j \in S, j \neq 1$
6 **do** $C[S, j] \leftarrow \min_{i \in S, i \neq j} \{C[S \setminus \{j\}, i] + d_{ij}\}$
7 **return** $\min_j C[\{1, \ldots, n\}, j] + d_{j1}$

Сложность алгоритма

- Время работы данного алгоритма есть $O(n^2 2^n) = O^*(2^n)$.

Сложность алгоритма

- Время работы данного алгоритма есть $O(n^2 2^n) = O^*(2^n)$.
- Более того, памяти ему требуется тоже $O^*(2^n)$, что делает его совсем непрактичным.

Содержание

Формула включений-исключений

Пусть A — некоторое множество, $f, g\colon 2^A \to \mathbb{R}$, т.ч. $f(X) = \sum_{Y \subseteq X} g(Y)$. Тогда

$$g(X) = \sum_{Y \subseteq X} (-1)^{|X-Y|} f(Y)\,.$$

Формула включений-исключений

Пусть A — некоторое множество, $f, g\colon 2^A \to \mathbb{R}$, т.ч. $f(X) = \sum_{Y \subseteq X} g(Y)$. Тогда

$$g(X) = \sum_{Y \subseteq X} (-1)^{|X-Y|} f(Y).$$

Доказательство

$$\begin{aligned}\sum_{Y \subseteq X} (-1)^{|X-Y|} f(Y) &= \sum_{Y \subseteq X} \sum_{Z \subseteq Y} (-1)^{|X-Y|} g(Z) = \\ &= \sum_{Z \subseteq X} g(Z) \sum_{Z \subseteq Y \subseteq X} (-1)^{|X-Y|} = g(X)\end{aligned}$$

(последняя сумма равна 1, если $Z = X$, и нулю иначе). □

Задача о гамильтоновом пути

- Формулировка задачи: необходимо проверить, есть ли в данном графе простой путь, проходящий через все вершины, начинающийся в заданной вершине s и заканчивающийся в заданной вершине t.

Задача о гамильтоновом пути

- Формулировка задачи: необходимо проверить, есть ли в данном графе простой путь, проходящий через все вершины, начинающийся в заданной вершине s и заканчивающийся в заданной вершине t.
- Для $\{s, t\} \subseteq X \subseteq V$ обозначим через $g(X)$ количество путей (не обязательно простых! путь может проходить по некоторым вершинам несколько раз, а по некоторым вообще не проходить) длины $n - 1$ из s в t, проходящих только по вершинам множества X.

Задача о гамильтоновом пути

- Формулировка задачи: необходимо проверить, есть ли в данном графе простой путь, проходящий через все вершины, начинающийся в заданной вершине s и заканчивающийся в заданной вершине t.
- Для $\{s, t\} \subseteq X \subseteq V$ обозначим через $g(X)$ количество путей (не обязательно простых! путь может проходить по некоторым вершинам несколько раз, а по некоторым вообще не проходить) длины $n - 1$ из s в t, проходящих только по вершинам множества X.
- Нетрудно видеть, что значение $g(X)$ содержится в строке s и столбце t матрицы A^{n-1}, где A — матрица смежности графа $G[X]$.

Задача о гамильтоновом пути (продолжение)

- Пусть теперь $f(X)$ есть количество путей длины $n-1$ из s в t, проходящих по всем вершинам множества X. В частности, $f(V)$ есть количество гамильтоновых путей из s в t.

Задача о гамильтоновом пути (продолжение)

- Пусть теперь $f(X)$ есть количество путей длины $n-1$ из s в t, проходящих по всем вершинам множества X. В частности, $f(V)$ есть количество гамильтоновых путей из s в t.
- Тогда

$$f(V) = \sum_{Y \subseteq V} (-1)^{|V-Y|} g(Y)\,.$$

Задача о гамильтоновом пути (продолжение)

- Пусть теперь $f(X)$ есть количество путей длины $n-1$ из s в t, проходящих по всем вершинам множества X. В частности, $f(V)$ есть количество гамильтоновых путей из s в t.
- Тогда
$$f(V) = \sum_{Y \subseteq V} (-1)^{|V-Y|} g(Y)\,.$$
- Таким образом, количество гамильтоновых путей в графе может быть найдено за время $O^*(2^n)$ и полиномиальную память.

Задача о гамильтоновом пути (продолжение)

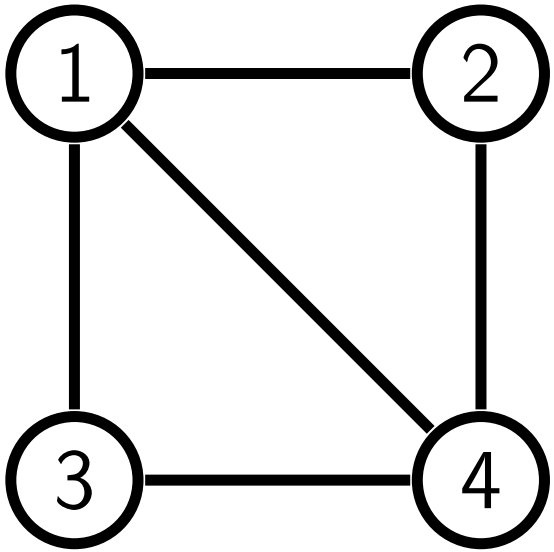
- Пусть теперь $f(X)$ есть количество путей длины $n-1$ из s в t, проходящих по всем вершинам множества X. В частности, $f(V)$ есть количество гамильтоновых путей из s в t.
- Тогда
$$f(V) = \sum_{Y \subseteq V} (-1)^{|V-Y|} g(Y)\,.$$
- Таким образом, количество гамильтоновых путей в графе может быть найдено за время $O^*(2^n)$ и полиномиальную память.
- Интересно отметить, что данный алгоритм переизобретался три раза.

Содержание

Замечание

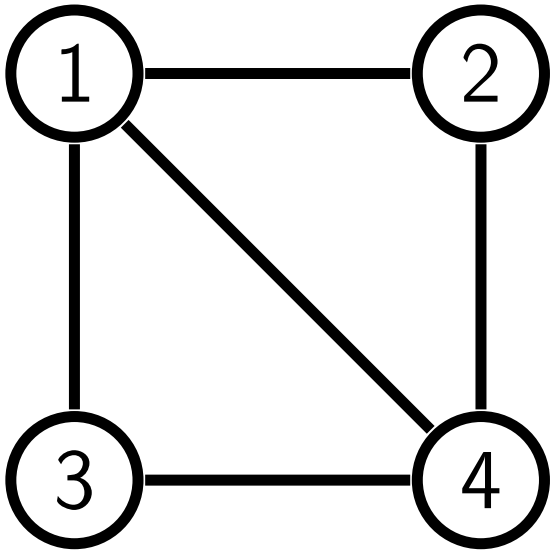
Далее мы рассмотрим алгоритм Бьорклунда для решения задачи о гамильтоновом цикле в двудольном графе за время $O^*(2^{n/2})$. Для общего случая задачи коммивояжёра оценка на время работы алгоритма Бьорклунда составляет $O^*(1.657^n \cdot W)$ (W — максимальный вес ребра).

Перманент матрицы Татта



$$\begin{pmatrix} & x_{12} & x_{13} & x_{14} \\ x_{12} & & & x_{24} \\ x_{13} & & & x_{34} \\ x_{14} & x_{24} & x_{34} & \end{pmatrix}$$

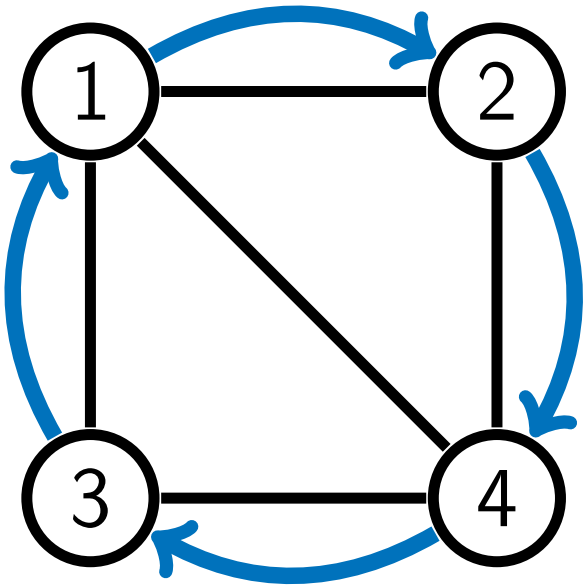
Перманент матрицы Татта

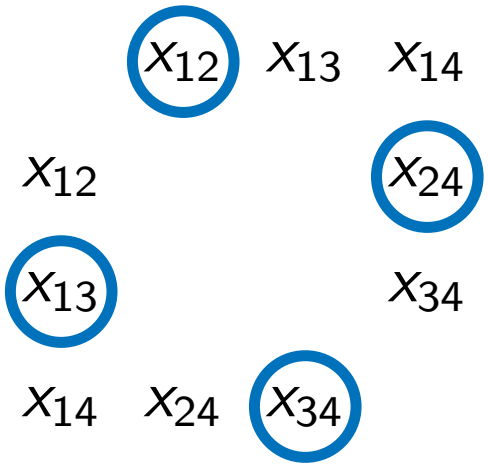


$$\begin{pmatrix} & x_{12} & x_{13} & x_{14} \\ x_{12} & & & x_{24} \\ x_{13} & & & x_{34} \\ x_{14} & x_{24} & x_{34} & \end{pmatrix}$$

$$\mathrm{perm}(M) =$$

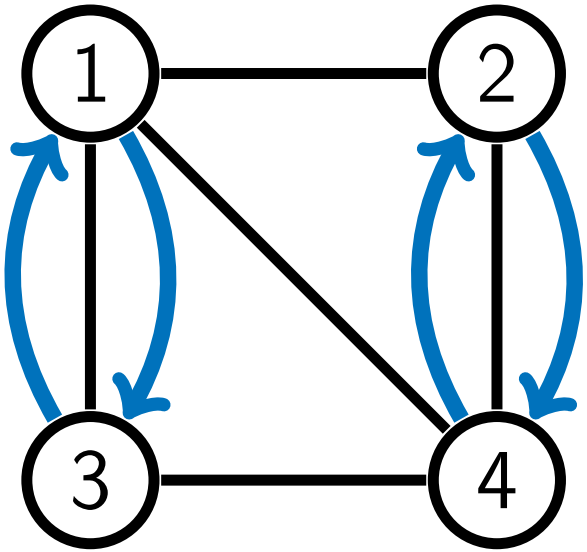
Перманент матрицы Татта

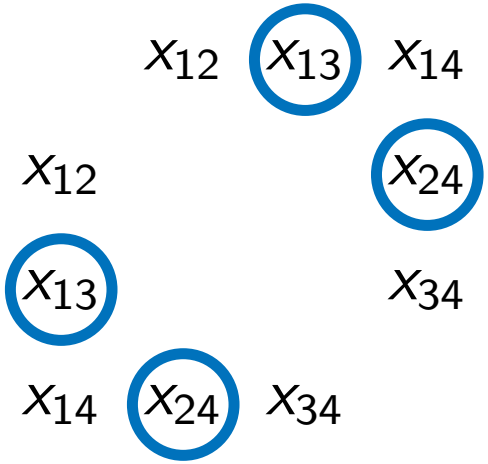




$$\mathrm{perm}(M) = x_{12}x_{24}x_{43}x_{31} + \dots$$

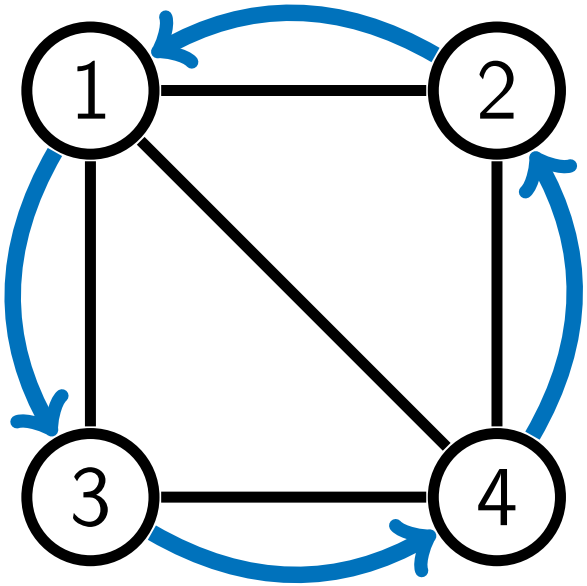
Перманент матрицы Татта

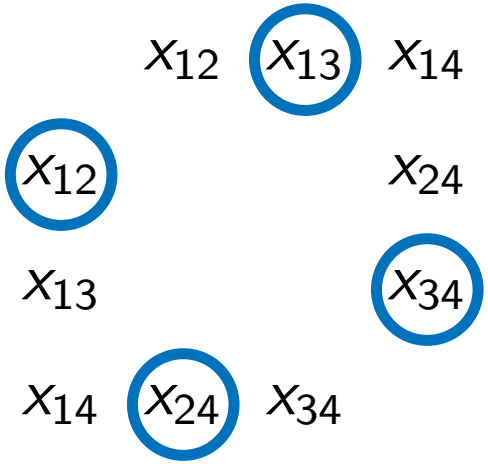




$$\mathrm{perm}(M) = x_{12}x_{24}x_{43}x_{31} + x_{13}^2x_{24}^2 + \dots$$

Перманент матрицы Татта





$$\mathrm{perm}(M) = x_{12}x_{24}x_{43}x_{31} + x_{13}^2x_{24}^2 + x_{12}x_{24}x_{43}x_{31} + \dots$$

Поле характеристики 2

- Если вычислять перманент над полем характеристики 2, то все циклы, не полностью состоящие из циклов длины 2, сократятся. Действительно, если в покрытии циклами есть цикл длины не 2, то возьмём первый из них (первый относительного какого-нибудь фиксированного порядка на вершинах) и обратим в нём все рёбра. Получим другое покрытие циклами, которому соответствует тот же самый моном.

Поле характеристики 2

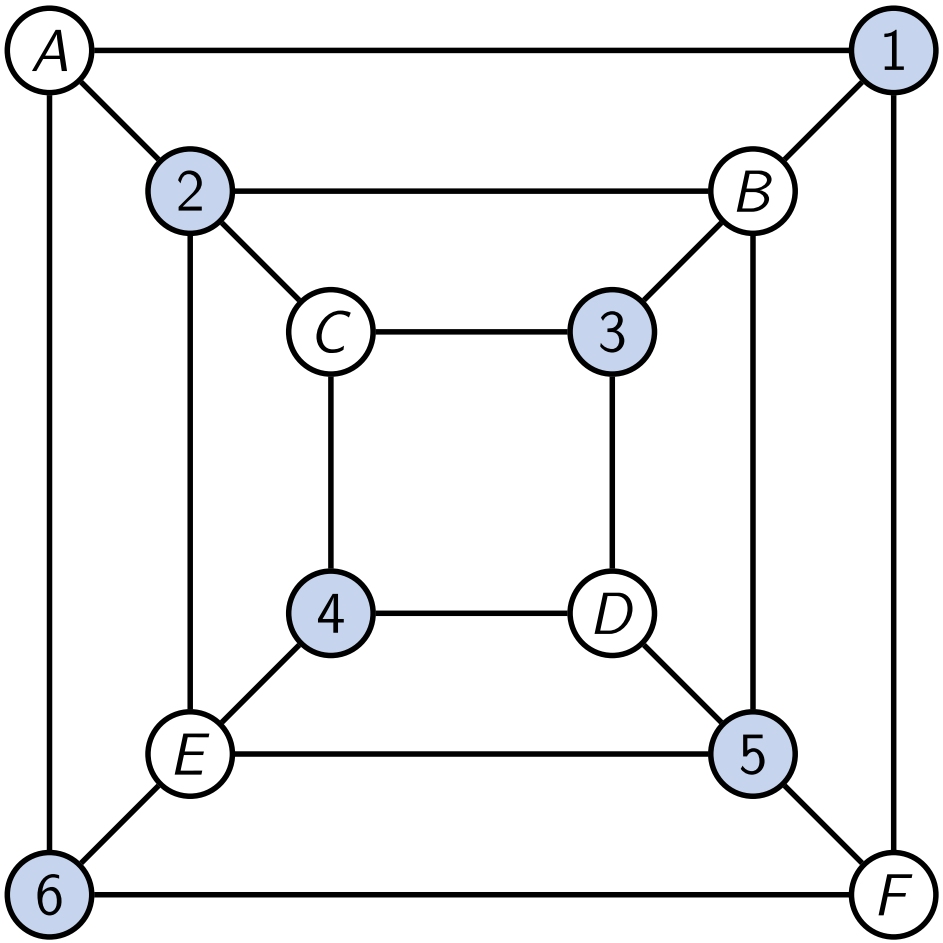
- Если вычислять перманент над полем характеристики 2, то все циклы, не полностью состоящие из циклов длины 2, сократятся. Действительно, если в покрытии циклами есть цикл длины не 2, то возьмём первый из них (первый относительного какого-нибудь фиксированного порядка на вершинах) и обратим в нём все рёбра. Получим другое покрытие циклами, которому соответствует тот же самый моном.
- Мы хотим исправить следующие два момента: во-первых, чтобы гамильтоновы циклы не сокращались, а во-вторых, чтобы покрытия с циклами длины 2 всё же пропадали.

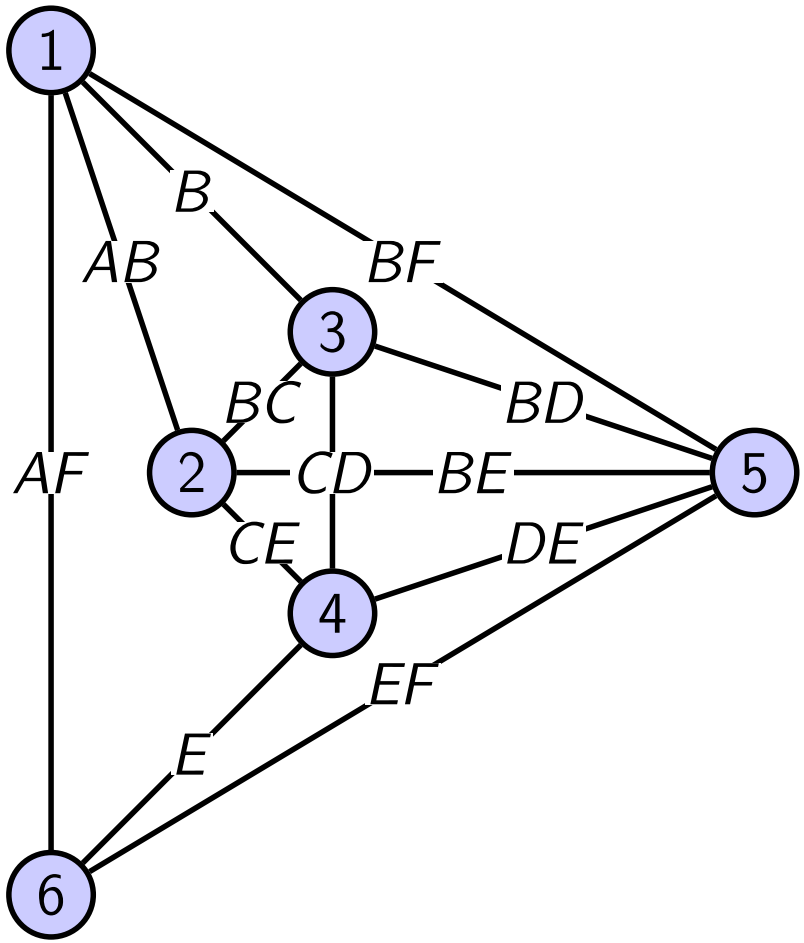
Первая цель: оставить гамильтоновы циклы

Сделаем вершину 1 графа выделенной: $T_G[1, j] = x_{1j}$, но $T_G[j, 1] = x_{j1}$ для ребра $\{1, j\} \in E$. Тогда каждому гамильтонову циклу будут соответствовать два разных монома.

$$
\begin{array}{cccc}
 & x_{12} & x_{13} & x_{14} \\
x_{21} & & & x_{24} \\
x_{31} & & & x_{34} \\
x_{41} & x_{24} & x_{34} &
\end{array}
$$

Вторая цель: сократить всё остальное





Почему же всё сократится?

- В новом графе нам нужен помеченный гамильтонов цикл.

Почему же всё сократится?

- В новом графе нам нужен помеченный гамильтонов цикл.
- В матрице Татта теперь будут помеченные переменные: вместо x_{24} будет $x_{24,C} + x_{24,E}$.

Почему же всё сократится?

- В новом графе нам нужен помеченный гамильтонов цикл.
- В матрице Татта теперь будут помеченные переменные: вместо x_{24} будет $x_{24,C} + x_{24,E}$.
- При вычислении над полем характеристики 2 гамильтоновы циклы по-прежнему не сократятся (из-за специальной переменной 1).

Почему же всё сократится?

- В новом графе нам нужен помеченный гамильтонов цикл.
- В матрице Татта теперь будут помеченные переменные: вместо x_{24} будет $x_{24,C} + x_{24,E}$.
- При вычислении над полем характеристики 2 гамильтоновы циклы по-прежнему не сократятся (из-за специальной переменной 1).
- Всё остальное:

Почему же всё сократится?

- В новом графе нам нужен помеченный гамильтонов цикл.
- В матрице Татта теперь будут помеченные переменные: вместо x_{24} будет $x_{24,C} + x_{24,E}$.
- При вычислении над полем характеристики 2 гамильтоновы циклы по-прежнему не сократятся (из-за специальной переменной 1).
- Всё остальное:
  - покрытия циклами, в которых используются не все пометки — сократятся по формуле включений-исключений (рассмотрим все $2^{n/2}$ подмножеств пометок);

Почему же всё сократится?

- В новом графе нам нужен помеченный гамильтонов цикл.
- В матрице Татта теперь будут помеченные переменные: вместо x_{24} будет $x_{24,C} + x_{24,E}$.
- При вычислении над полем характеристики 2 гамильтоновы циклы по-прежнему не сократятся (из-за специальной переменной 1).
- Всё остальное:
  - покрытия циклами, в которых используются не все пометки — сократятся по формуле включений-исключений (рассмотрим все $2^{n/2}$ подмножеств пометок);
  - негамилтьтоновы покрытия циклами, в которых есть все пометки, разобьются на пары и сократятся (из-за пометок).

Литература


A. Björklund.
Determinant sums for undirected Hamiltonicity
Proc. 51st IEEE Symposium on Foundations of Computer Science (FOCS '10), pp. 173–182.


M. Held, R. M. Karp
A Dynamic Programming Approach to Sequencing Problems
Journal of the Society for Industrial and Applied Mathematics 10 (1): 196–210.


Paluch, K., Elbassioni, K., Zuylen, A. van.
Simpler Approximation of the Maximum Asymmetric Traveling Salesman Problem.
STACS' 12.

Спасибо!

Спасибо за внимание!