Дискретная оптимизация
Простые классы задач

Роланд Хильдебранд
МФТИ, Лаборатория ММО

Институт Вычислительной Математики им. Г.И. Марчука РАН

10 марта 2026 г.

План

- **введение**
- простые комбинаторные проблемы

Комбинаторные задачи

- маршрутизация
- планирование
- упаковка

планирование

- рабочих смен
- производства
- транспортных средств
- ремонтных работ / тех. обслуживания

упаковка

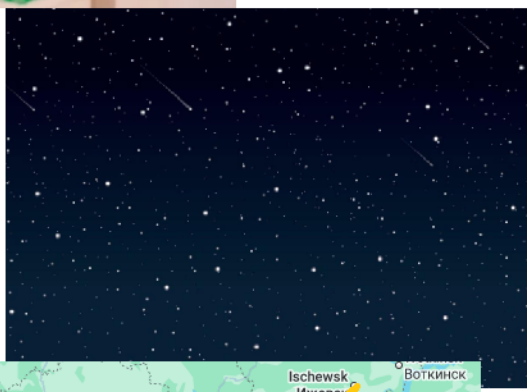
- грузов в контейнеры
- виртуальных машин на сервера
- деталей на листах прокатного материала

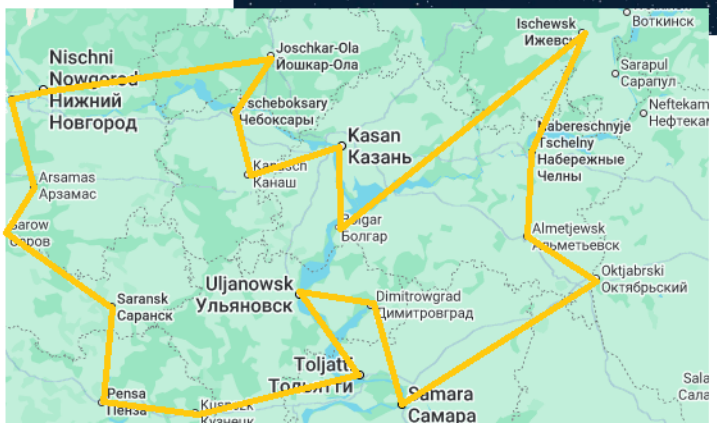
Задачи маршрутизации

найти *маршрут* для одного или многих *агентов*, посещающих локации

нужно минимизировать суммарную стоимость (например, пройденное расстояние)

- задача коммивояжёра (TSP): один агент должен посетить все локации
- маршрутизация транспортных средств (VRP): несколько агентов должны посетить все локации





Задачи маршрутизации

локациями могут быть

- объекты для ремонта / тех. обслуживания
- астрономические / космические объекты для наблюдения / слежения
- адреса заказов для курьерской службы
- магазины, снабжаемые со складов
- точки, посещаемые роботом

Комбинаторные задачи

общий вид задачи

$$\min_{x \in X} f(x)$$

где $f : X \to \mathbb{R}$ и допустимое множество X — **конечно**

обычно мощность X очень большая
перебор не практичен

методы решения

- мета-эвристики (на общих принципах, применимы к широкому спектру задач)
- специализированные эвристики (например, жадные алгоритмы)
- непрерывные / выпуклые релаксации (например, через формулировку в виде смешанно-целочисленной ЛП)

Классы задач

простые комбинаторные задачи

- 2-SAT
- Horn-SAT
- кратчайший путь
- максимальный поток
- наибольшее паросочетание
- хордальность / порядок совершенного исключения
- сепарирующая клика

сложные комбинаторные задачи

- выполнимость булевых формул
- рюкзак
- задача об упаковке
- покрытие подмножествами
- покраска графа
- максимальный разрез
- наибольшая клика

План

простые комбинаторные задачи

- **2-SAT**
- Horn-SAT
- кратчайший путь
- максимальный поток
- наибольшее паросочетание
- хордальность / порядок совершенного исключения
- сепарирующая клика

Булевы формулы

булева формула с логическими $(0,1)$-переменными $x_1, \ldots, x_n$ — выражение, задающее функцию

$$f : \{0,1\}^n \to \{0,1\}$$

использует

- скобки ()
- отрицания $\neg$
- конъюнкции $\wedge$
- дизъюнкции $\vee$

в бинарной записи

$$\neg x = 1 - x, \;\; x \wedge y = \max(0, x + y - 1), \;\; x \vee y = \min(1, x + y)$$

Конъюктивная нормальная форма

формула в *КНФ*

- не содержит вложенных друг в друга скобок
- дизъюнкции — только внутри скобок
- конъюнкции — только вне скобок

пример:

$$(x_1 \vee \neg x_2) \wedge \neg x_3 \wedge (\neg x_1 \vee x_3 \vee \neg x_4)$$

k-конъюнктивная нормальная форма:

в скобках стоят дизъюнкции ровно k выражений типа x_i или $\neg x_i$

пример:

$$(x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_1 \vee x_3 \vee x_4)$$

можно считать, что в каждой скобке *не более* k выражений например, $x_1 \vee \neg x_2 = x_1 \vee x_1 \vee \neg x_2$

Задача k-SAT

пусть задана функция f в k-конъюнктивной нормальной форме

нужно

- либо найти значения переменных $x_1, \ldots, x_n$ такие, что $f(x) = 1$
- либо выявить, что таких значений нет

имеем $f(x) = 1$ если хотя бы одно из k выражений в каждой скобке принимает значение 1

Задача 2-SAT

пусть в задаче n переменных $x_1, \ldots, x_n$

определим направленный граф $G = (V, E)$, где

$$V = \{x_1, \ldots, x_n, \neg x_1, \ldots, \neg x_n\}$$

пусть $y \vee z$ — скобка формулы в КНФ, где y и z равны x_i или $\neg x_i$ для некоторых i

для каждой скобки добавим в граф два ребра $(\neg y, z), (\neg z, y)$

рёбра графа представляют импликации, которые должны быть истинны при значениях переменных $x_1, \ldots, x_n$ в случае, когда сама формула истинна

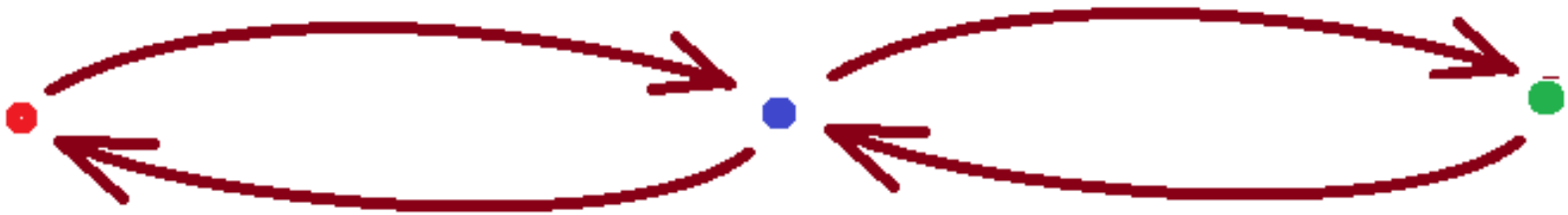
- если есть путь от x_i к $\neg x_i$, то $f(x) = 1$ влечёт $x_i = 0$
- если есть путь от $\neg x_i$ к x_i, то $f(x) = 1$ влечёт $x_i = 1$
- если есть пути в обе стороны, то формула невыполнима

Задача 2-SAT

Определение

Пусть G — направленный граф. Компонентой сильной связности называется максимальное подмножество вершин такое, что от любой вершины подмножества к любой другой вершины подмножества есть путь.

корректность следует из транзитивности



если есть путь из красной в синюю, и из синей в зелёную вершину, то есть путь из красной в зелёную, и наоборот

Задача 2-SAT

граф распадается на дизъюнктные компоненты сильной связности

если $f(x) = 1$, то все вершины каждой компоненты должны иметь одно и то же значение

формула выполнима тогда и только тогда, когда ни одна пара $(x_i, \neg x_i)$ не находится в одной и той же компоненте [Aspvall, Plass, Tarjan 1979]

граф можно разложить на компоненты за полиномиальное (линейное) время (поиском вглубь, [Tarjan, 1979])

Задача 2-SAT

алгоритм построения x [Aspvall, Plass, Tarjan 1979]

- разложить граф на компоненты сильной связности
- построить *конденсированный* ацикличный направленный граф, в котором вершины — компоненты исходного
- топологически упорядочить вершины конденсированного графа (порядок уже построен алгоритмом разложения на компоненты)
- начиная с конца, приравнивать компоненты 1, а комплементы 0

корректность построения

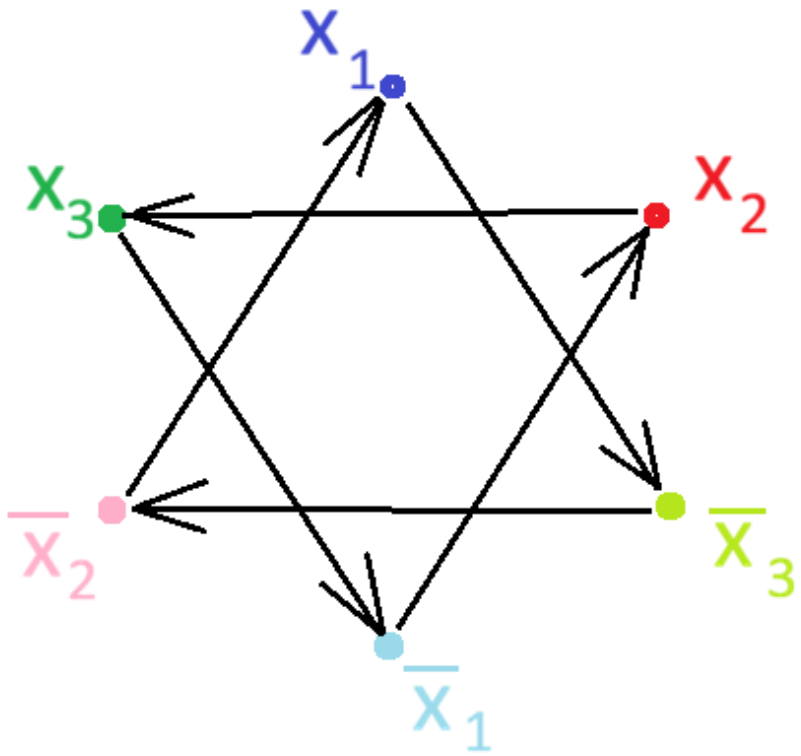
- все следующие из истинной компоненты компоненты уже обозначены как истинные
- все компоненты, имплицирующие ложную компоненту, уже были назначены ложными

Задача 2-SAT

пример: рассмотрим формулу

$$(x_1 \vee x_2) \wedge (\neg x_2 \vee x_3) \wedge (\neg x_1 \vee \neg x_3)$$

получаем направленный граф



Задача 2-SAT

две компоненты сильной связности

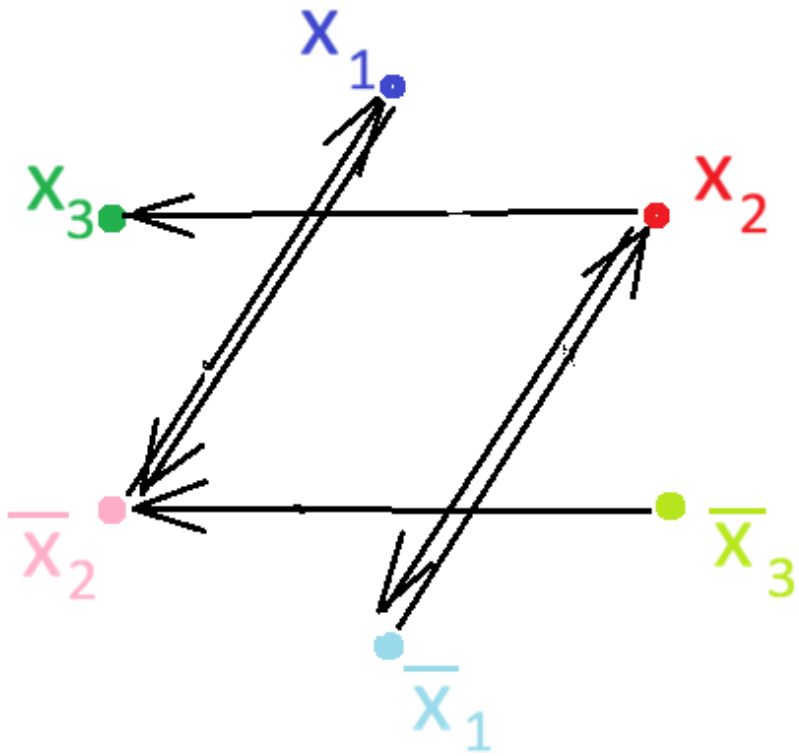
любой из компонент можно присвоить значение 1, тогда другая примет значение 0

Задача 2-SAT

пример: рассмотрим формулу

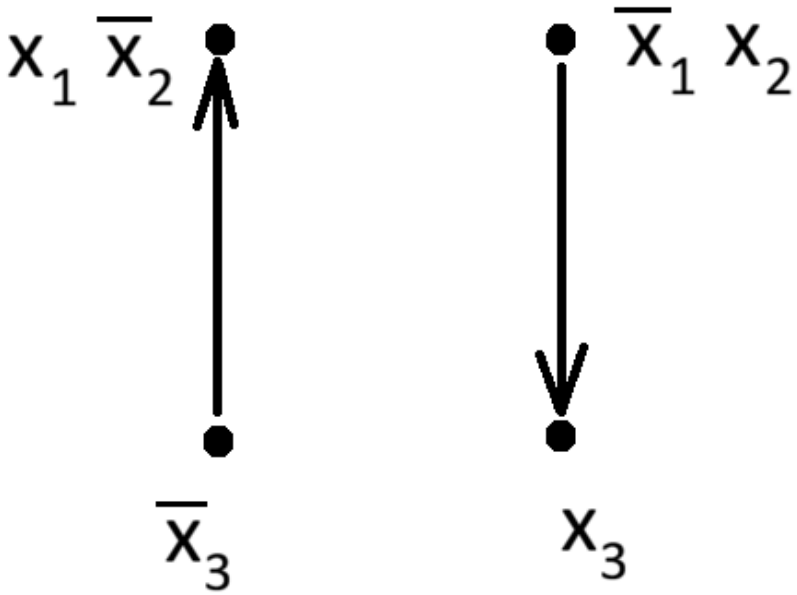
$$(x_1 \vee x_2) \wedge (\neg x_2 \vee x_3) \wedge (\neg x_1 \vee \neg x_2)$$

получаем направленный граф



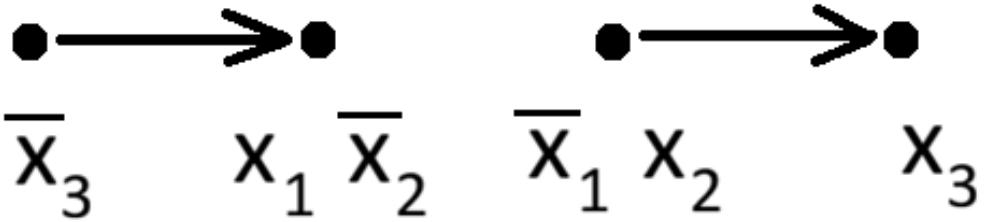
Задача 2-SAT

4 компоненты сильной связности



Задача 2-SAT

упорядочиваем вершины

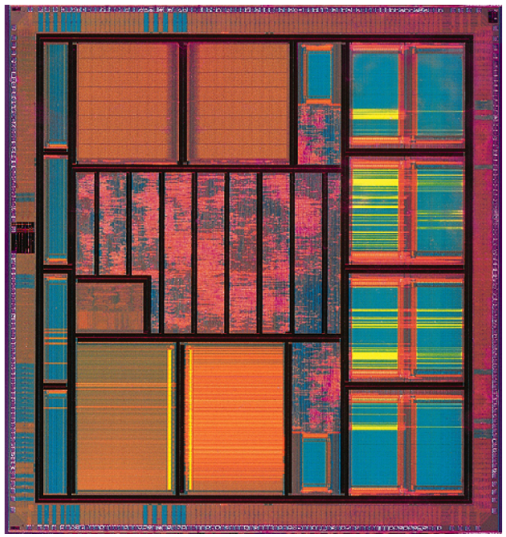


начинаем с конца

$$x_3 = 1, \quad \bar{x}_3 = 0$$

$$\bar{x}_1 = x_2 = 1, \quad x_1 \bar{x}_2 = 0$$

Приложения 2-SAT



любые ситуации, когда

- для каждого из n объектов нужно выбрать одно из двух состояний
- недопустимые комбинации задаются конфликтным графом

пример: размещение объектов

- на карте
- на плате

План

простые комбинаторные задачи

- 2-SAT
- **Horn-SAT**
- кратчайший путь
- максимальный поток
- наибольшее паросочетание
- хордальность / порядок совершенного исключения
- сепарирующая клика

Задача Horn-SAT

пусть в булевой формуле $f(x)$ в КНФ в каждой скобке

- любое количество выражений
- из них не более одного вида x_i

задача установления выполнимости такой формулы — класса Horn-SAT

простой рекурсивный полиномиальный алгоритм

- если во всех скобках присутствуют выражения вида $\neg x_i$, то $f(0) = 1$ (формула выполняется назначением $x = 0$)
- если есть пустая скобка, то $f(x) = 0$ для всех x (формула невыполнима)
- в ином случае есть скобка вида (x_i) — тогда можно положить $x_i = 1$ и сократить число переменных

задача Horn-SAT — P-полная (любая задача в P сводится к инстанции задачи Horn-SAT [Cook, Nguyen 2010]

Задача Horn-SAT

сложность рекурсивного алгоритма — $O(m^2)$, где m — суммарное число выражений во всех скобках

есть *линейные* алгоритмы [Dowling, Gallier 1984]

строим *направленный* граф с множеством вершин

$$V = \{x_1, \ldots, x_n, 0, 1\}$$

и *маркированными* рёбрами

маркировки рёбер — индексы скобок в формуле f

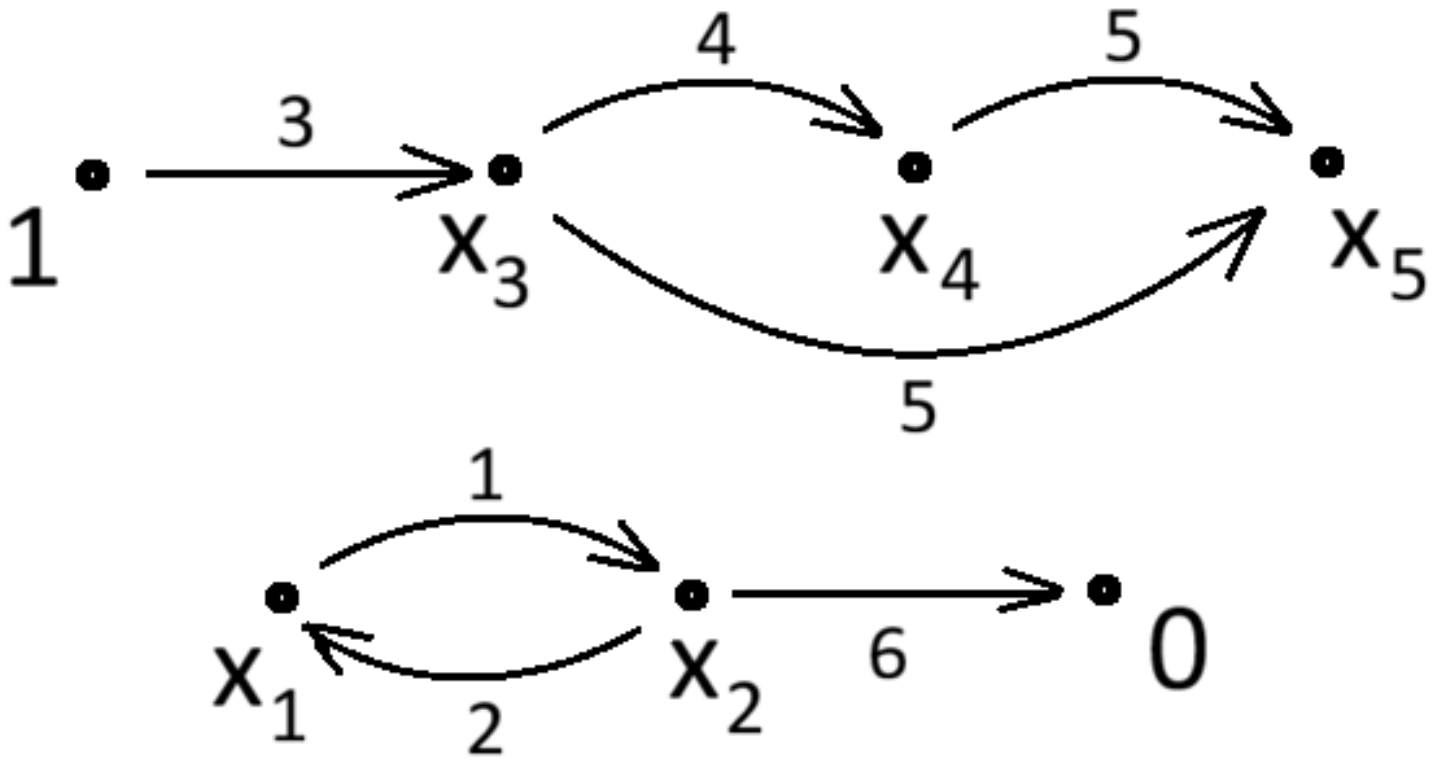
- скобка (x_i) маркирует ребро $1 \to x_i$
- скобка $(\neg x_{i_1} \vee \cdots \vee \neg x_{i_k})$ маркирует k рёбер вида $x_{i_j} \to 0$
- скобка $(\neg x_{i_1} \vee \cdots \vee \neg x_{i_k} \vee x_i)$ маркирует k рёбер вида $x_{i_j} \to x_i$

Задача Horn-SAT

пример: рассмотрим формулу

$$f(x) = (\neg x_1 \vee x_2) \wedge (x_1 \vee \neg x_2) \wedge x_3 \wedge (\neg x_3 \vee x_4) \wedge (\neg x_3 \vee \neg x_4 \vee x_5) \wedge \neg x_2$$

ей соответствует граф



Задача Horn-SAT

пусть известно, что подмножество $I \subset \{1, \ldots, n\}$ булевых переменных имеют значение 1

Какие ещё переменные заведомо примут значение 1?

$x_j = 1$ если

- $j \in I$
- существует скобка (x_j)
- существует скобка $(\neg x_{i_1} \vee \cdots \vee \neg x_{i_k} \vee x_j)$, и $x_{i_1} = \cdots = x_{i_k} = 1$

два последних правила соответствуют следующей ситуации:

существует индекс i такой, что все маркированные индексом i рёбра показывают на i и происходят от переменных со значением 1

таким образом значение 1 можно "распространять" по графу, начиная с определённого подмножества вершин

Задача Horn-SAT

Теорема

Пусть f — булева формула в задаче Хорна, G — соответствующий направленный граф. Тогда следующие утверждения эквивалентны:

- *формула невыполнима*
- *начиная с подмножества $\{1\}$, значение 1 можно распространить на вершину $\{0\}$*

Если условия не удовлетворяются, то выполняющее формулу назначение можно получить распространением значения 1 с подмножества $\{1\}$.

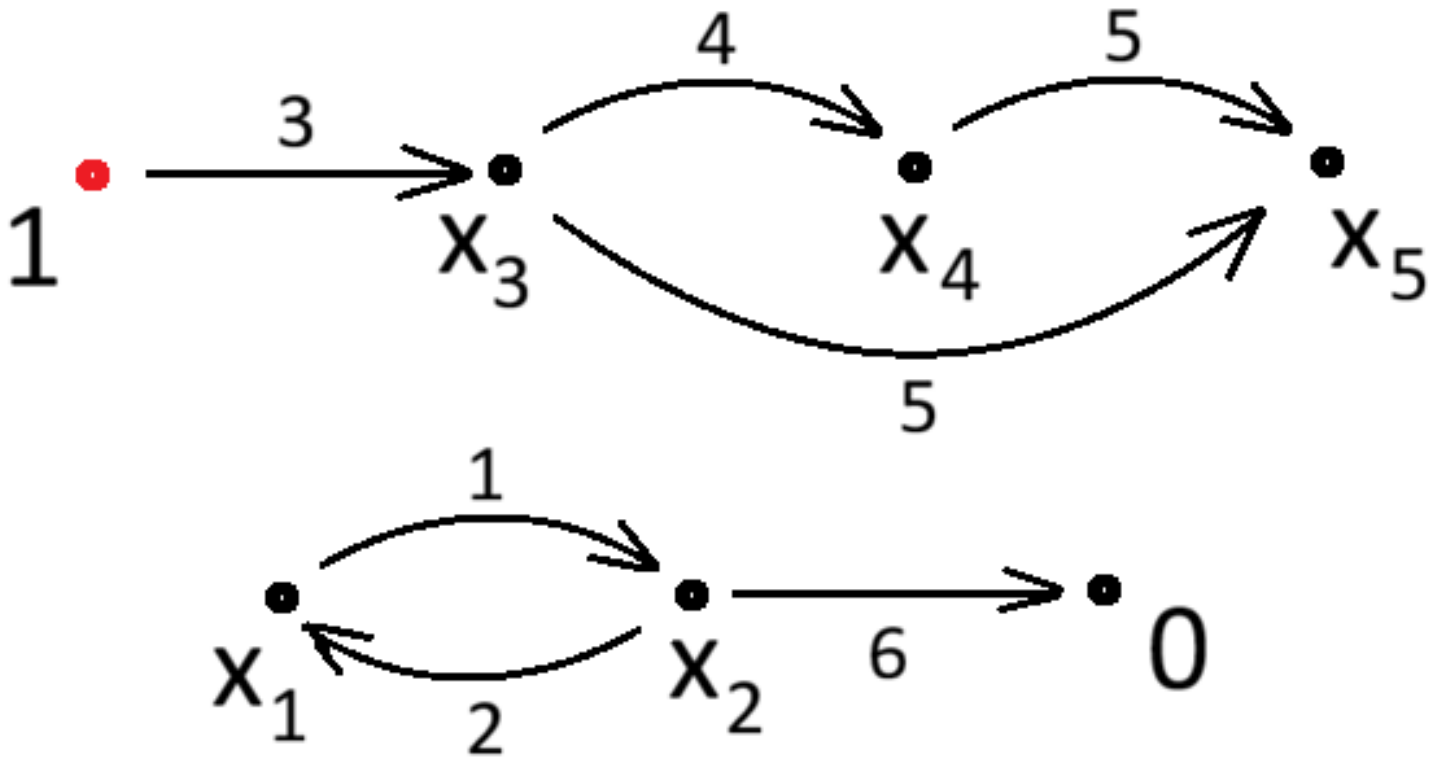
под "распространением" имеется ввиду многократное применение правила

Задача Horn-SAT

пример:

$$f(x) = (\neg x_1 \vee x_2) \wedge (x_1 \vee \neg x_2) \wedge x_3 \wedge (\neg x_3 \vee x_4) \wedge (\neg x_3 \vee \neg x_4 \vee x_5) \wedge \neg x_2$$

начальное состояние: только вершина 1

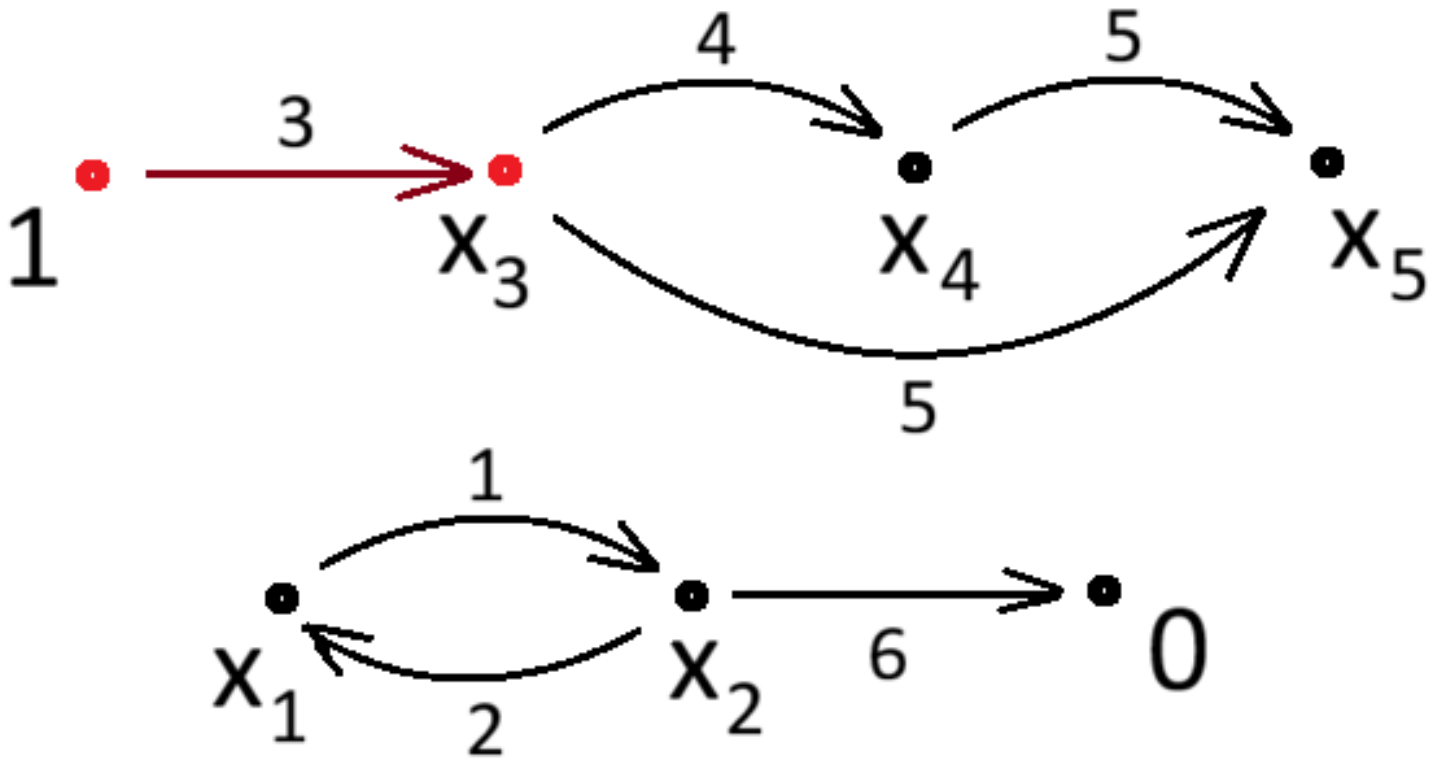


Задача Horn-SAT

пример:

$$f(x) = (\neg x_1 \vee x_2) \wedge (x_1 \vee \neg x_2) \wedge x_3 \wedge (\neg x_3 \vee x_4) \wedge (\neg x_3 \vee \neg x_4 \vee x_5) \wedge \neg x_2$$

скобка 3: $x_3 = 1$

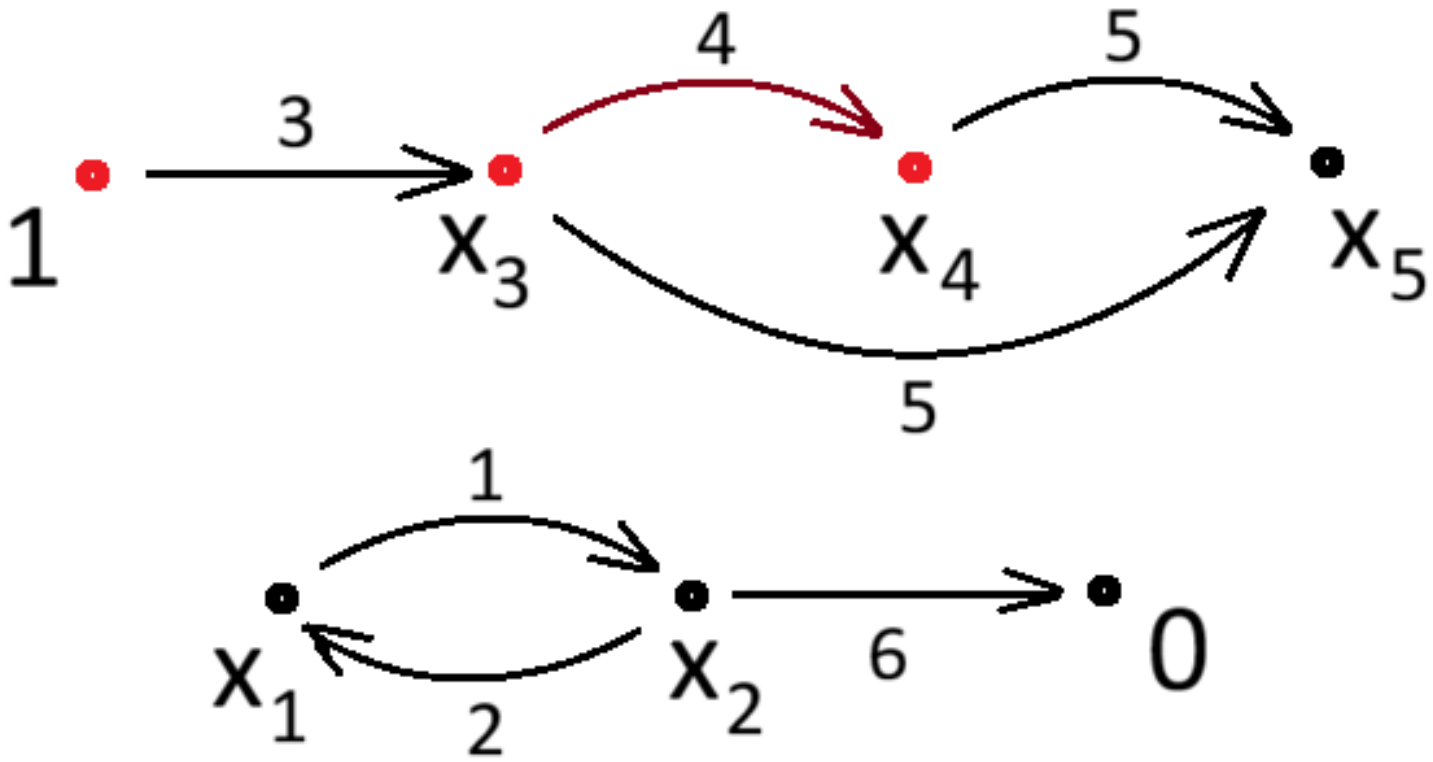


Задача Horn-SAT

пример:

$$f(x) = (\neg x_1 \vee x_2) \wedge (x_1 \vee \neg x_2) \wedge x_3 \wedge (\neg x_3 \vee x_4) \wedge (\neg x_3 \vee \neg x_4 \vee x_5) \wedge \neg x_2$$

скобка 4: $x_4 = 1$

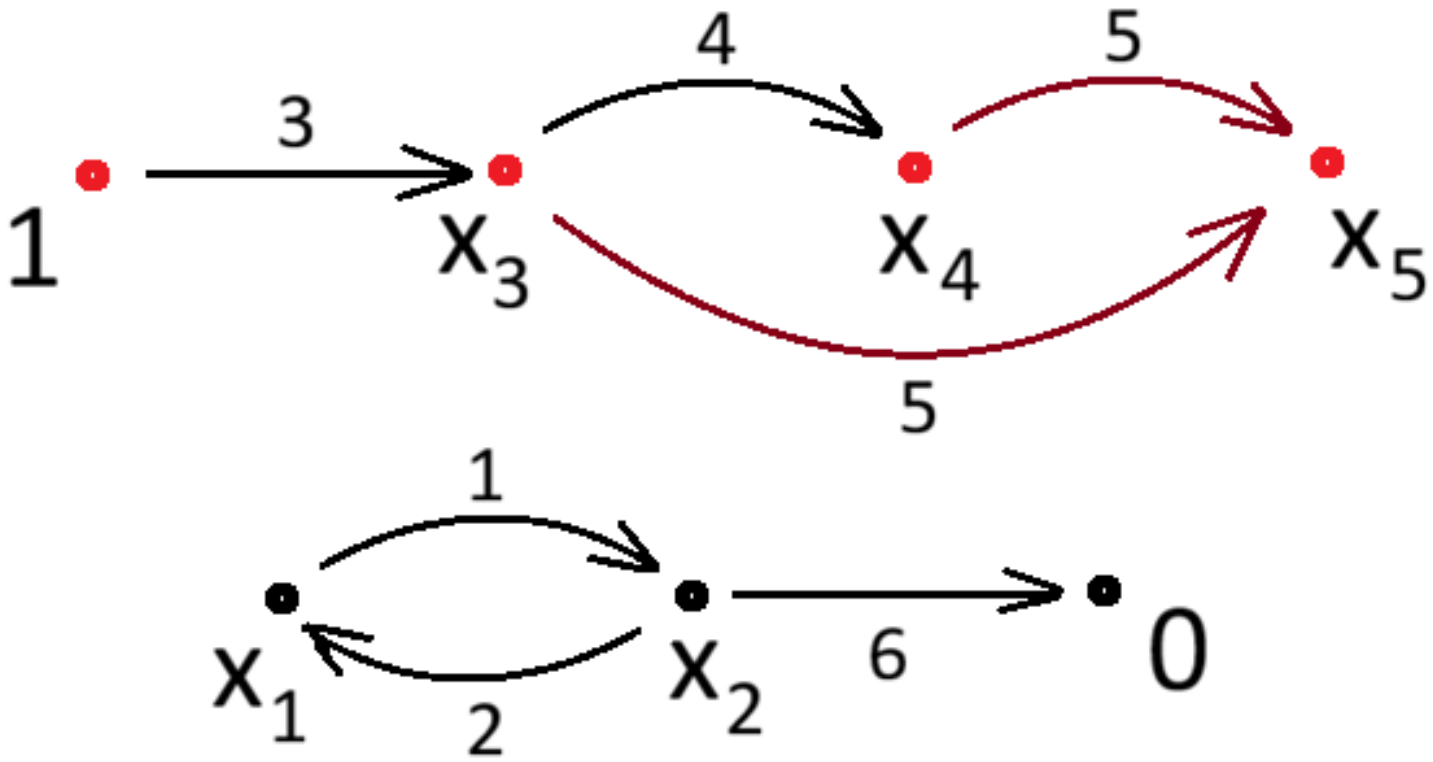


Задача Horn-SAT

пример:

$$f(x) = (\neg x_1 \vee x_2) \wedge (x_1 \vee \neg x_2) \wedge x_3 \wedge (\neg x_3 \vee x_4) \wedge (\neg x_3 \vee \neg x_4 \vee x_5) \wedge \neg x_2$$

скобка 5: $x_5 = 1$

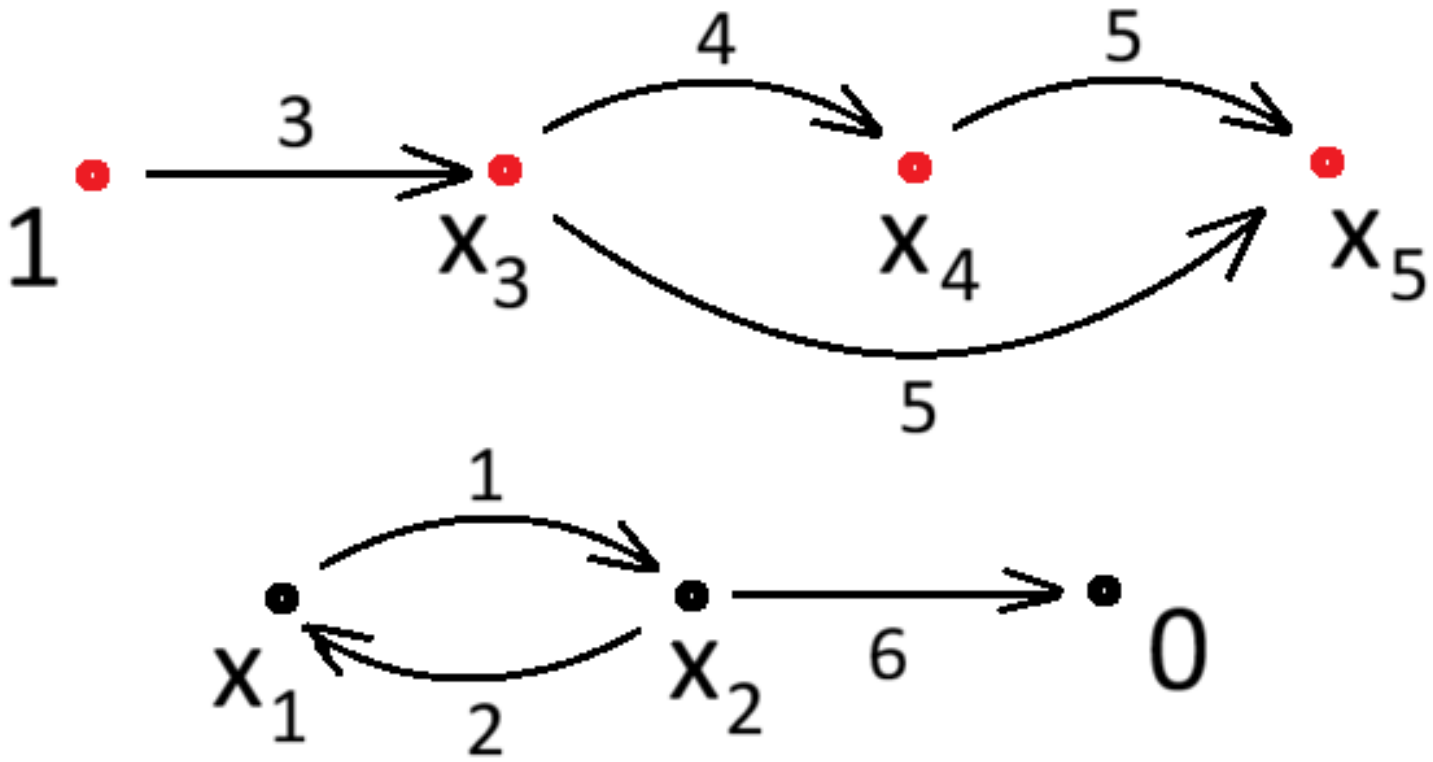


Задача Horn-SAT

пример:

$$f(x) = (\neg x_1 \vee x_2) \wedge (x_1 \vee \neg x_2) \wedge x_3 \wedge (\neg x_3 \vee x_4) \wedge (\neg x_3 \vee \neg x_4 \vee x_5) \wedge \neg x_2$$

конечное назначение: $x_3 = x_4 = x_5 = 1$



Задача Horn-SAT

есть алгоритм, производящий поиск *вширь* за *линейное* по числу выражений количество итераций

структуры представления графа

- список переменных с текущими назначениями
- для каждой переменной x_i, список скобок, содержащих $\neg x_i$
- список всех скобок
- для каждой скобки j, число ν_j выражений $\neg x_i$ в ней, которые ещё не приравнены 0
- для каждой скобки j, индекс $\xi_j = i$ выражения x_i, в неё входящего, если такое есть

Задача Horn-SAT

алгоритм строит и обрабатывает очередь из скобок j, для которых $\nu_j = 0$

- инициализация всеми скобками вида (x_i)
- если $\xi_j = \emptyset$ для обрабатываемой скобки, то формула недопустима
- если **новой** переменной $x_i = x_{\xi_j}$ назначается значение 1, число ν_l в скобках l, содержащих $\neg x_i$, уменьшается на 1
- скобки, для которых стало $\nu_l = 0$, встают в очередь

решение найдено если очередь опустеет

ключевой момент: не надо запоминать всё содержимое скобки, а только компоненту x_i и число ещё не определённых компонент $\neg x_i$

План

простые комбинаторные задачи

- 2-SAT
- Horn-SAT
- **кратчайший путь**
- максимальный поток
- наибольшее паросочетание
- хордальность / порядок совершенного исключения
- сепарирующая клика

Кратчайший путь

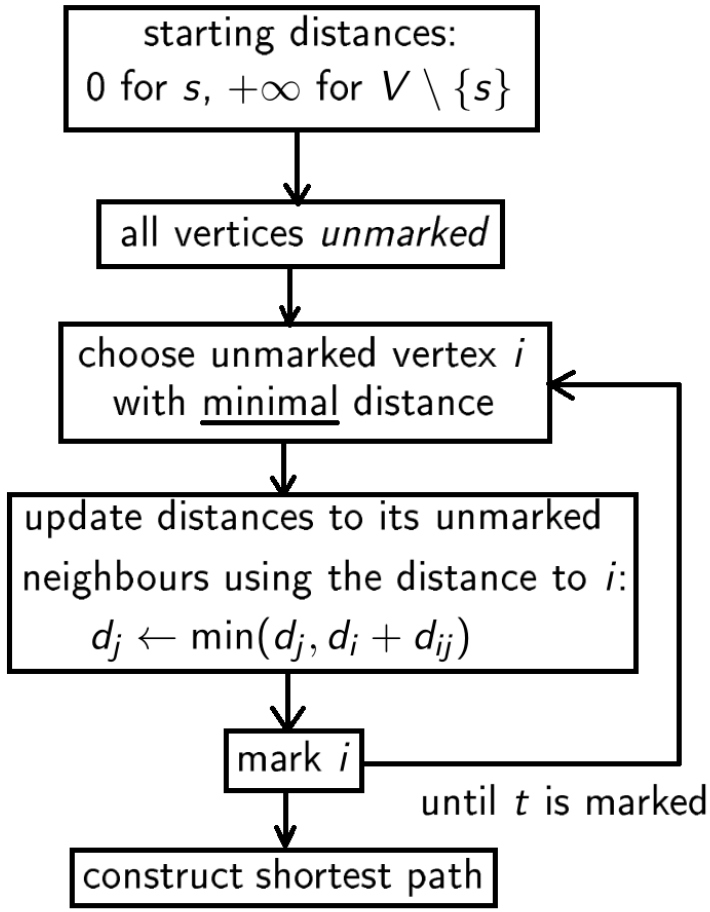
пусть $G = (V, E)$ — направленный граф с весами $w_e \geq 0$

пусть s, t — начальная и конечная вершина

нужно найти путь наименьшего веса, соединяющий s с t

алгоритм Дейкстры находит решение за $O(|V|^2)$ операций (улучшается до $O(|E| + |V| \log |V|)$)

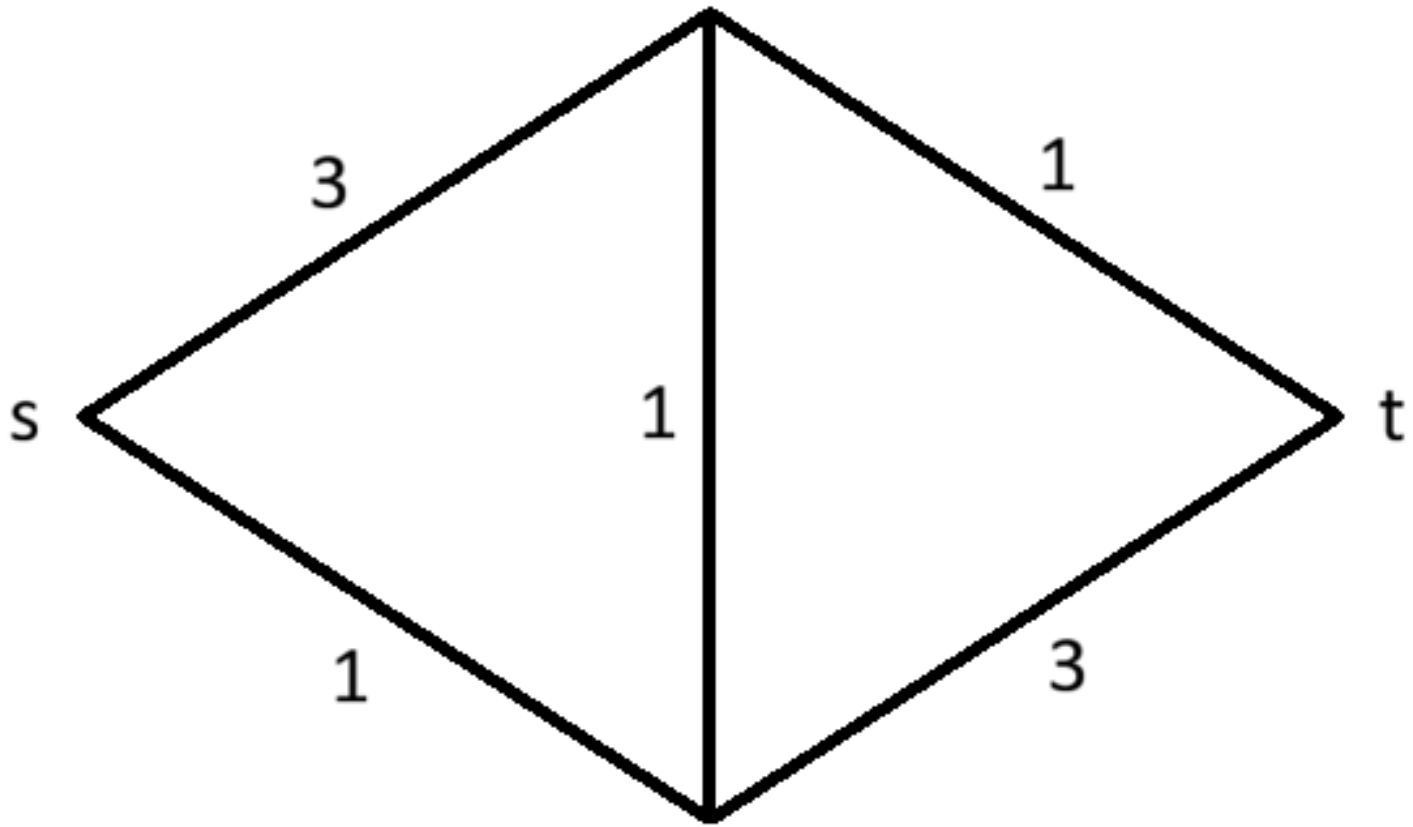
кратчайший путь строится в обратном порядке



если $d_j = d_i + d_{ij}$, то i может предшествовать j в кратчайшем пути

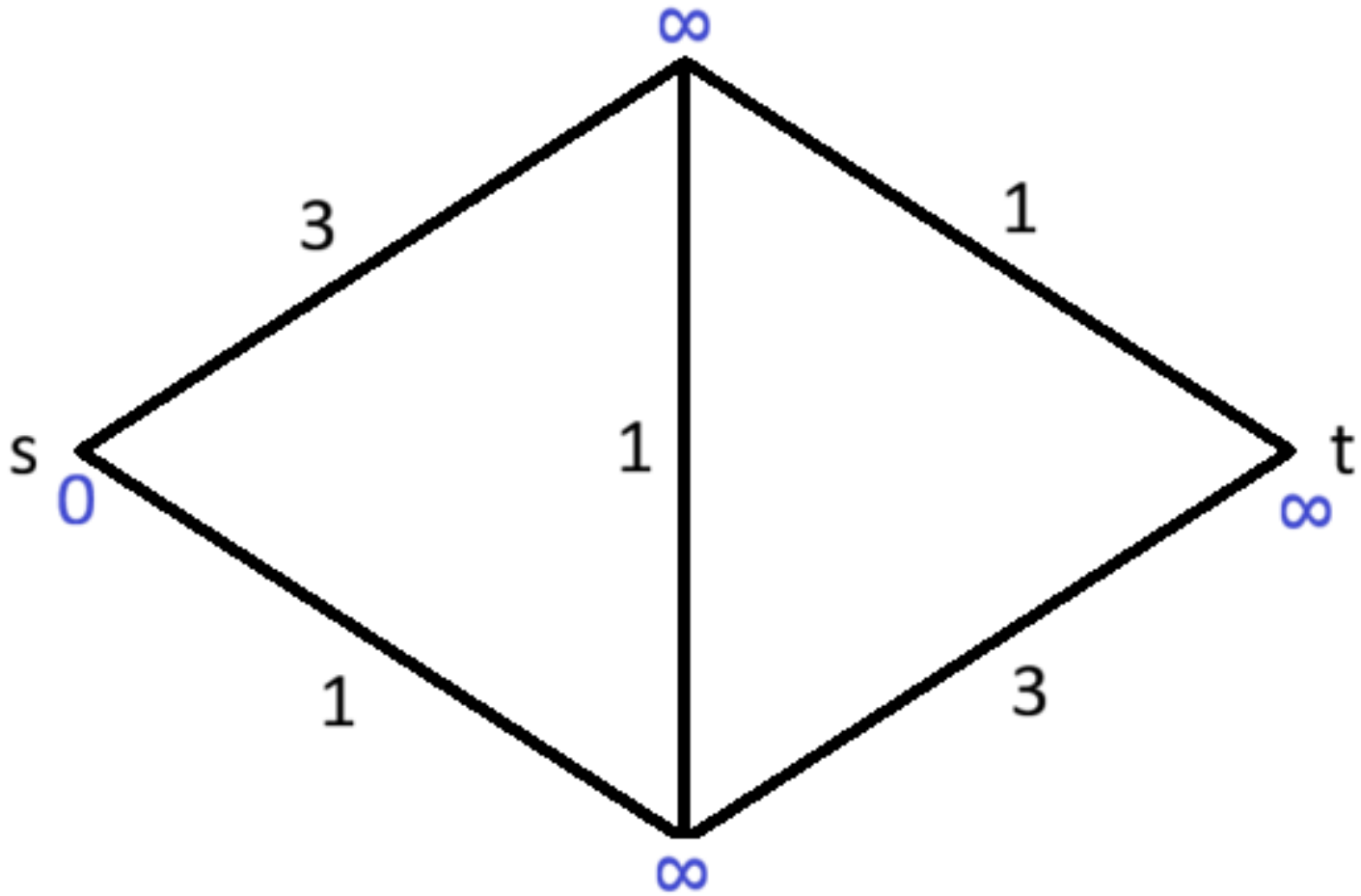
Кратчайший путь

пример:



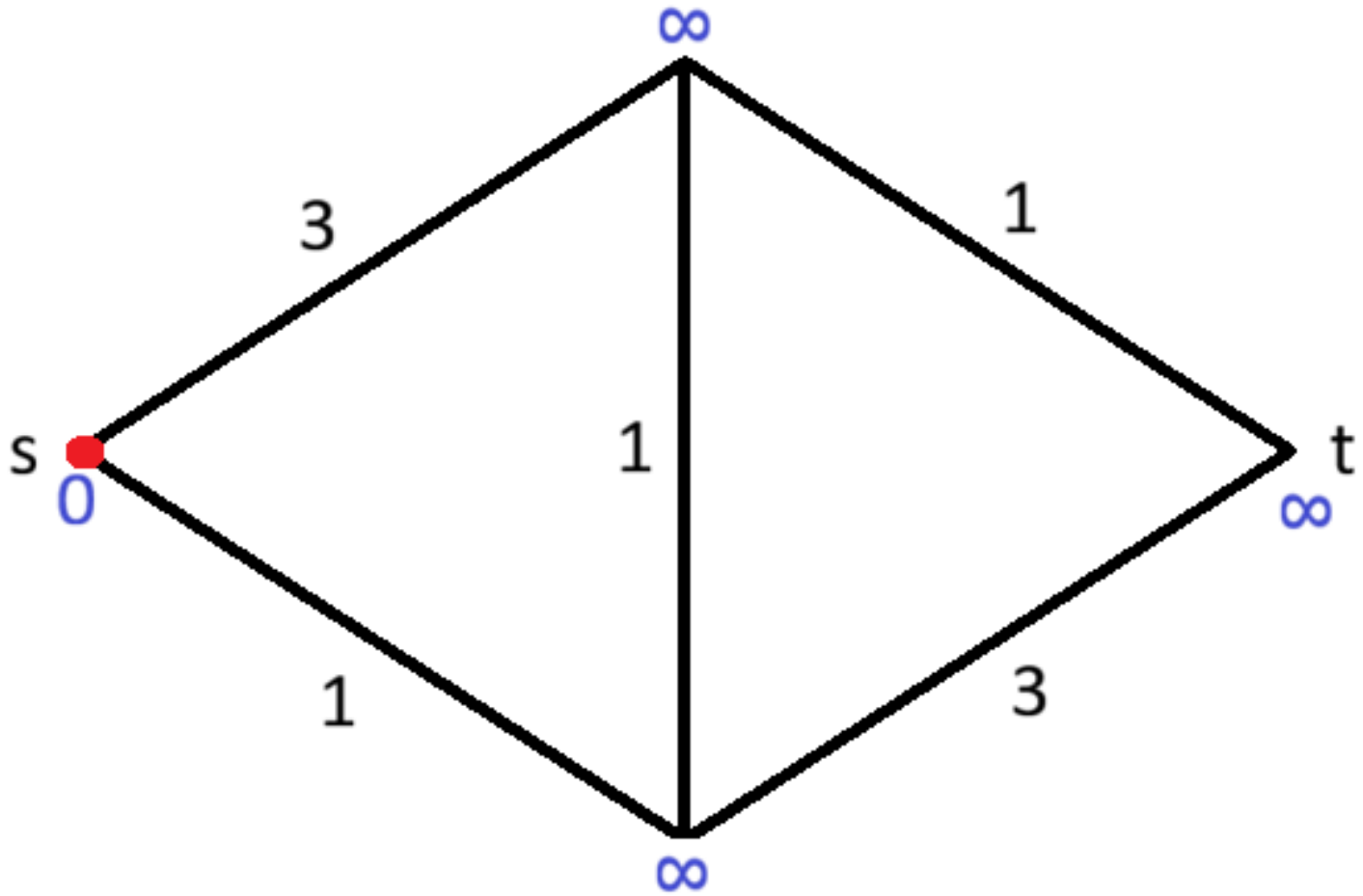
Кратчайший путь

инициализация расстояний



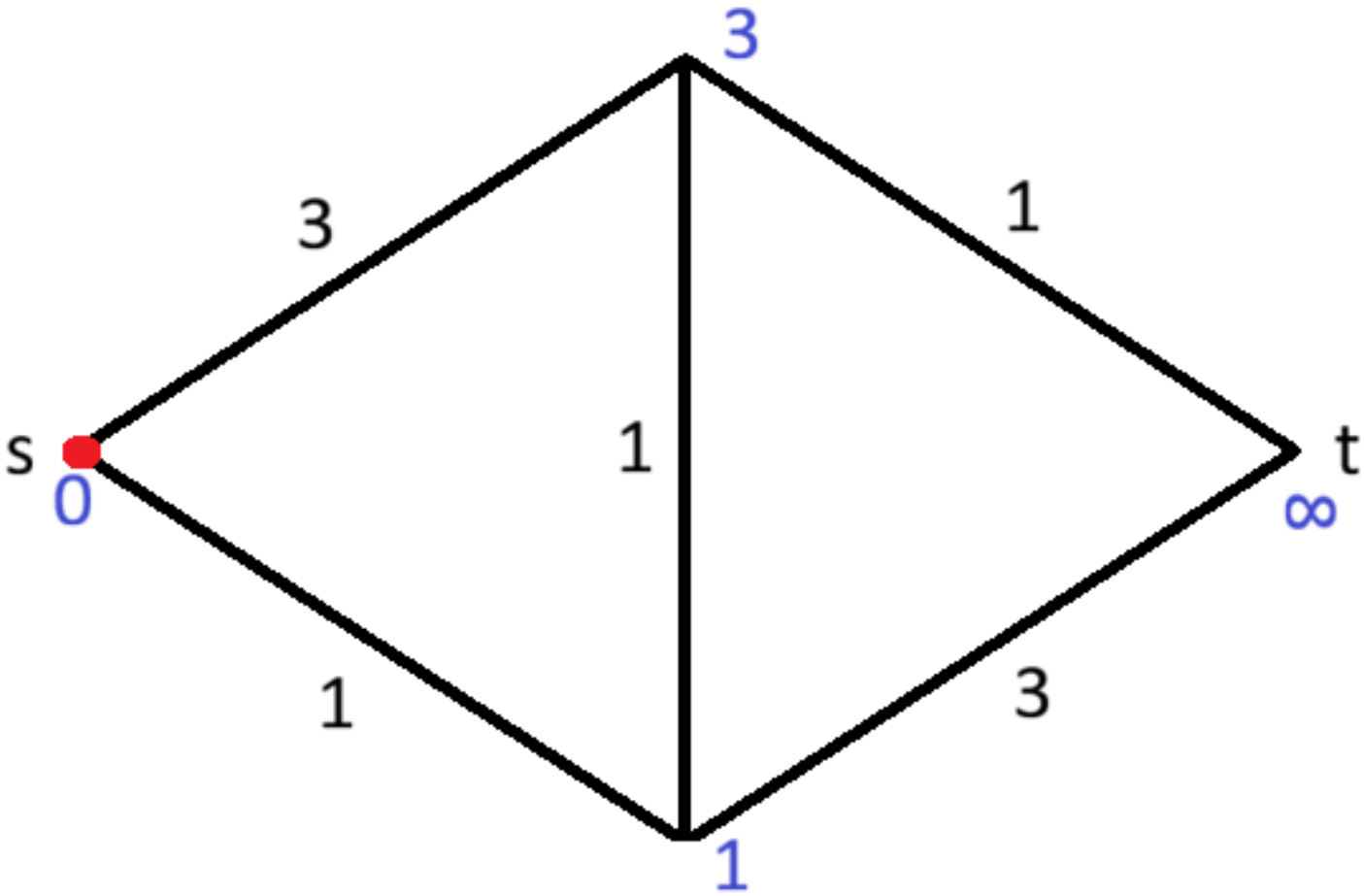
Кратчайший путь

выбор непомеченной вершины с минимальным расстоянием



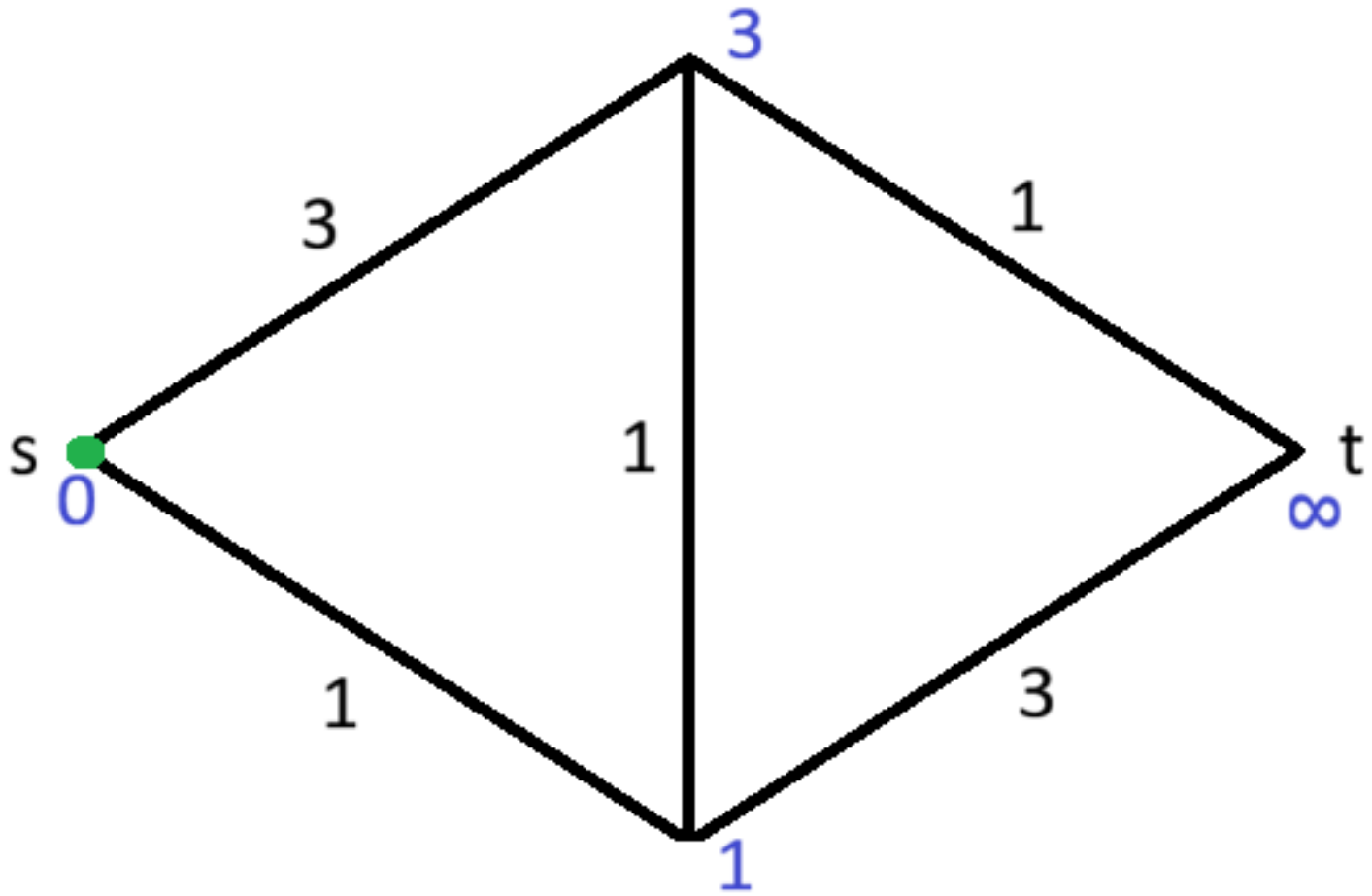
Кратчайший путь

обновление расстояний до непомеченных соседей



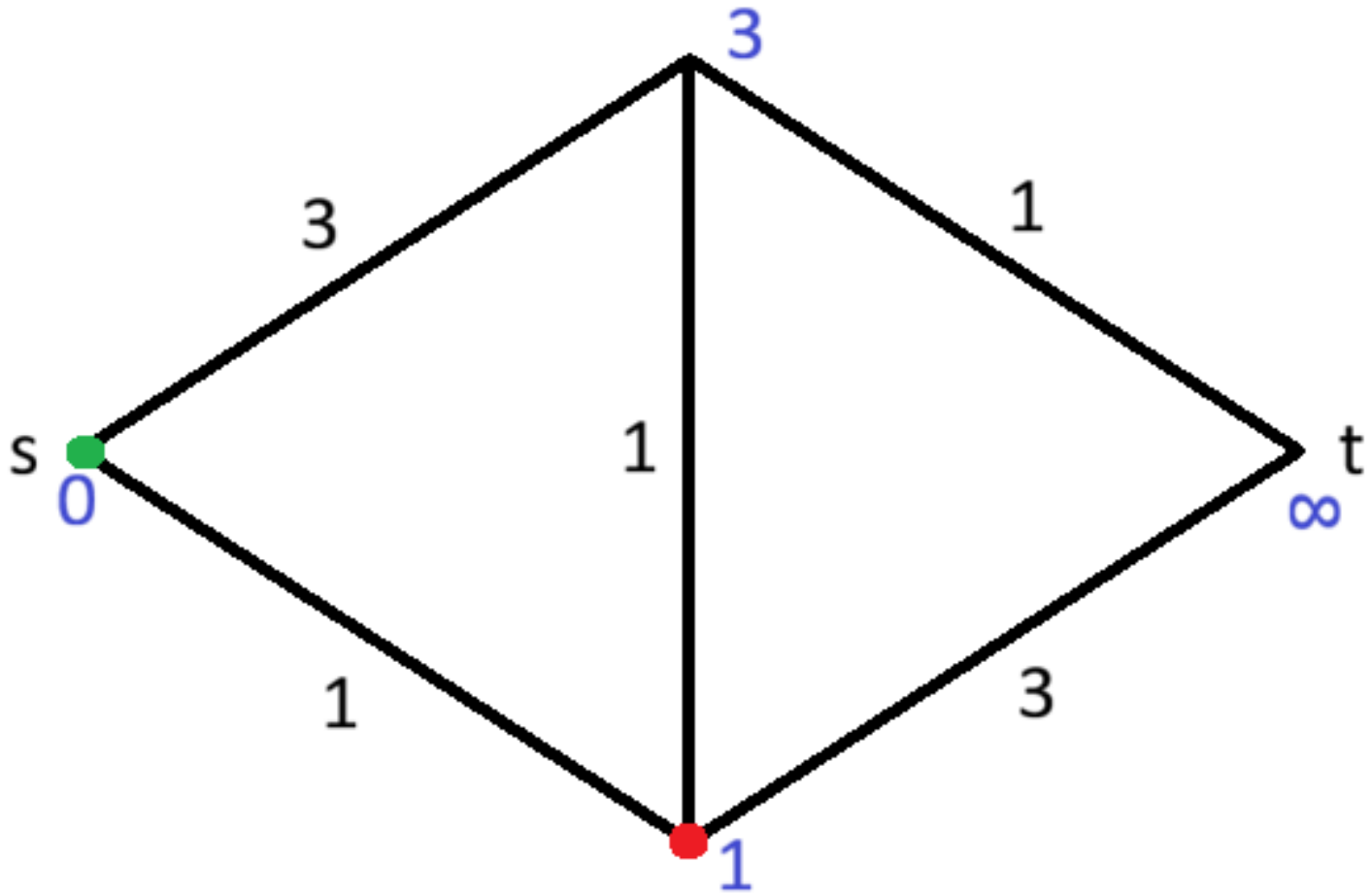
Кратчайший путь

пометка обработанной вершины



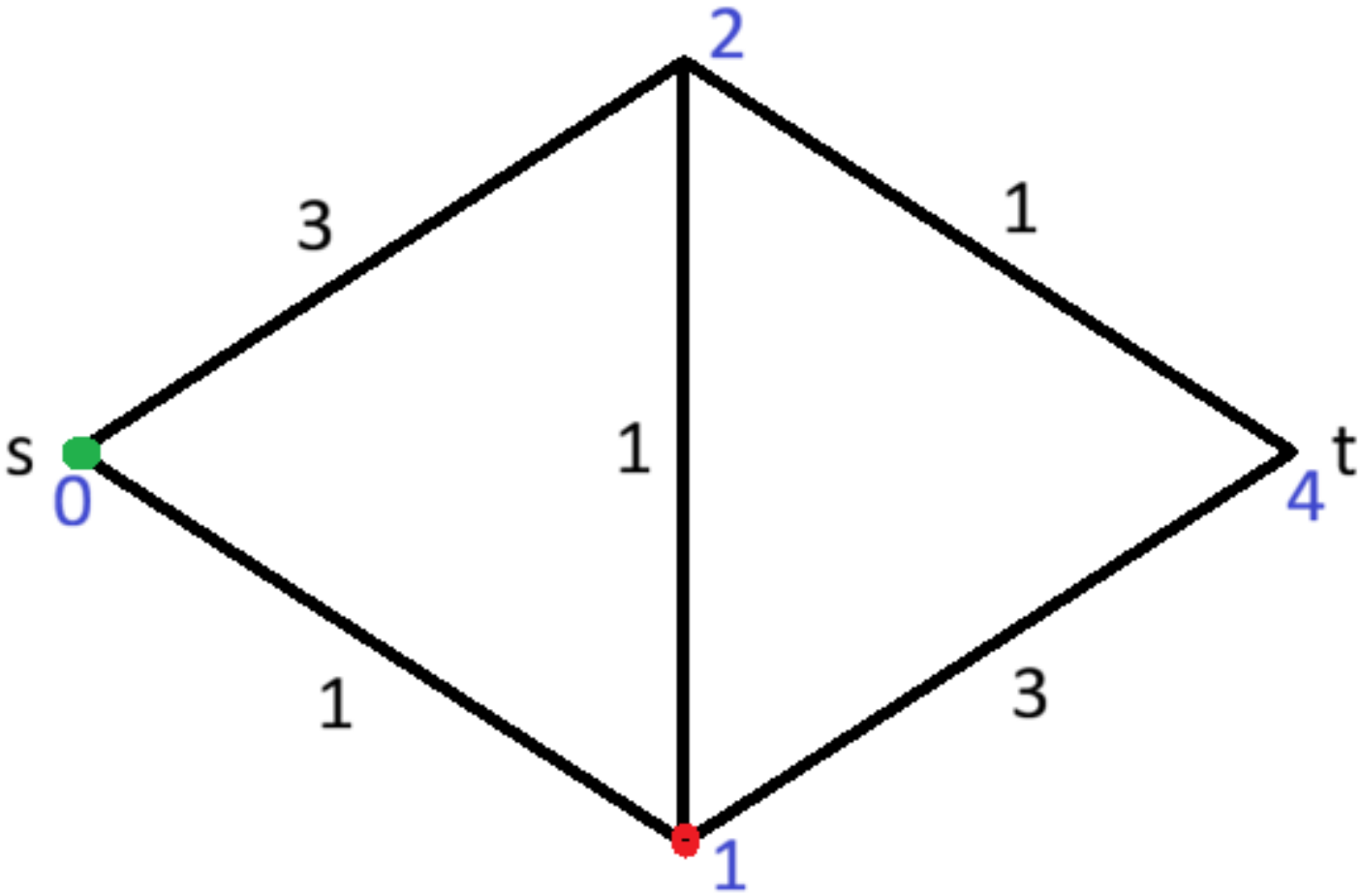
Кратчайший путь

выбор непомеченной вершины с минимальным расстоянием



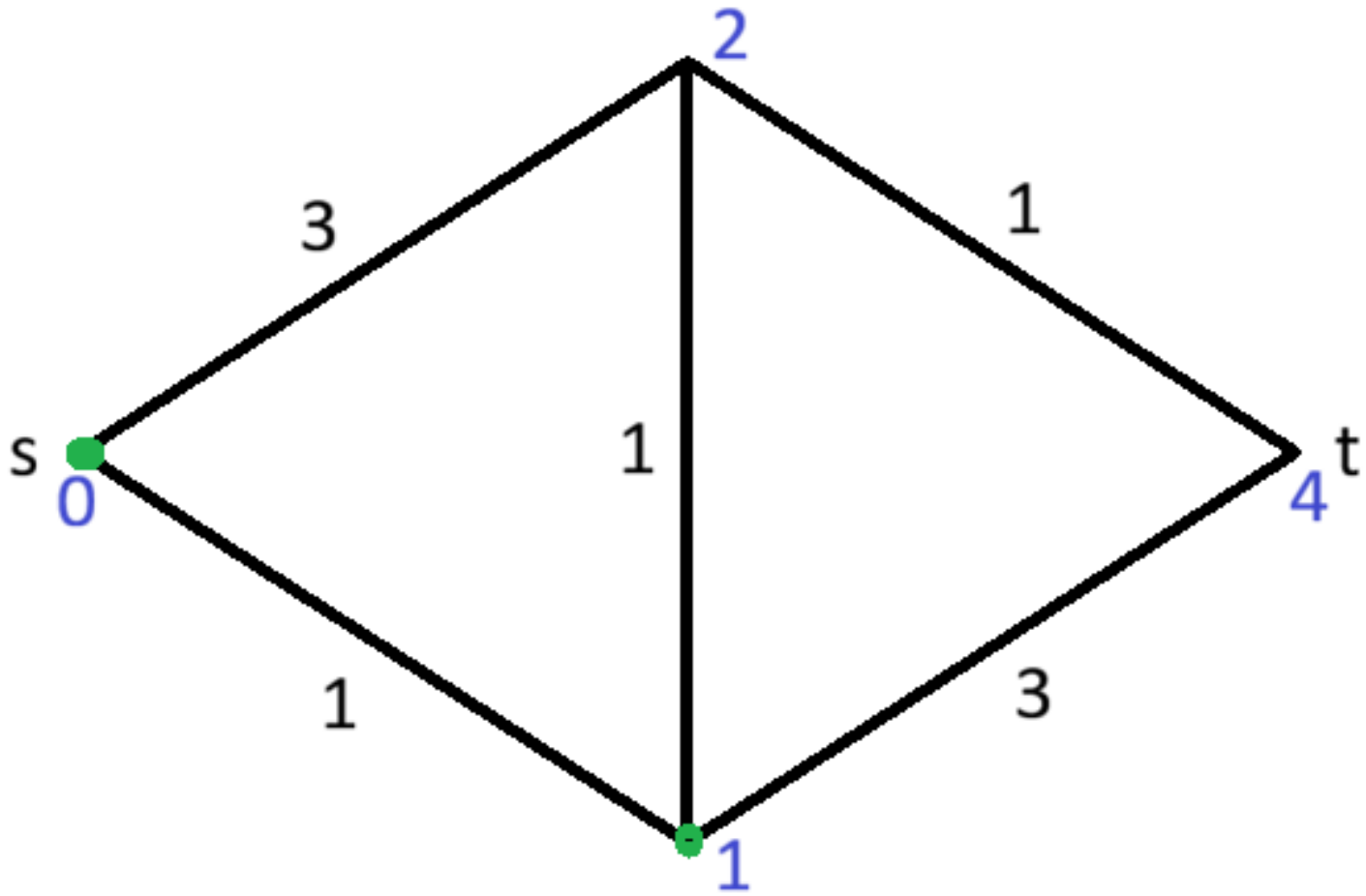
Кратчайший путь

обновление расстояний до непомеченных соседей



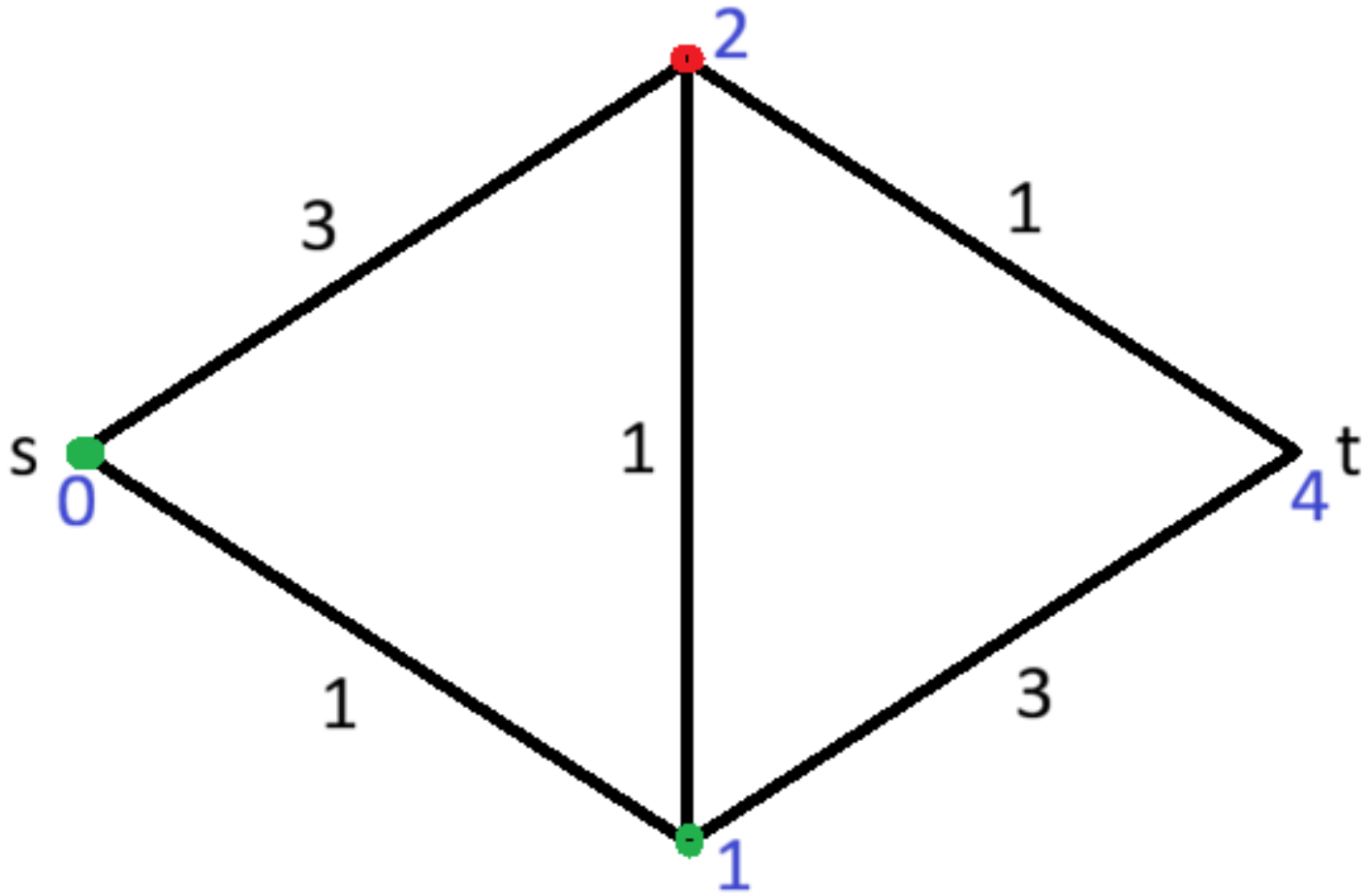
Кратчайший путь

пометка обработанной вершины



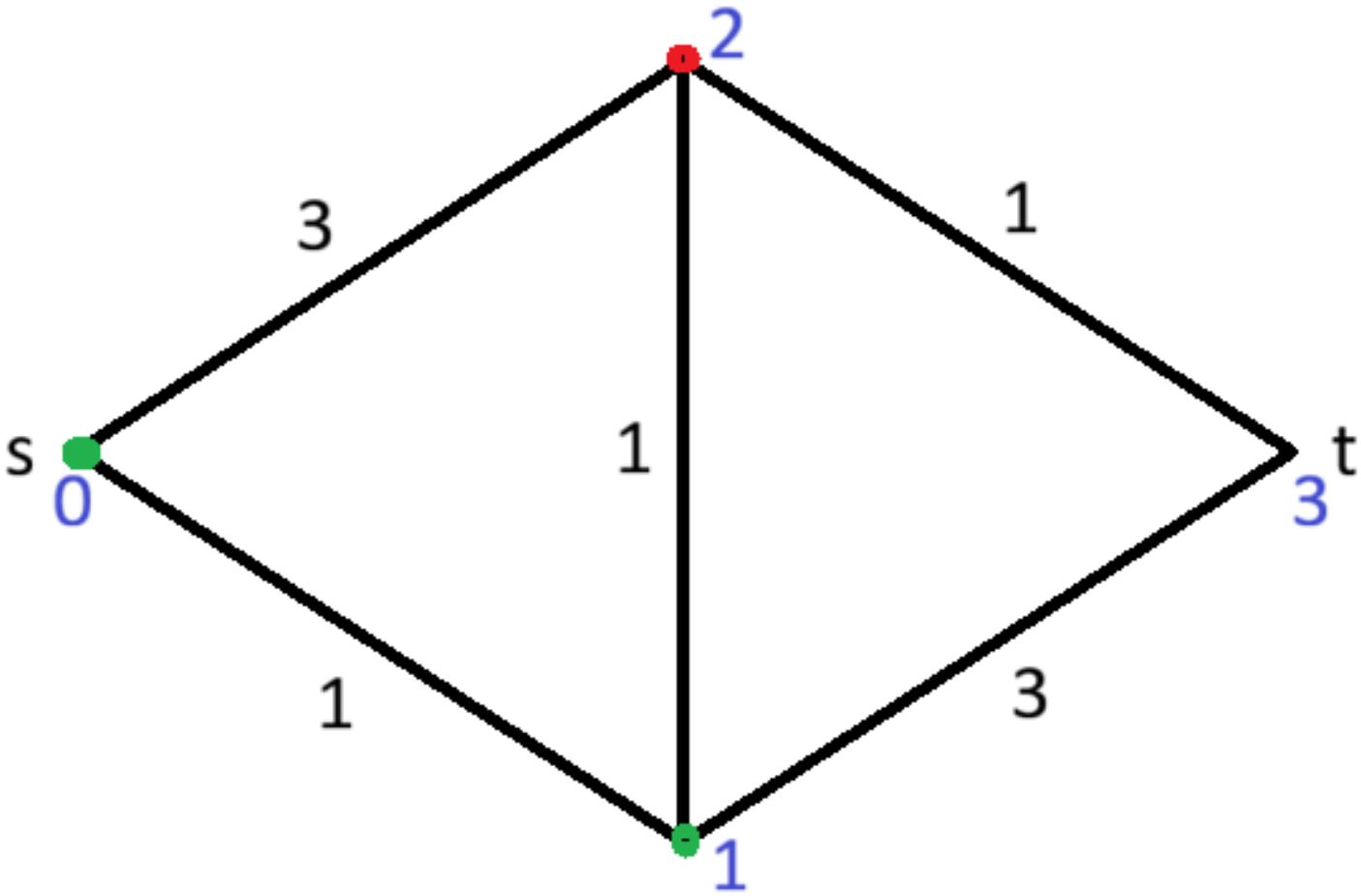
Кратчайший путь

выбор непомеченной вершины с минимальным расстоянием



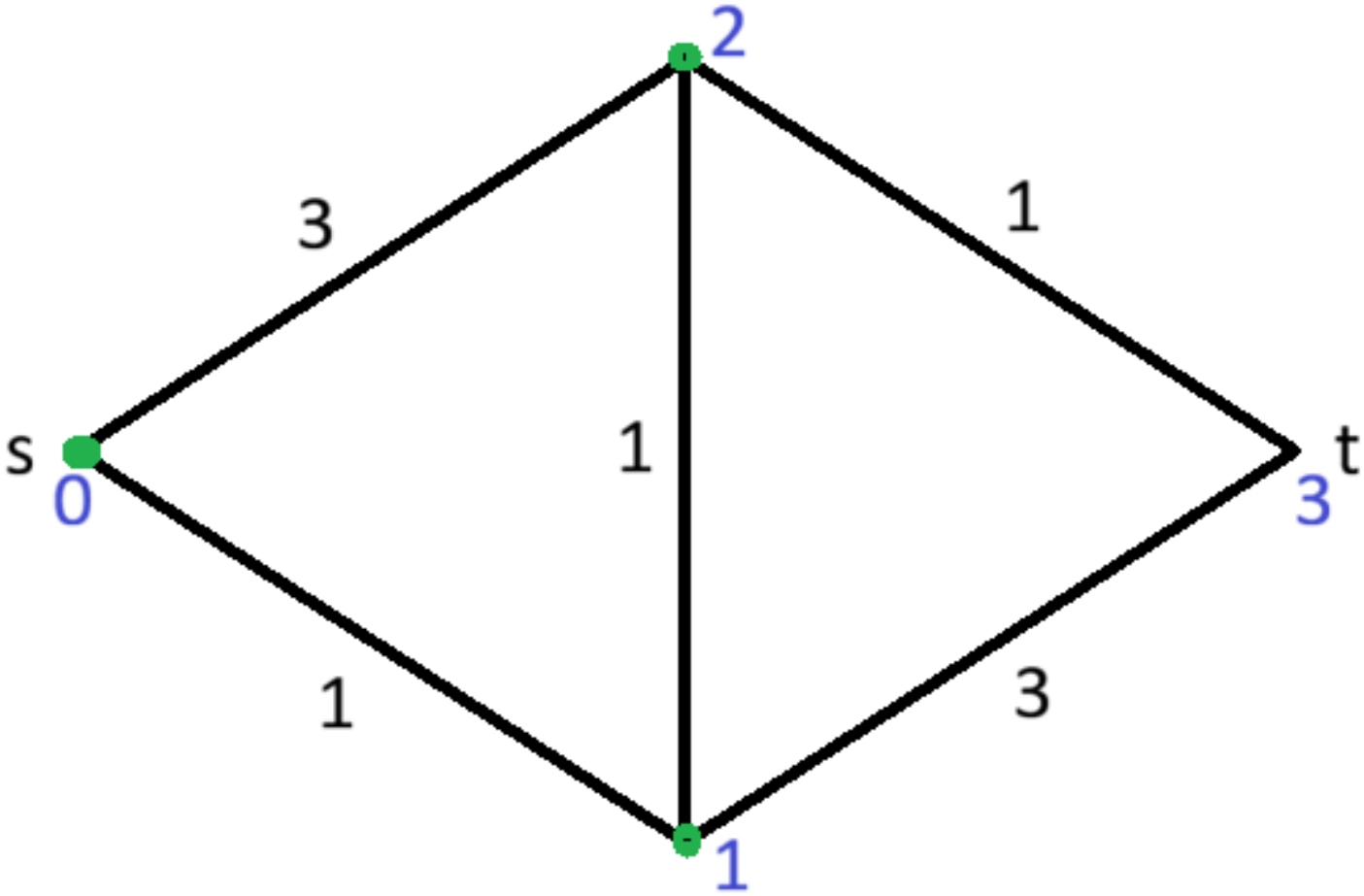
Кратчайший путь

обновление расстояний до непомеченных соседей



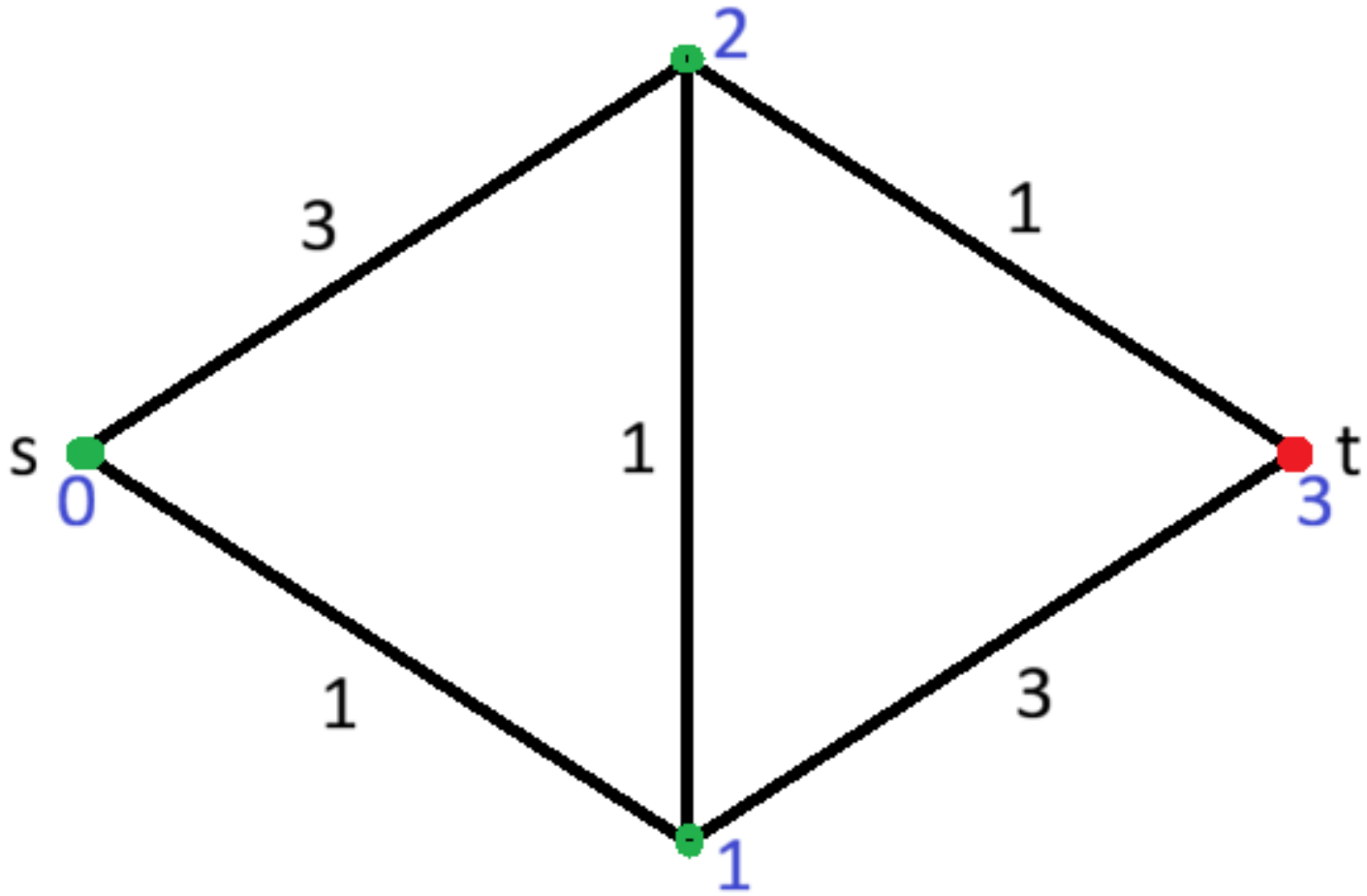
Кратчайший путь

пометка обработанной вершины



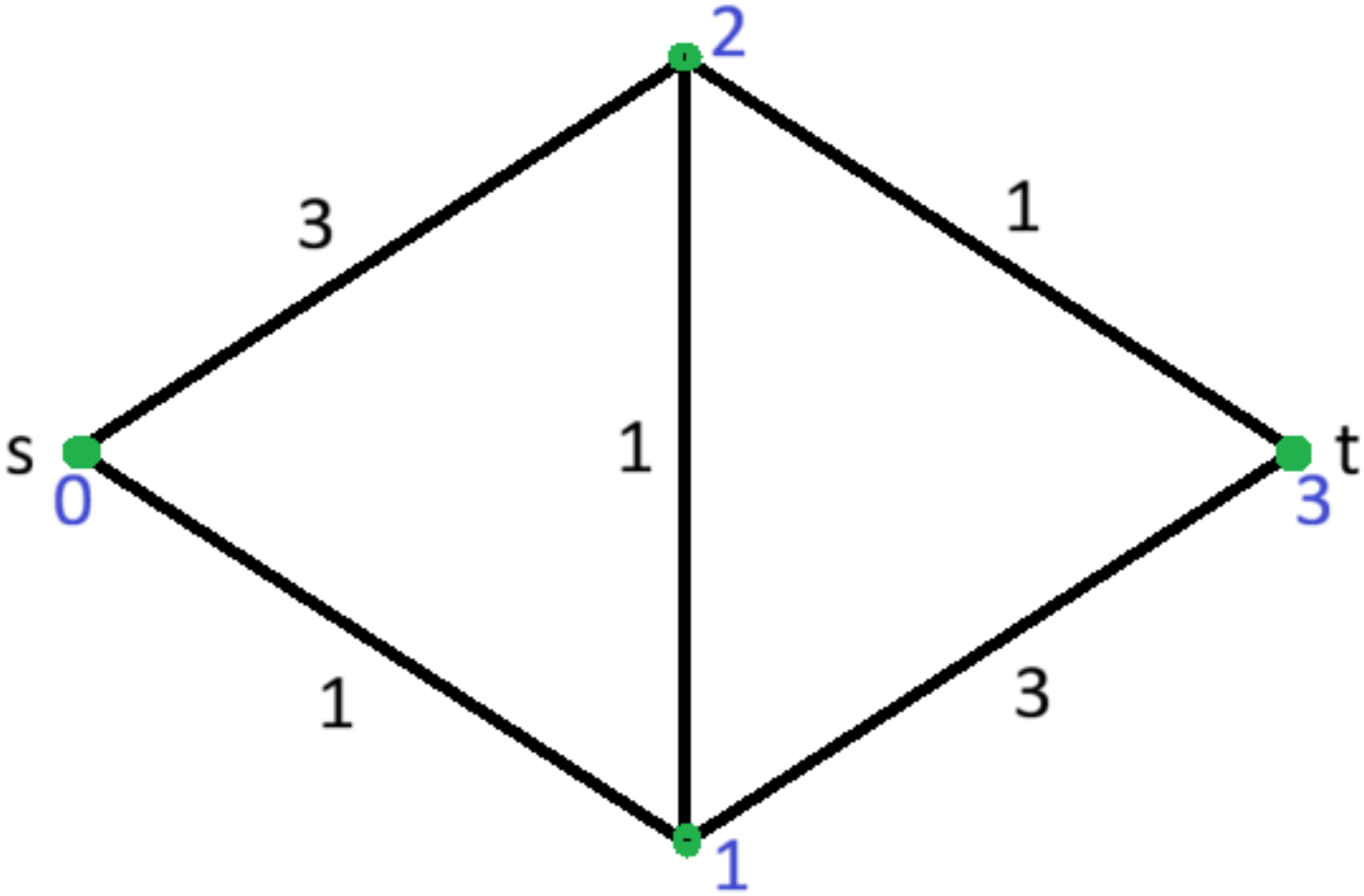
Кратчайший путь

выбор непомеченной вершины с минимальным расстоянием



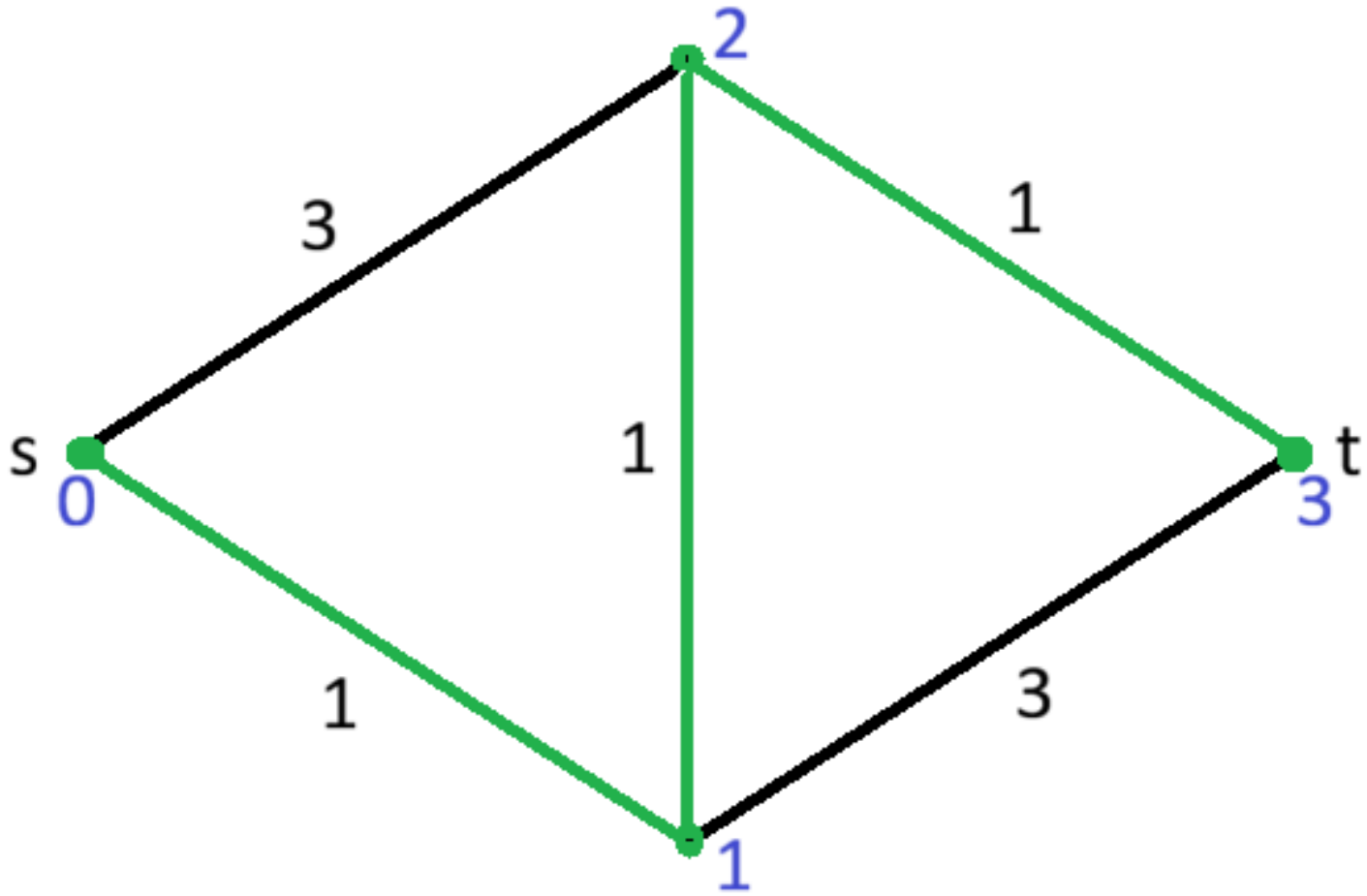
Кратчайший путь

пометка обработанной вершины



Кратчайший путь

построение кратчайшего пути



Кратчайший путь

если нужно найти кратчайший путь для *всех* пар вершин:

алгоритм Флойда-Варшала обходится $O(|V|^3)$ операциями

толерирует отрицательные веса если нет цикла с суммой < 0

- построить начальную матрицу расстояний $d_{ij} = w_{(i,j)}$ ($d_{ij} = +\infty$ если нет ребра (i, j), $d_{ii} = 0$)
- for $i, j, k = 1 : |V|$ do $d_{jk} \leftarrow \min(d_{jk}, d_{ji} + d_{ik})$

после завершения внешнего цикла для $i = l$, d_{jk} содержит кратчайшее расстояние между j и k если только вершины $1, \ldots, l$ разрешены как промежуточные

идея доказательства:

- улучшение расстояния для $i = l$ возможно только если путь через l короче, чем самый короткий доселе найденный
- путь может только 1 раз пройти через l
- оптимальные куски от и до l уже посчитаны

План

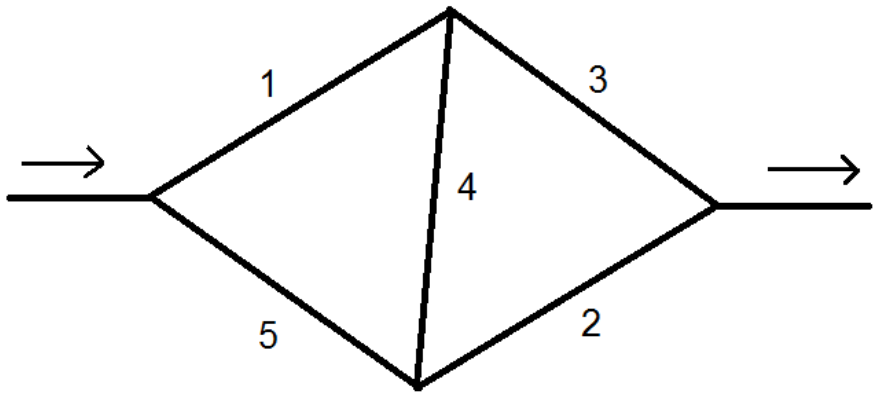
простые комбинаторные задачи

- 2-SAT
- Horn-SAT
- кратчайший путь
- **максимальный поток**
- наибольшее паросочетание
- хордальность / порядок совершенного исключения
- сепарирующая клика

Максимальный поток

пусть $G = (V, E)$ — направленный граф с пропускными способностями рёбер $c_e \geq 0$

пусть s, t — начальная и целевая вершины
нужно найти максимальный поток из s в t



методы решения

- линейная программа
- алгоритм Эдмондса-Карпа со сложностью $O(|V| \cdot |E|^2)$
- Orlin или King-Rao-Tarjan: сложность $O(|V| \cdot |E|)$

Максимальный поток

обозначим вершины $V = \{1, \ldots, n\}$, $s = 1$, $t = n$

формулировка ЛП

$$\max_{F=-F^T} \sum_{j=1}^{n} F_{1j} : \quad \sum_{j=1}^{n} F_{ij} = 0,\ i = 2, \ldots, n-1;\ F_{ij} \le c_{(i,j)}$$

- F_{ij} — направленный поток из вершины i в j
- $\sum_{j=1}^{n} F_{1j}$ — выходящий из s поток
- равенства — уравнения баланса в промежуточных вершинах

решать ЛП — не самый эффективный способ:
$|E|$ переменных, $|V|$ уравнений, сложность решения $\sim |E|^{3.5}$

Максимальный поток

методы типа Форда-Фалкерсона

- инициализируем текущий поток нулевым
- находим путь от s до t, состоящий из *ненасыщенных* рёбер
- добавляем максимальный поток по пути к текущему потоку
- алгоритм завершает работу, если путь не находится

свойства алгоритма

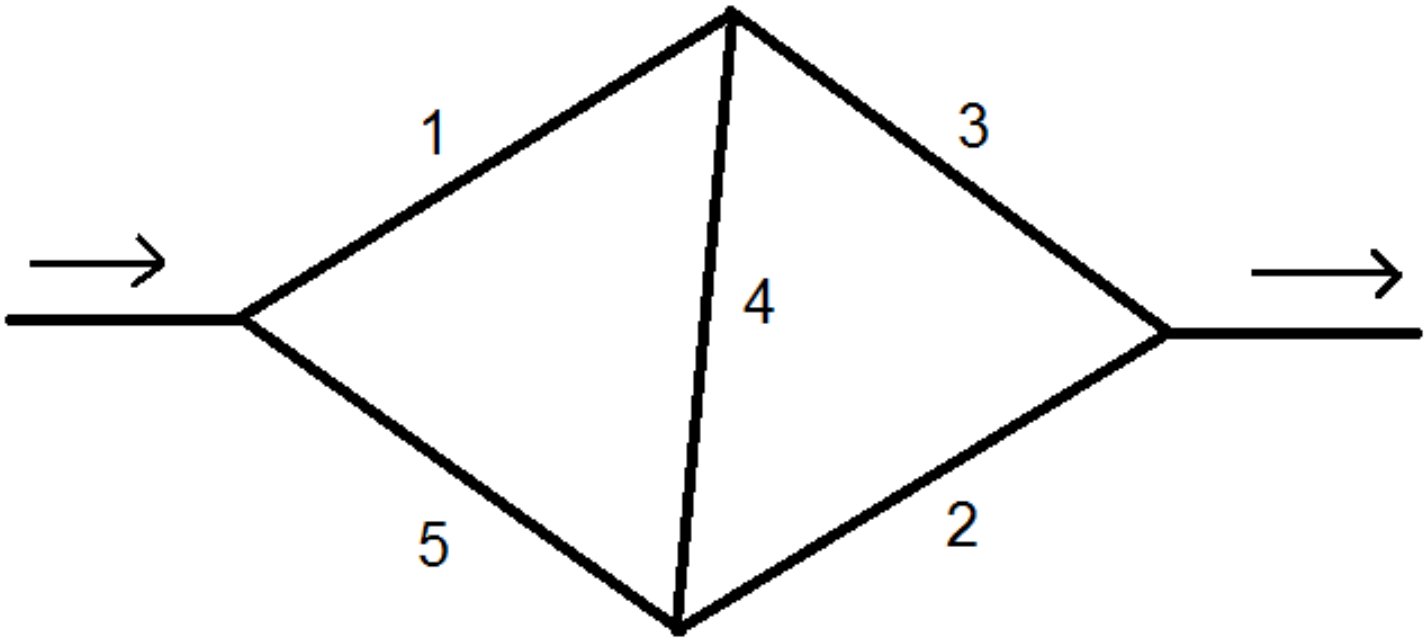
- текущий поток монотонно растёт
- алгоритм останавливается, когда находится *минимальный разрез*, отделяющий s от t
- есть несходящиеся версии алгоритма, даже в пределе (в зависимости от способа поиска пути)

алгоритм Эдмондса-Карпа [Диниц, 1970] — версия, использующая поиск вширь для нахождения кратчайшего нового пути

- длина кратчайшего пути монотонно растёт
- алгоритм терминирует за конечное число шагов

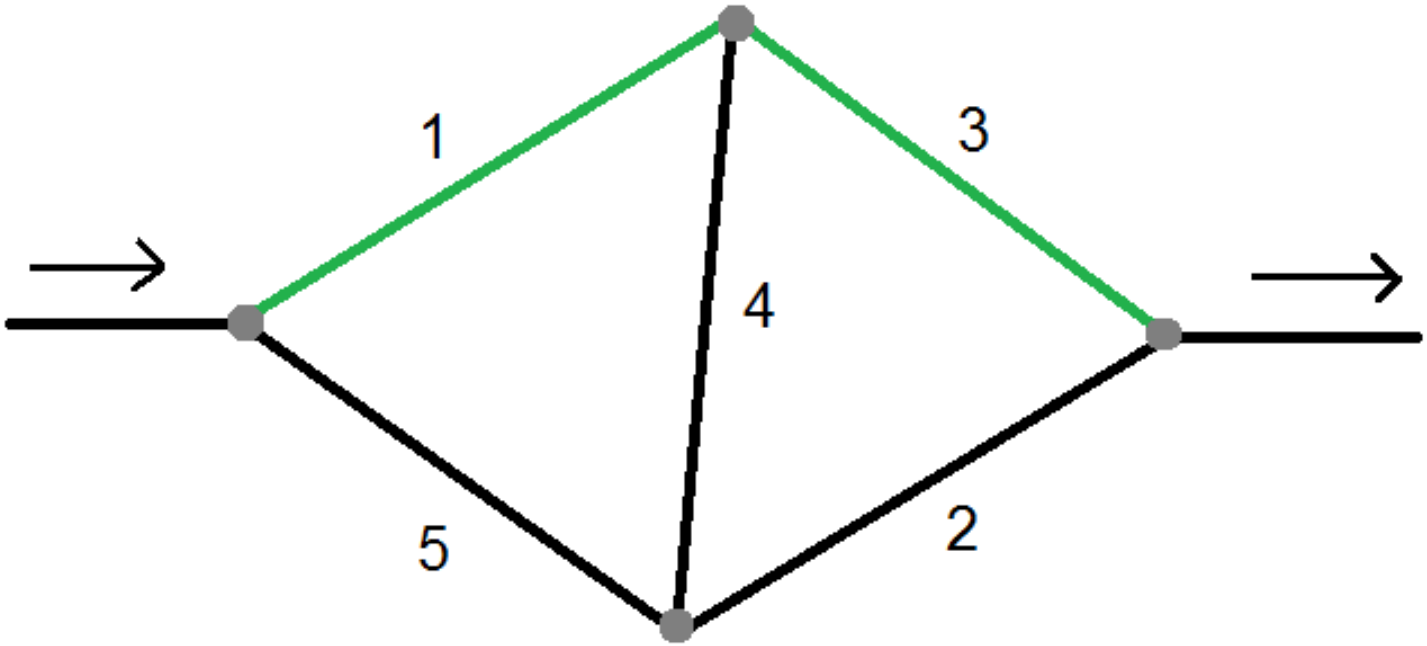
Максимальный поток

пример



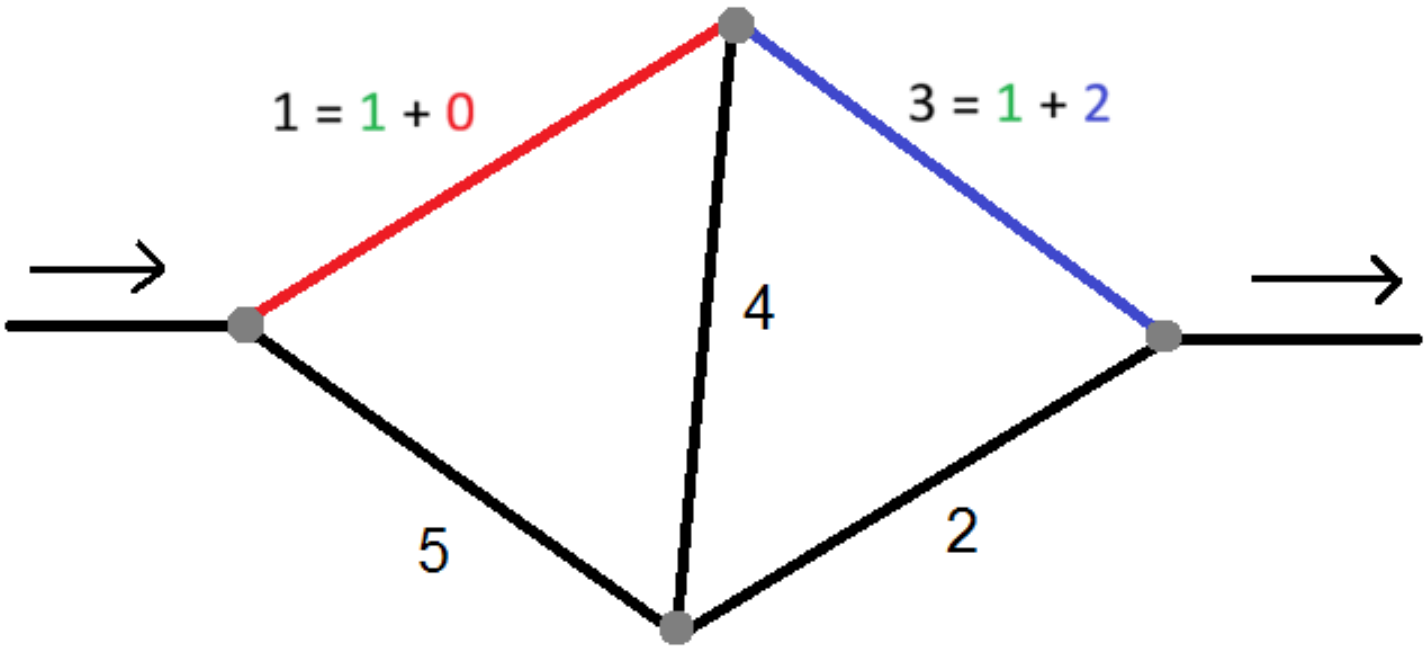
Максимальный поток

найден кратчайший путь



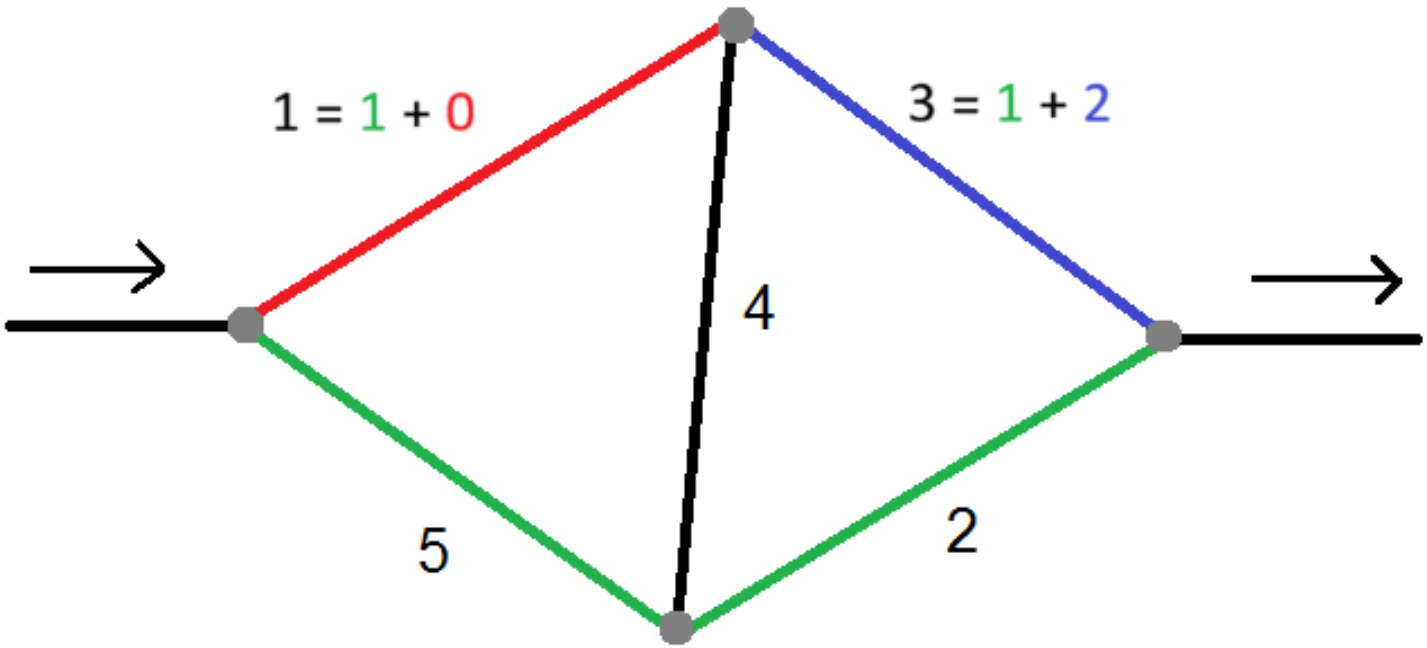
Максимальный поток

поток насыщает ребро



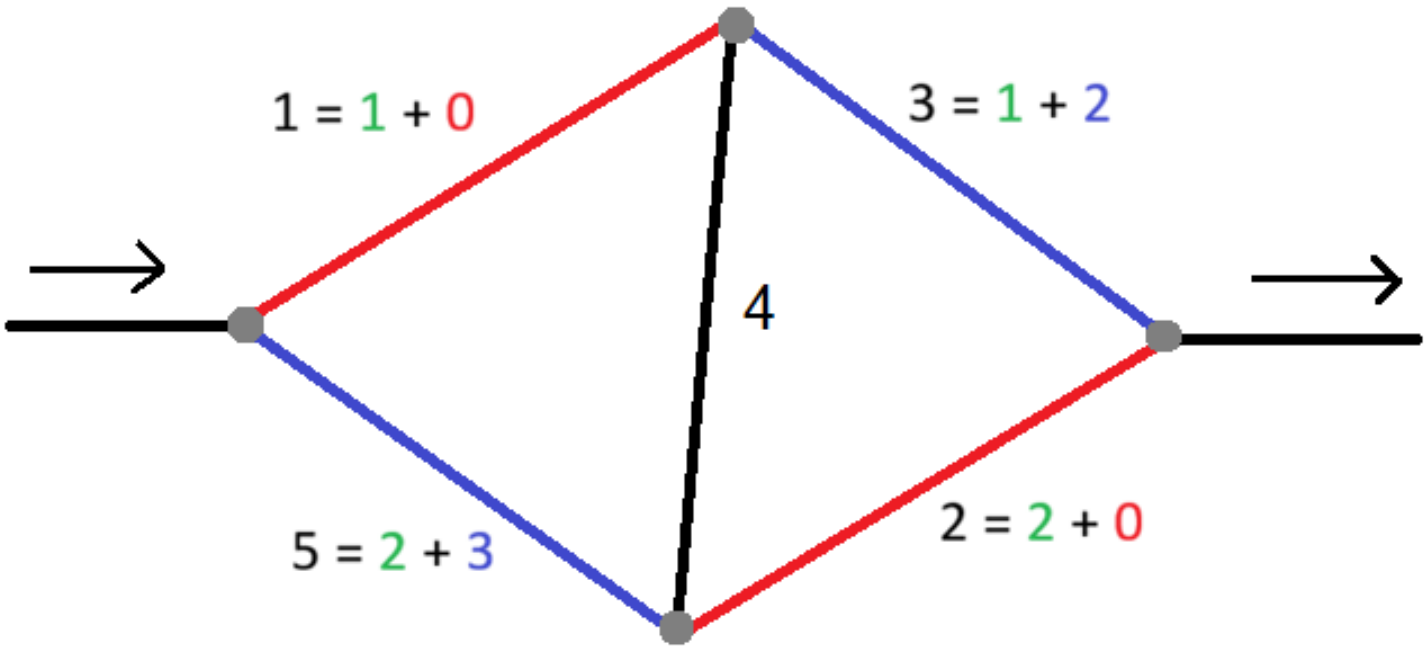
Максимальный поток

найден кратчайший путь



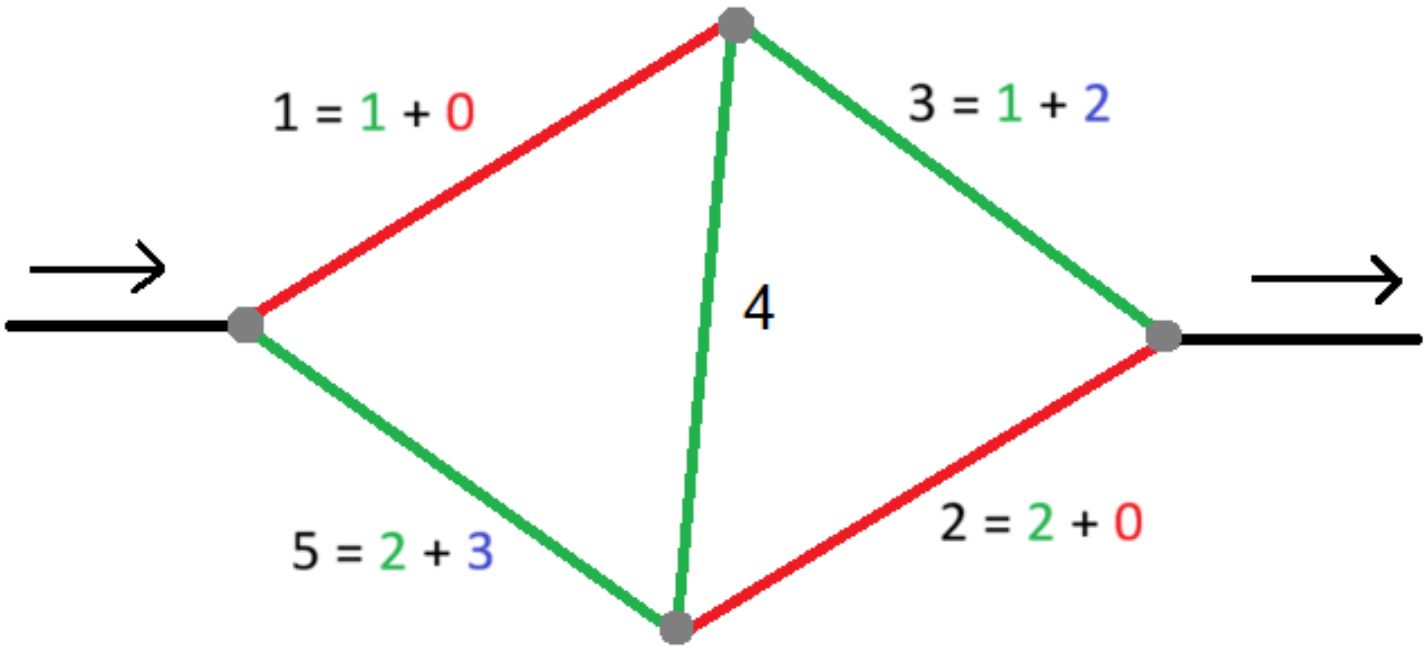
Максимальный поток

поток насыщает ещё одно ребро



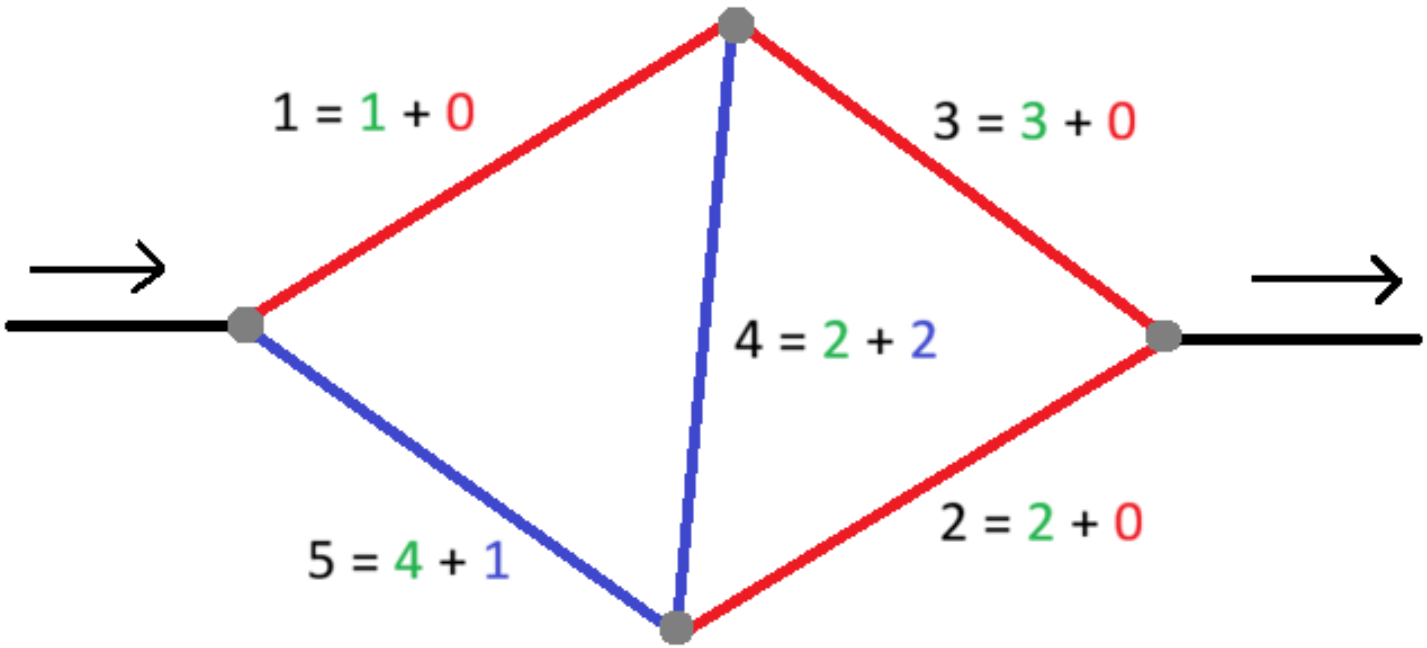
Максимальный поток

найден кратчайший путь



Максимальный поток

поток насыщает ещё одно ребро, остаточный граф несвязный



План

простые комбинаторные задачи

- 2-SAT
- Horn-SAT
- кратчайший путь
- максимальный поток
- **наибольшее паросочетание**
- хордальность / порядок совершенного исключения
- сепарирующая клика

Наибольшее паросочетание

рассмотрим ненаправленный граф $G = (V, E)$

Определение

Паросочетанием в G называют подмножество $M \subset E$ попарно несмежных рёбер.
Наибольшим паросочетанием называют паросочетание максимальной мощности.

задача *о наибольшем паросочетании* состоит в построении такого

максимальное паросочетание — это паросочетание, к которому невозможно добавить ребро

максимальное — не обязательно наибольшее

Наибольшее паросочетание

если для каждой вершины $v \in V$ существует ребро $(v, v') \in M$, то паросочетание M — наибольшее (более того, *совершенное* — все вершины им покрыты)

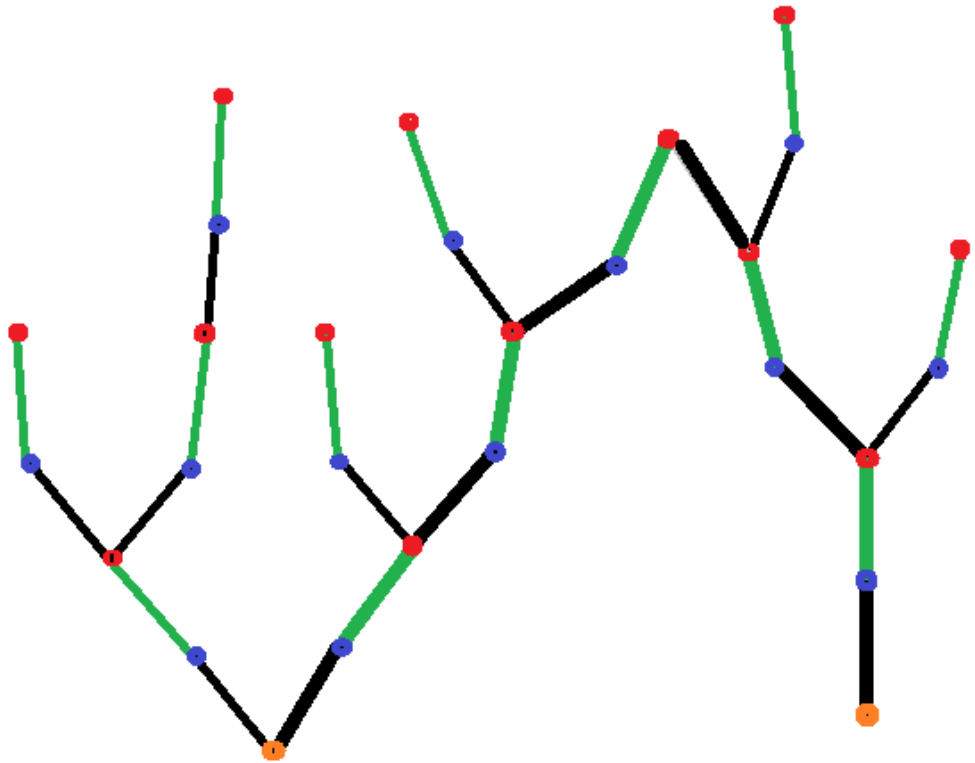
если осталась ровно *одна* непокрытая вершина, то M — также наибольшее (*почти совершенное*)

б.о.о. предположим, что есть не менее *двух* непокрытых вершин

Определение

Путь $(v_0, v_1, \ldots, v_{2N+1})$ *между непокрытыми вершинами* v_0, v_{2N+1} *называется увеличивающим если* $(v_{2k}, v_{2k+1}) \in E \setminus M$ *и* $(v_{2k-1}, v_{2k}) \in M$ *для всех* k.

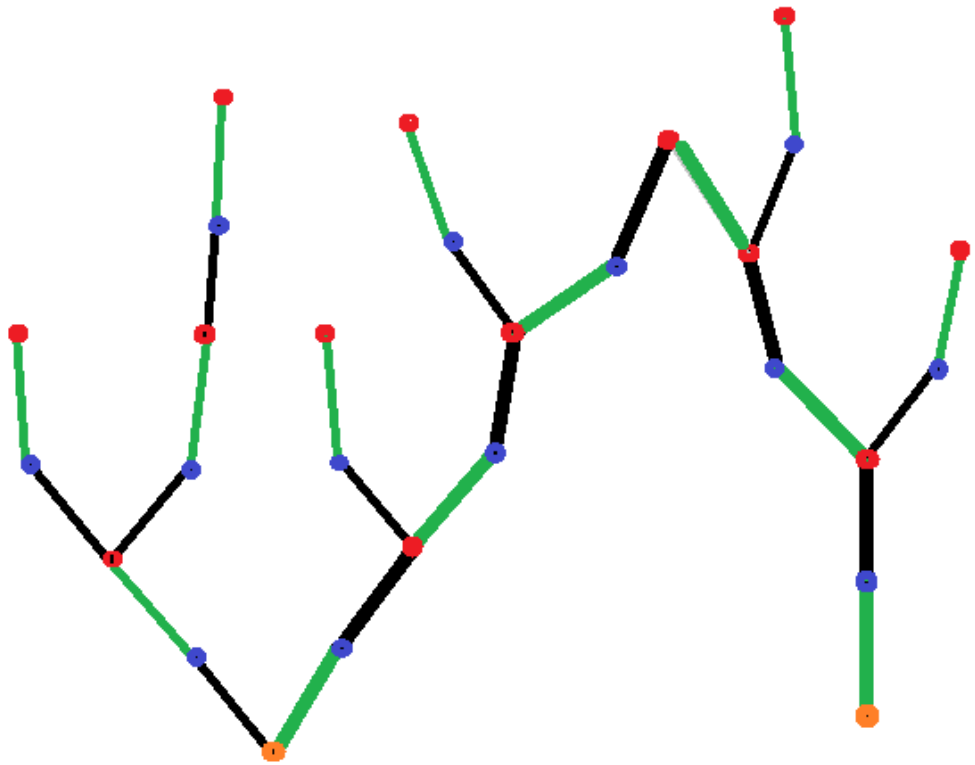
Наибольшее паросочетание



жирные рёбра — увеличивающий путь
оранжевые вершины — непокрытые
зелёные рёбра — элементы паросочетания
чёрные рёбра — вне паросочетания

Наибольшее паросочетание

если задан увеличивающий путь, то можно *увеличить* паросочетание M, переназначив $(v_{2k-1}, v_{2k}) \notin M$ и $(v_{2k}, v_{2k+1}) \in M$



Наибольшее паросочетание

Лемма (Берже)

Пусть M — паросочетание, не являющееся наибольшим. Тогда существует увеличивающий путь.

пусть M' — паросочетание и $|M| < |M'|$
определим граф $G' = (V, E')$, где $E' = (M \cup M') \setminus (M \cap M')$

тогда в G'

- любая вершина может соединять максимально два ребра
- рёбра из M, M' в путях чередуются
- компоненты связности — синглтоны, чётные циклы, пути

так как $|M'| > |M|$, есть компонента связности, в которой больше рёбер из M', чем из M

этот путь будет увеличивающим в G

Наибольшее паросочетание

алгоритм *сжатия цветков* находит увеличивающий путь за полиномиальное число операций

задача о наибольшем паросочетании решается повторным поиском увеличивающего пути

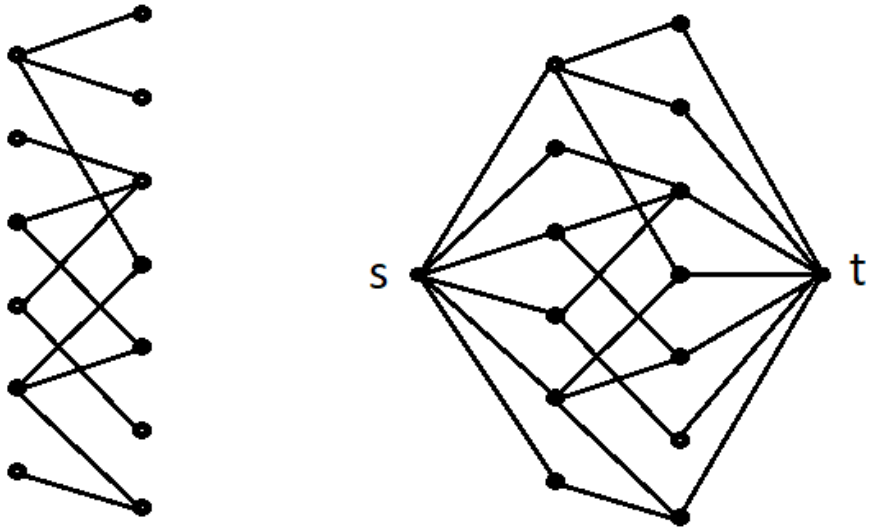
сложность алгоритма $O(|E| \cdot |V|^2)$

для более узких классов графов есть более быстрые алгоритмы

наибольшее паросочетание — не обязательно совершенное или почти совершенное

Наибольшее паросочетание: двудольные графы

для двудольного графа задача сводится к задаче о максимальном потоке



слева: исходный двудольный граф
справа: вспомогательный граф, в котором решается задача о максимальном потоке (все пропускные способности промежуточных рёбер — единичные, начальных и конечных — равны $|E|$)

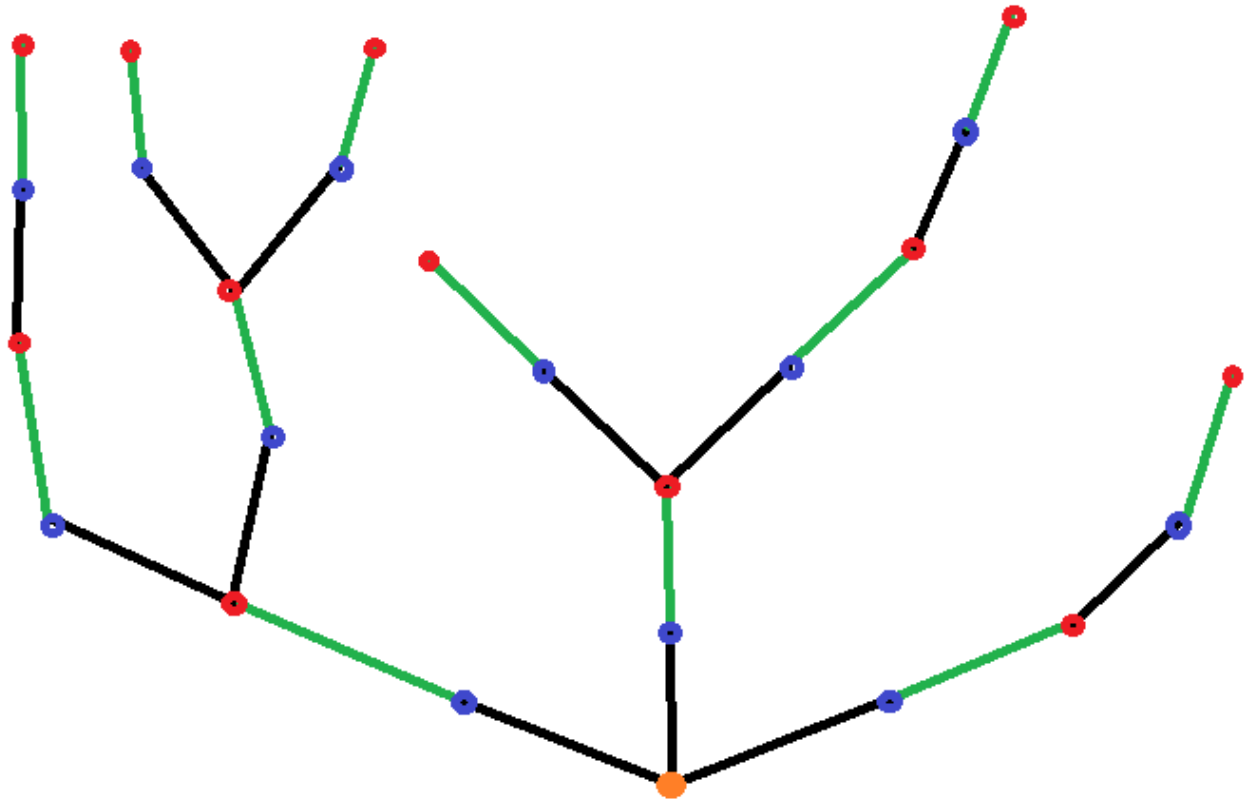
Алгоритм сжатия цветков

алгоритм строит лес из деревьев с корневой вершиной

- все рёбра деревьев являются рёбрами графа G
- все вершины деревьев являются дизъюнктными между собой подмножествами вершин графа G
- для ребра e, которое имеет вид (u_i, u_j) в дереве и (v_k, v_l) в графе G, имеем $v_k \in u_i$ и $v_l \in u_j$ или $v_k \in u_j$ и $v_l \in u_i$
- корневая вершина u_i дерева содержит ровно одну вершину $v_j \in V$, не покрытую текущим паросочетанием M
- остальные вершины дерева не содержат непокрытых вершин графа G
- любой путь в дереве чередующийся
- ветвления в дереве происходят только во *внешних* вершинах, от которых путь к корневой вершине начинается с ребра $e \in M$, или в самой корневой вершине

Алгоритм сжатия цветков

одно дерево леса



вершины: корневая, внешние, внутренние

рёбра: не из M, из M

Алгоритм сжатия цветков

свойства дерева в лесу

- листья дерева (кроме случая, когда корневая вершина — единственная в дереве) висят на рёбрах из паросочетания M и являются внешними
- ни одно ребро $e \in M$, не входящее в число рёбер дерева, не может быть смежным с вершиной дерева

построение деревьев

- начинаем с корней деревьев — это синглтоны $\{v\}$, где $v \in V$ — не покрытые
- на каждом шаге рассматривается ребро $e \in E \setminus M$, ещё не входящее в деревья и которое смежно некоторой вершине $v \in u \subset V$, где u — внешняя вершина некоторого дерева T

Алгоритм сжатия цветков

в зависимости от свойств $e = (v, v')$ (напомним $v \in u$):

a $v' \in u$: изменений не производится

b $v' \notin u'$ для всех вершин u' леса: e добавляется в дерево T, соединяя u с новой внутренней вершиной $u' = \{v'\}$; в T добавляется единственное ребро $e' = (v', v'') \in M$; e' соединяет u' с новой внешней вершиной $u'' = \{v''\}$

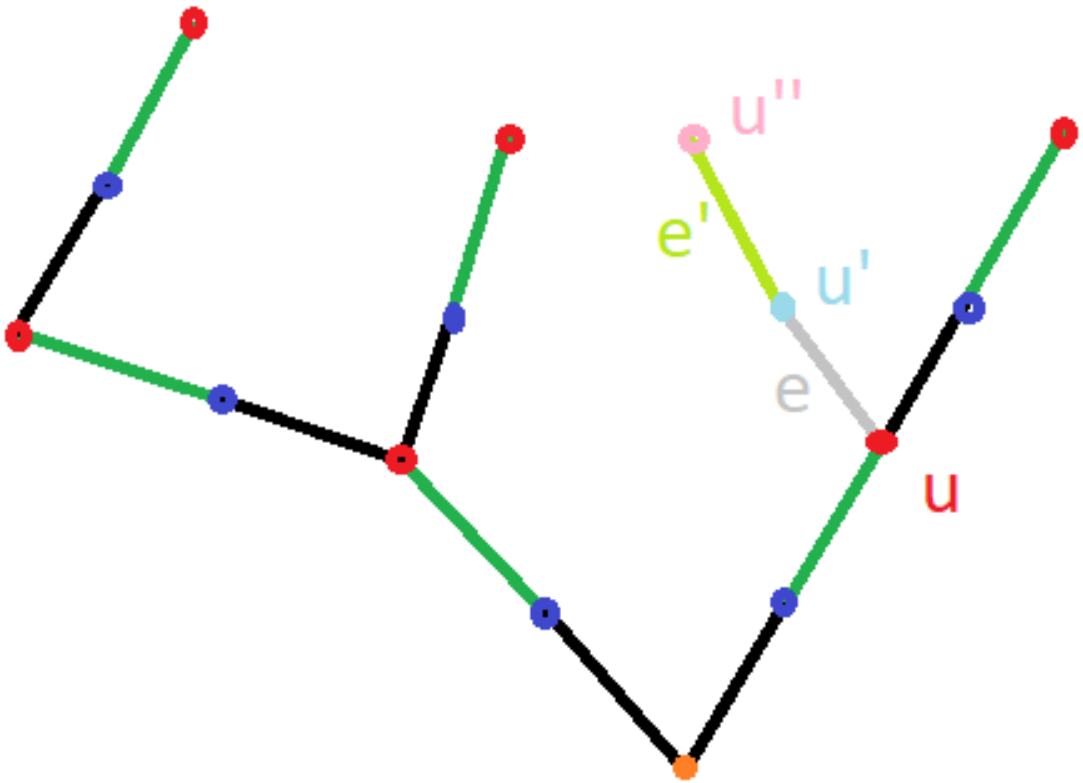
c $v' \in u'$, u' — внешняя вершина другого дерева T': можно построить увеличивающий путь

d $v' \in \tilde{u}$, $\tilde{u}$ — внешняя вершина T, и добавление e в T создаёт нечётный цикл (*цветок*): цикл стягивается в новую вершину u', которая объединяет все вершины из V, входящие в вершины цикла; все рёбра, исходящие из вершин цикла, будут исходить из новой вершины u'

e $v' \in u'$, u' — внутренняя вершина дерева T или другого дерева T': изменений не производится

Алгоритм сжатия цветков

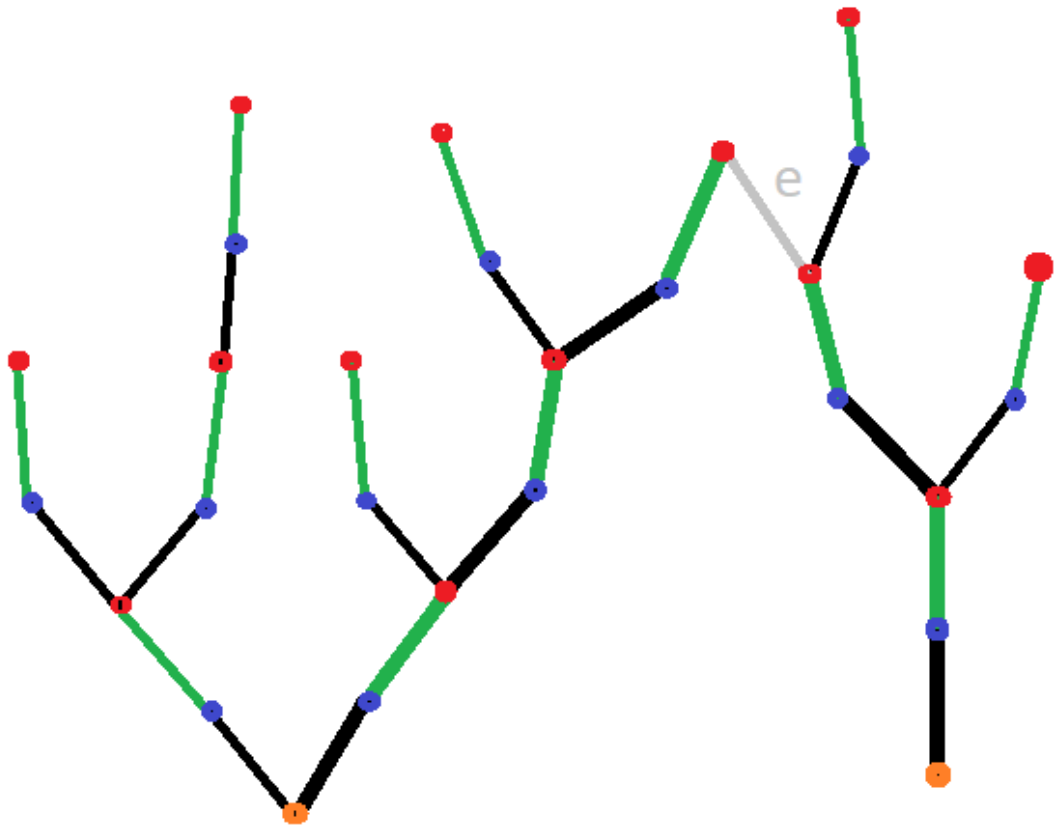
случай b



добавление в дерево двух новых вершин — внутренней и внешней, и двух новых рёбер — не из M и из M

Алгоритм сжатия цветков

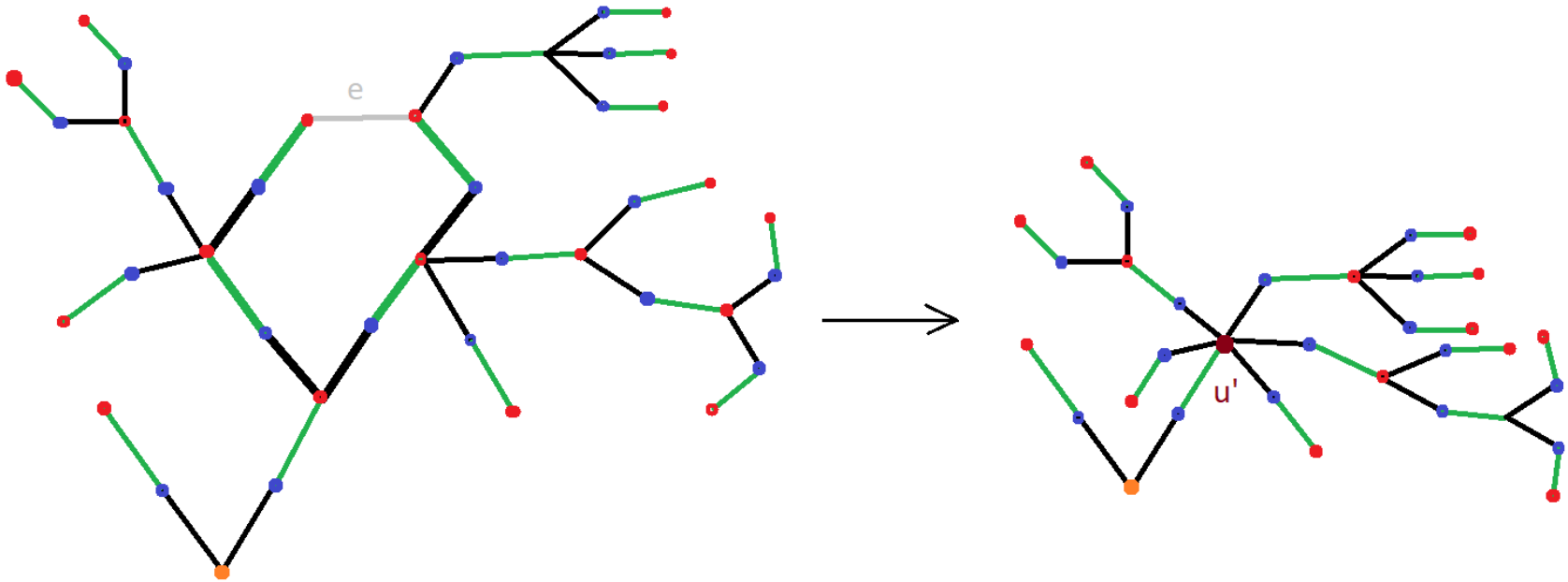
случай с



если найдётся ребро, соединяющее две внешние вершины разных деревьев, то через него можно построить увеличивающий путь (жирные рёбра)

Алгоритм сжатия цветков

случай d



добавление ребра e создаёт нечётный цикл (жирные рёбра), который сжимается в вершину u'

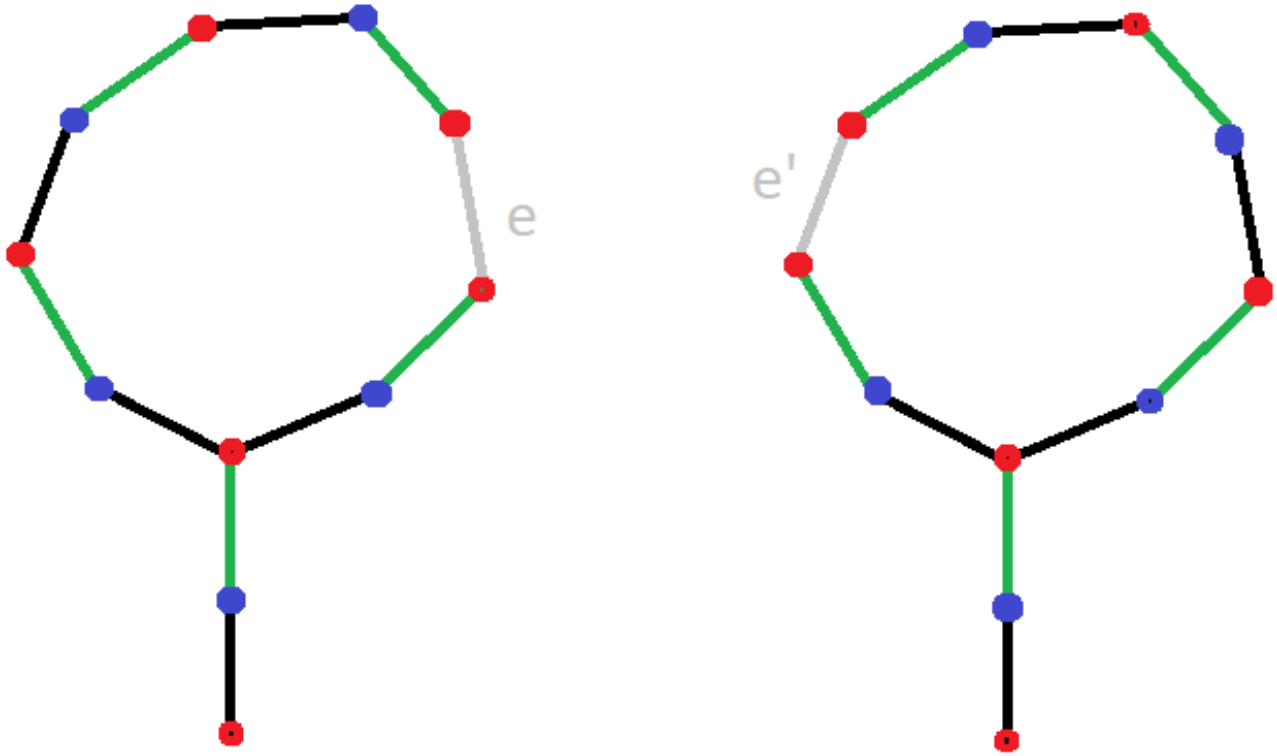
Алгоритм сжатия цветков

критерий останова

- если все рёбра, смежные с внешними вершинами деревьев, рассмотрены, и увеличивающий путь не найден, то его не существует
- увеличивающий путь, соединяющий две корневые вершины деревьев, поднимается до увеличивающего пути в G раскрытием сжатых цветков

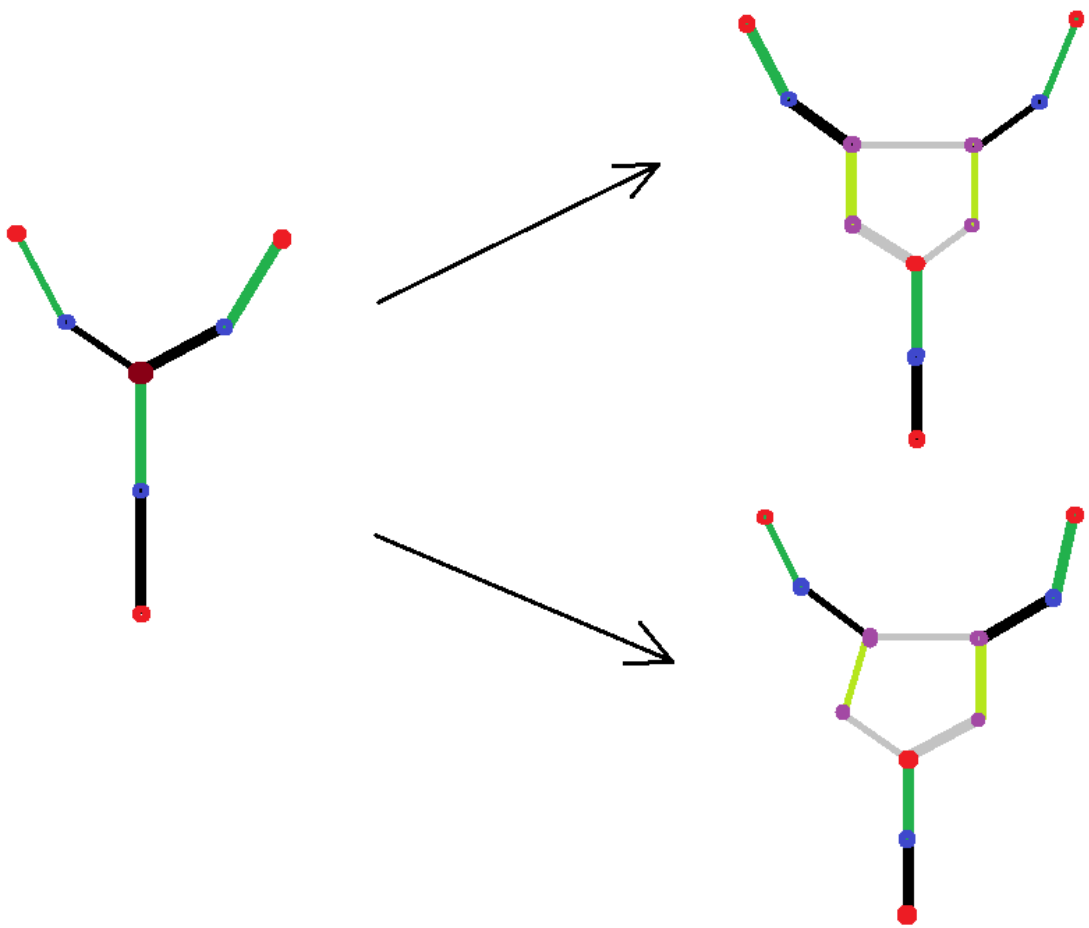
Алгоритм сжатия цветков

раскрытие цветков



вершины цветка при построении могут быть как внешними, так и внутренними (это зависит от местоположения ребра, которое замкнёт цикл)

Алгоритм сжатия цветков



как поднимать увеличивающий путь (жирные рёбра), зависит от того, через какую вершину он покидает цикл

План

простые комбинаторные задачи

- 2-SAT
- Horn-SAT
- кратчайший путь
- максимальный поток
- наибольшее паросочетание
- **хордальность / порядок совершенного исключения**
- сепарирующая клика

Заполнение в алгоритме Гаусса

рассмотрим линейную систему уравнений

$$Ax = b$$

где $A = A^T \succ 0$ — *разреженная* матрица

применим *алгоритм Гаусса* для факторизации матрицы

строка i матрицы преобразуется по формуле

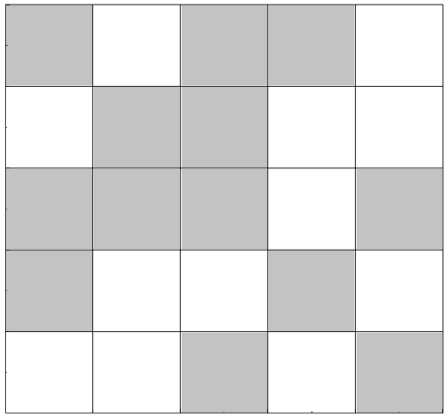
$$A_{i*} \leftarrow A_{*i} - \frac{A_{i1}}{A_{11}} A_{1*}$$

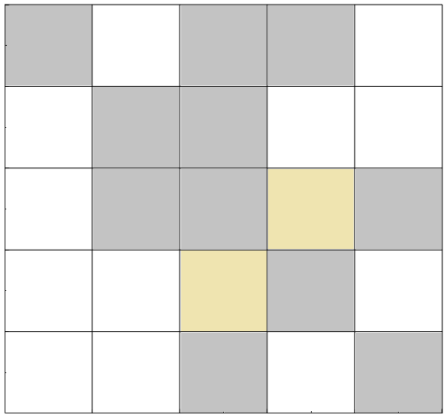
элементы матрицы меняются в ячейках

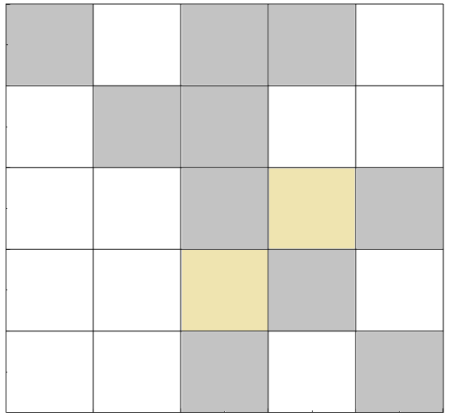
$$\{A_{ij} \mid A_{1i} \neq 0,\ A_{1j} \neq 0,\ i,j \neq 1\}$$

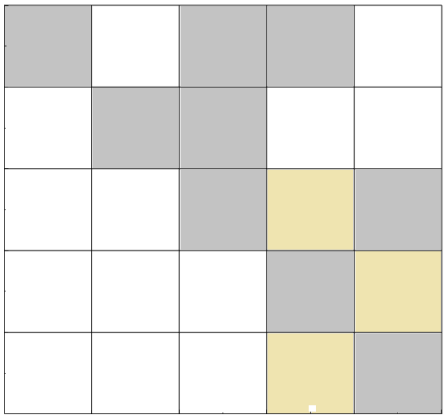
если в этих ячейках стояли нули, то происходит *заполнение*

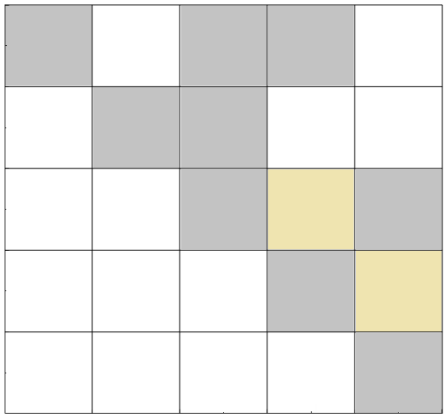
Заполнение в алгоритме Гаусса











Заполнение в алгоритме Гаусса

пусть шаблон заполнения задаётся графом $G = (V, E)$

вершины $V = \{1, \ldots, n\}$ соответствуют строкам матрицы

тогда на шаге k добавляются рёбра

$$A_{ij}: \quad (i,k) \in E,\ (j,k) \in E,\ \min(i,j) > k$$

т.е., смежные с k вершины с *большими* индексами образуют клику

Порядок совершенного исключения

Определение

Пусть $G = (V, E)$ — граф на n вершинах. Порядок $\alpha : V \to \{1, \ldots, n\}$ называется порядком совершенного исключения если для каждого $v \in V$ множество вершин

$$\{u \mid \alpha(u) > \alpha(v),\ (u, v) \in E\}$$

образуют клику.

если строки и столбцы матрицы A упорядочены в порядке совершенного исключения, то заполнения при факторизации не происходит

Когда граф имеет порядок совершенного исключения?

Если порядок существует, то как его найти?

Порядок совершенного исключения

Определение

Пусть $G = (V, E)$ — граф. G называется хордальным, если каждый цикл длины ≥ 4 имеет хорду.

Лемма

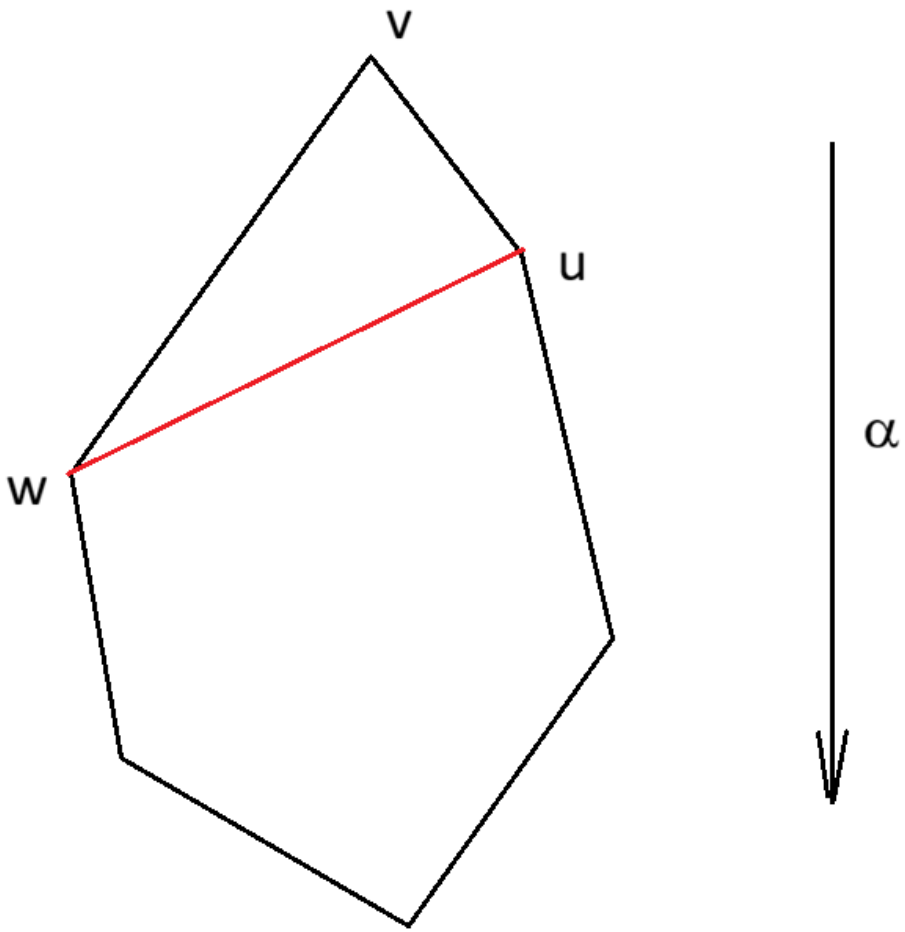
Если граф обладает порядком α совершенного исключения, то он хордальный.

пусть C — цикл длины ≥ 4

пусть $v \in C$ — вершина с минимальным индексом по отношению к α

тогда смежные с v в цикле C вершины u, w соединены хордой

Порядок совершенного исключения



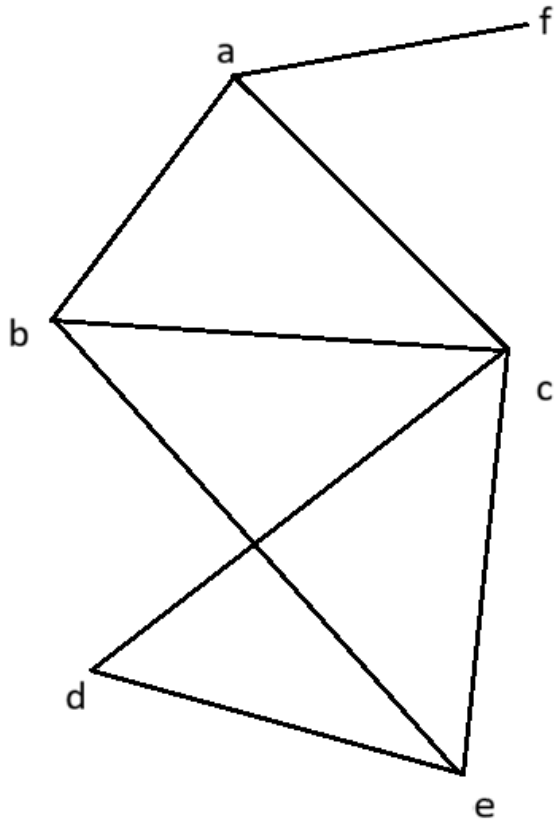
Порядок совершенного исключения

жадный алгоритм

построим порядок α вершин графа G

- последнюю вершину выберем произвольно
- смежные с последней вершиной вершины имеют больший порядковый номер, чем несмежные
- в каждой из двух групп, смежные с предпоследней вершиной вершины имеют больший порядковый номер, чем несмежные
- в каждой из 4-х групп, смежные с вершиной $\alpha^{-1}(n-2)$ вершины имеют больший порядковый номер, чем несмежные
- ...

Порядок совершенного исключения



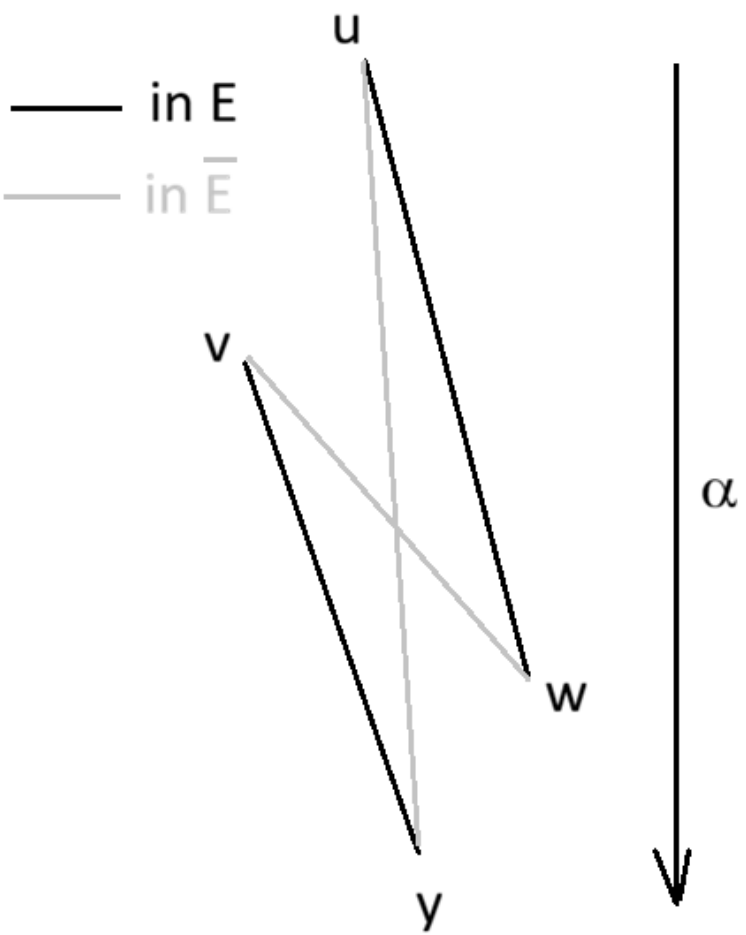
действия жадного алгоритма

- пусть $\alpha(e) = n = 6$
- тогда (a,f) < (b,c,d) < e
- пусть $\alpha(d) = 5$
- тогда (a,f) < b < c < d < e
- и $\alpha(c) = 4$
- тогда f < a < b < c < d < e

получаем порядок

$$f, a, b, c, d, e$$

Порядок совершенного исключения



алгоритм выдаёт порядок α (не обязательно совершенного исключения), удовлетворяющий следующему **Свойству 1**

если существуют вершины u, v, w с

$$\alpha(u) < \alpha(v) < \alpha(w)$$

такие, что $(u, w) \in E$, $(v, w) \notin E$, то существует y с

$$\alpha(w) < \alpha(y)$$

такая, что $(u, y) \notin E$, $(v, y) \in E$

Порядок совершенного исключения

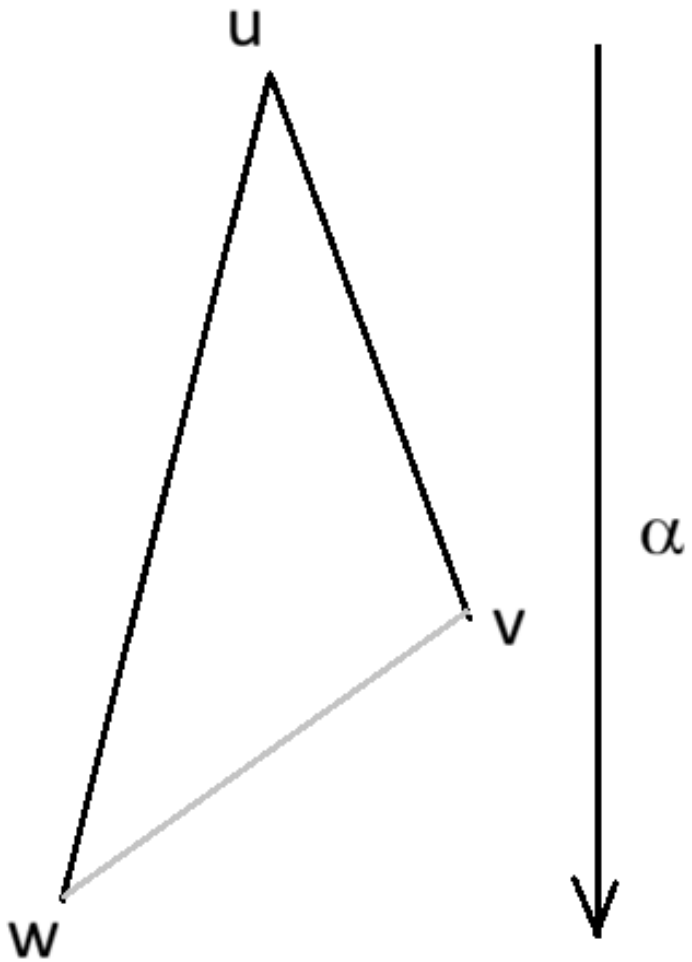
Теорема

Если порядок совершенного исключения существует, то алгоритм его выдает.

Теорема

Граф обладает порядком совершенного исключения тогда и только тогда, когда является хордальным.

Порядок совершенного исключения



пусть G — *хордальный граф* и алгоритм выдал порядок α вершин

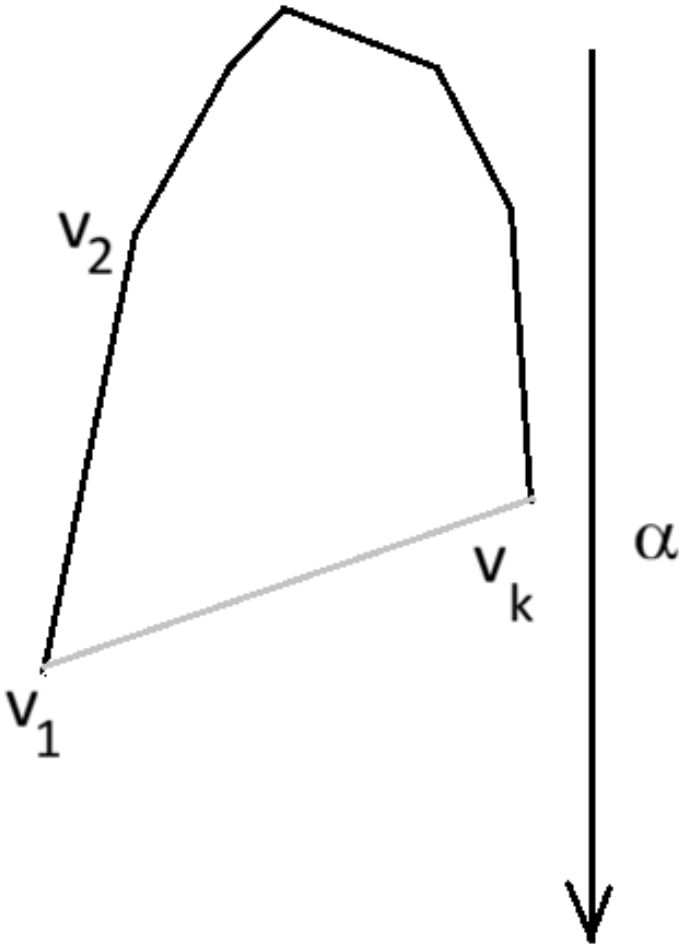
порядок является порядком совершенного исключения если **нет** ситуации слева

не существует вершин u, v, w с

$$\alpha(u) < \alpha(v) < \alpha(w)$$

и $(u, v) \in E$, $(u, w) \in E$, $(v, w) \notin E$

Порядок совершенного исключения



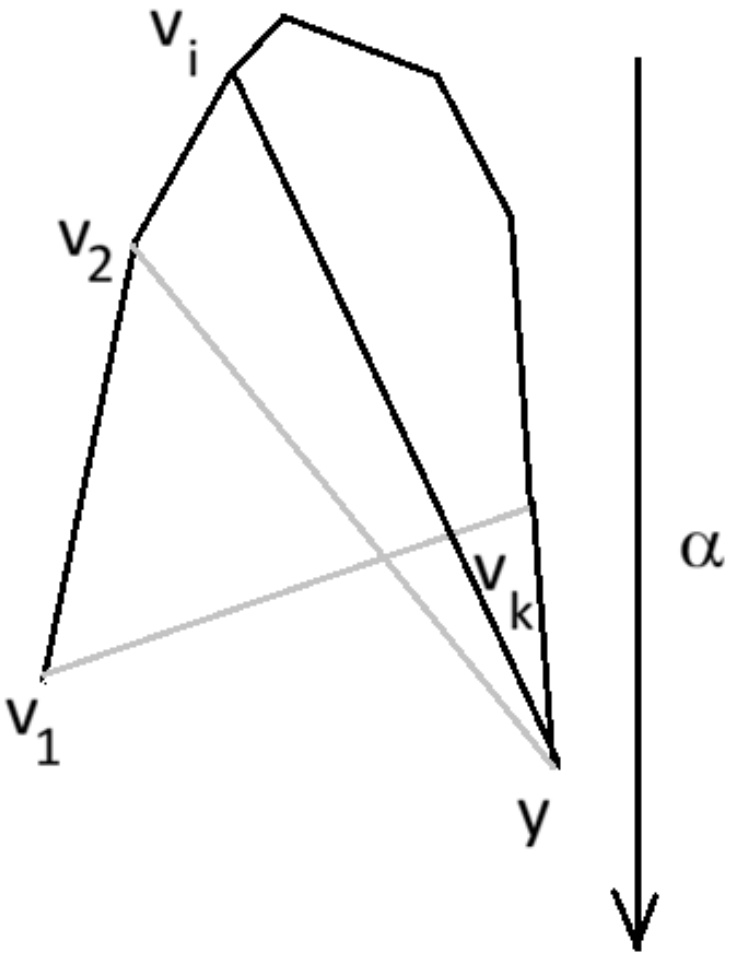
на самом деле верно более сильное утверждение

не существует пути $(v_1, \ldots, v_k)$ **без хорды** такого, что

$$\alpha(v_1) < \alpha(v_k) < \alpha(v_2)$$

допустим, что он есть, и $\alpha(v_1)$ — максимально

Порядок совершенного исключения



применим Свойство 1:
существует вершина y с

$$\alpha(v_2) < \alpha(v_k) < \alpha(v_1) < \alpha(y)$$

и $(v_k, y) \in E$, $(v_2, y) \notin E$

среди хорд $(v_3, y), \ldots, (v_k, y)$ есть (v_i, y) с наименьшим индексом i

- $(v_1, y) \notin E$ в силу хордальности G
- $(v_1, v_2, \ldots, v_i, y)$ — путь без хорды, и $\alpha(y) > \alpha(v_1) > \alpha(v_i)$

противоречие с максимальностью $\alpha(v_1)$

Порядок совершенного исключения

резюме

- если G — не хордальный, то порядка совершенного исключения нет
- если G — хордальный, то алгоритм выдаёт порядок совершенного исключения

Для данного графа, сколько минимум нужно добавить рёбер, чтобы получить хордальный граф?

- задача НП полная [Yannakakis 1981]
- существует алгоритм полиномиальной сложности, который гарантирует добавление не более, чем $8m^2$ рёбер, если на самом деле достаточно m рёбер [Natanzon 2000]
- существует суб-экспоненциальный алгоритм, решающий задачу точно [Fomin, Villanger 2013]

Частичное положительное заполнение матриц

пусть A — частично заполненная симметричная матрица с шаблоном заполнения, заданным графом G

пусть максимальным кликам G соответствуют неотрицательно определённые подматрицы (включая диагональные элементы)

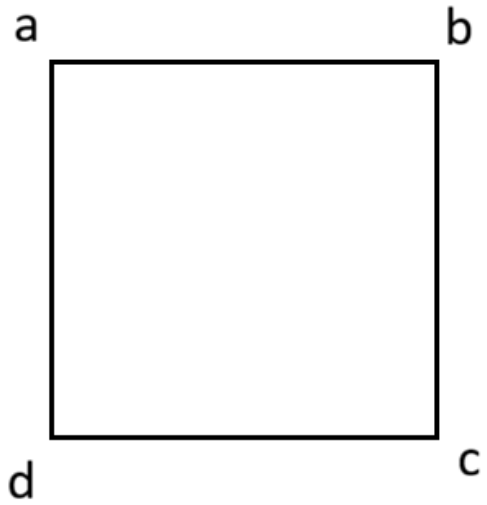
Когда гарантировано можно дополнить матрицу до неотрицательно определённой?

ответ: это всегда возможно тогда и только тогда, когда граф — хордальный [Grone, Johnson, Sa, Wolkowicz 1984]

Частичное положительное заполнение матриц

контр-пример: дополнение невозможно

	a	b	c	d
a	1	1		-1
b	1	1	1	
c		1	1	1
d	-1		1	1



Точность полуопределённых релаксаций

пусть $G = (V, E)$ — граф на вершинах $1, \ldots, n$

рассмотрим невыпуклую задачу

$$\min_{x \in \mathbb{R}^n} x^T C x : \qquad \|x\| = 1, \quad x_i x_j = 0 \ \forall \ i \neq j, \ (i,j) \notin E$$

где $C = C^T$ — произвольная матрица

если G хордальный, то задача эквивалентна (выпуклой) полуопределённой программе

$$\min_X \langle C, X \rangle : \qquad \operatorname{tr} X = 1, \quad X_{ij} = 0 \ \forall \ i \neq j, \ (i,j) \notin E$$

План

простые комбинаторные задачи

- 2-SAT
- Horn-SAT
- кратчайший путь
- максимальный поток
- наибольшее паросочетание
- хордальность / порядок совершенного исключения
- **сепарирующая клика**

Сепарирующая клика

Определение

Пусть $G = (V, E)$ — граф. Клику $C \subset V$ называют сепарирующей если индуцированный над $V \setminus C$ подграф — несвязный.

сепарирующие клики полезны, например, в решении задачи о минимальной покраске графа

для нахождения сепарирующих клик существует алгоритм полиномиальной сложности [Tarjan 1985]

Сепарирующая клика

Лемма

Пусть G — хордальный граф, C — сепарирующая клика, $(u, v) \in E$ — ребро в $G \setminus C$ такое, что его удаление разрушает хордальность.
Тогда u, v находятся в одной компоненте связности $G \setminus C$.

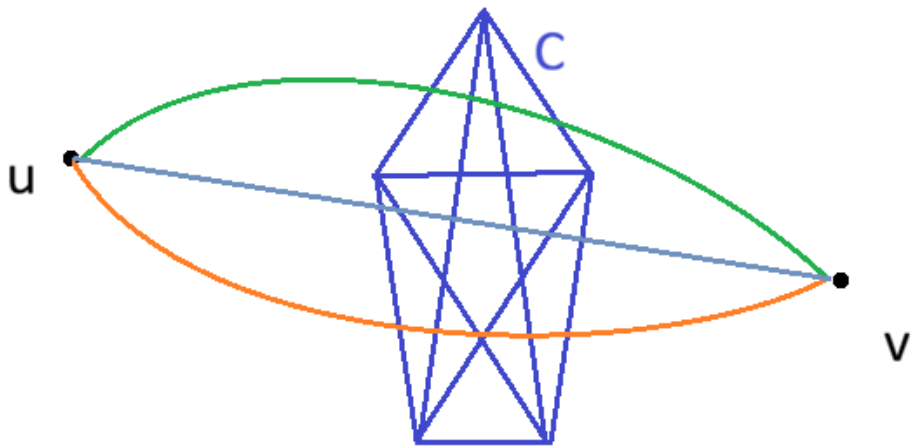
пусть G — произвольный граф

пусть E' — минимальное дополнение графа G до хордального

тогда все вершины, смежные с рёбрами из E', лежат в C или одной компоненте связности

т.е. в расширенном графе $G' = (V, E \cup E')$ компоненты связности в $G \setminus C$ всё те же

Сепарирующая клика



от противного: пусть u, v в разных компонентах

пусть Z — цикл, в котором $(u, v) \in E$ — единственная хорда

тогда Z дважды проходит через C

отсутствие второй хорды противоречит тому, что C — клика

Сепарирующая клика

алгоритм нахождения сепарирующей клики:

- построить минимальное дополнение E' до хордального графа
- упорядочить $G' = (V, E \cup E')$ в порядке α совершенного исключения
- проверить клики вида $C(v) = \{u \mid \alpha(u) > \alpha(v),\ (u, v) \in E\}$

среди них найдётся сепарирующая клика, если она существует

Сепарирующая клика

алгоритм основан на следующем результате

Лемма

Пусть E' и α — как выше. Пусть C — минимальная сепарирующая клика. Пусть $V_1, \ldots, V_k$ — компоненты связности подграфа $G \setminus V$. Для каждой компоненты V_i, пусть $v_i \in V_i$ максимизирует α. Пусть V_j — компонента с минимальным $\alpha(v_j)$. Тогда $C = C(v_j)$.

Сепарирующая клика

- рёбра из E' не могут связывать вершины из разных компонент
- для $c \in C$ и $V_i \neq V_j$ существует путь π из v_j в v_i через c, распадающийся при изъятии c на куски в V_i, V_j
- $\alpha(c) > \alpha(v_j)$ для всех $c \in C$; от противного
  - α на пути π имеет локальный минимум, так как $\alpha(c) < \alpha(v_j) < \alpha(v_i)$
  - минимумы можно "срезать" в силу свойства совершенного исключения
  - после срезания c возникнет ребро, связывающее V_i с V_j
- срезанием вершин в пути из v_j в c получим ребро $(v_j, c) \in E \cup E'$
- отсюда $C \subset C(v_j)$
- но в $C(v_j)$ не могут входит ни вершины из V_j (максимальность $\alpha(v_j)$), ни из V_i (разные компоненты связности)
- значит $C = C(v_j)$