Исследование операций

Задача о назначениях, Задача о наибольшем паросочетании и Задача о паросочетании наибольшего веса

Виктор Васильевич Лепин

План

- Задача о назначениях;
- Задача о наибольшем паросочетании в двудольных графах: чередующийся и пополняющий пути; лемма Бержа, теорема Кёнига; алгоритм Хопкровта-Карпа; ЦЛП-формулировка;
- Паросочетание максимального веса в двудольных графах: идея Эгервари; Венгерский алгоритм;
- наибольшее паросочетание в общих графах: алгоритм Эдмонса;
- Паросочетание максимального веса в общих графах;

История

- В 1784, G. Monge сформулировал задачу о назначениях.
- В 1916, D. König сформулировал задачу о паросочетаниях в двудольных графах.
- В 1931, J. Egerváry сформулировал задачу о паросочетании максимального веса.
- В 1950, von Neumann доказал эквивалентность задачи о назначениях и игры двух игроков с нулевой суммой.
- В 1955, H. Kuhn предложил Венгерский алгоритм.
- В 1957 году Munkres дал анализ венгерского алгоритма.
- В 1964, J. Edmonds предложил алгоритм решения задачи о паросочетаниях в общих графах.

1. Задача о назначениях и задача о наибольшем паросочетании в двудольных графах

Задача о назначениях в матричной постановке

- Рассмотрим следующую задачу о назначениях: есть три человека (обозначены как $I = \{I_1, I_2, I_3\}$) и три вакансии $J = \{J_1, J_2, J_3\}$. Любой человек может претендовать на одну или несколько рабочих мест. Эта информация может быть эффективно представлена **матрицей назначений**, где $q_{ij} = 1$, если I_i назначается на J_j, и $q_{ij} = 0$ в противном случае. Например,

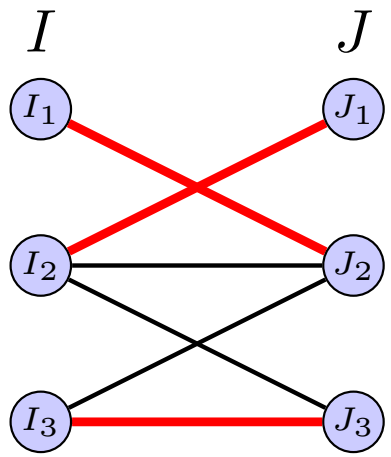
$$Q = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix}$$

- Вопрос: каково наибольшее количество рабочих мест, которые могут быть назначены квалифицированным специалистам (при этом не более одной должности назначается физическому лицу)? Другими словами, какое наибольшее количество 1 можно выбрать, если нет двух в одной строке или столбце?

Задача о назначениях в терминах теории графов

- Задача о назначениях может быть описана на языке теории графов, где узлы представляют людей и рабочие места, а ребра представляют назначения.

$$Q = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix}$$



- Т.о., в задаче о назначениях необходимо найти наибольшие паросочетание в двудольном графе.

ВХОД: двудольный граф $G = (L \cup R, E)$;
ВЫХОД: **паросочетание** M наибольшей размерности.
Отметим, что **паросочетание**, также называется **независимым множеством ребер**, т.е **множеством ребер без общих вершин**.

Паросочетание в графе

Определение

Паросочетанием (matching) простого графа называется его подмножество рёбер, никакие два из которых не имеют общего конца.

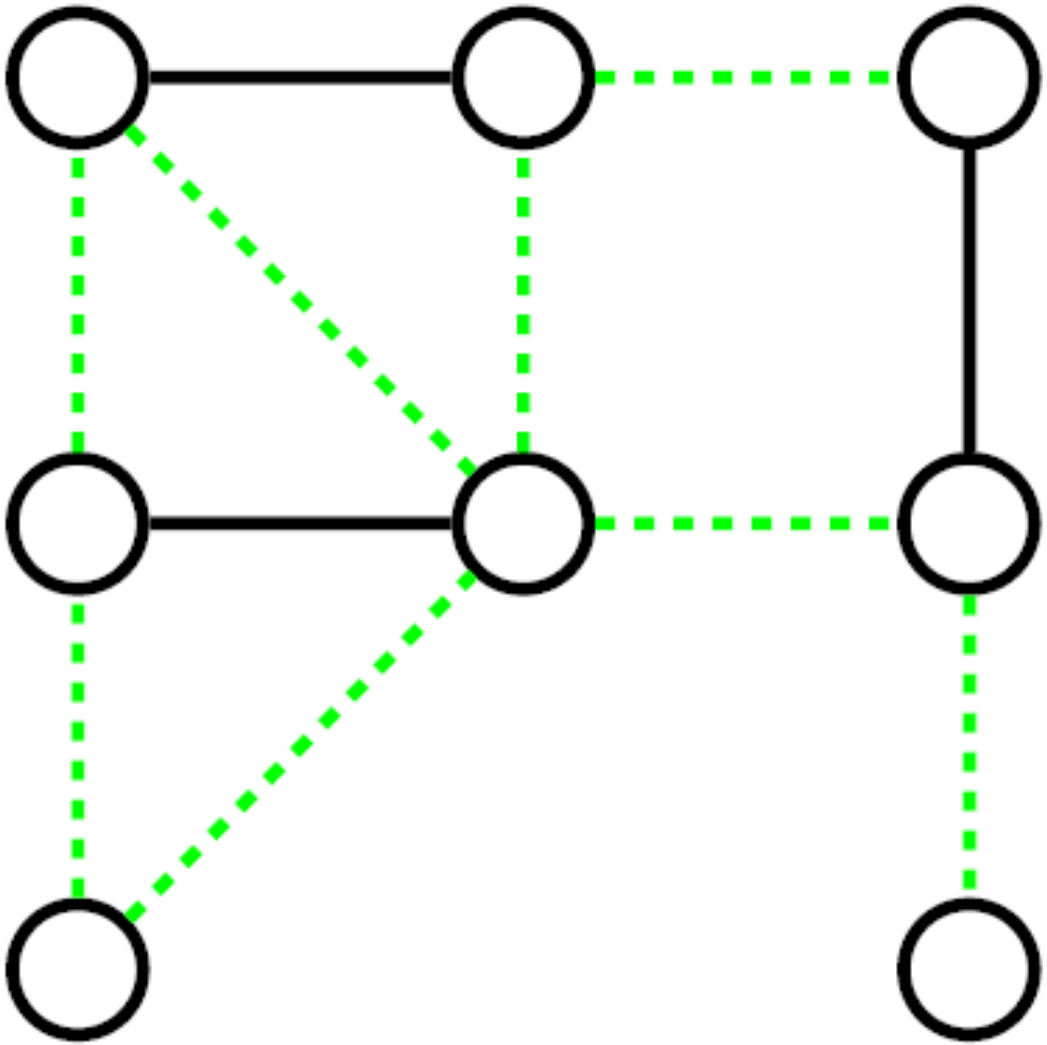
Паросочетание в графе

Определение

Паросочетанием (matching) простого графа называется его подмножество рёбер, никакие два из которых не имеют общего конца.

Задача о максимальном паросочетании (matching problem) заключается в нахождении по данному графу паросочетания максимального размера.

Пример паросочетания



1.1 Техника пополняющего пути

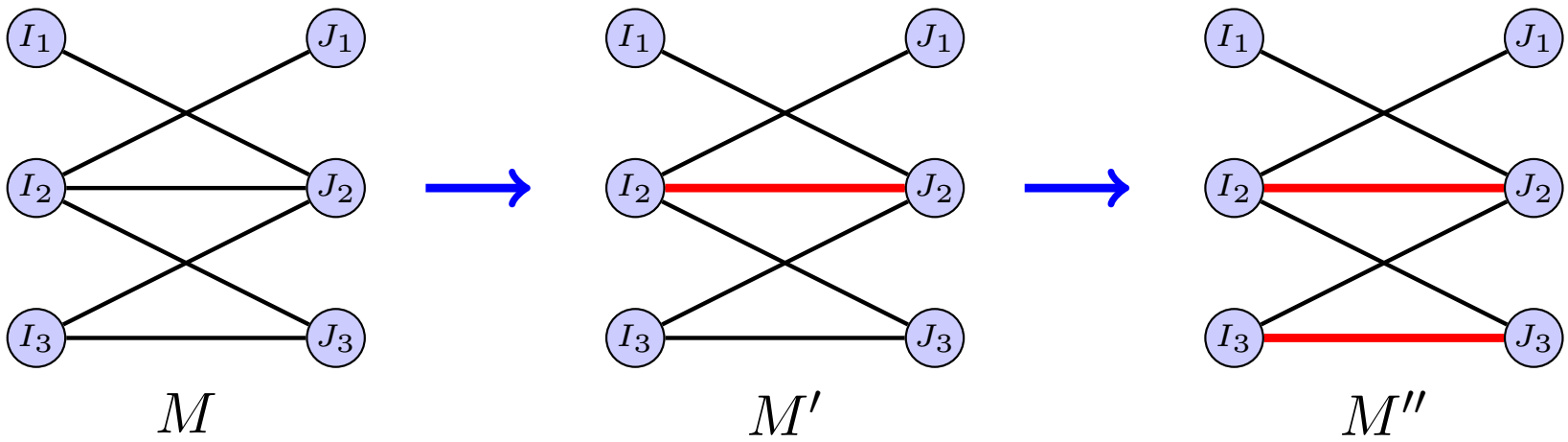
Применение общей стратегии Улудшения

1: $\mathbf{M} = \mathbf{M_0}$; //исходное решение;
2: **while** TRUE **do**
3: $\mathbf{M}$ =Улудшение($\mathbf{M}$); //движение к оптимуму;
4: **if** выполняется критерий остановки ($\mathbf{M}$) **then**
5: break;
6: **end if**
7: **end while**
8: **return** $\mathbf{M}$;

- Основная идея — поэтапно увеличивать мощность паросочетания.
- Установить начальное решение, например, нулевое паросочетание, тривиально. Таким образом, остается вопрос, как улучшить паросочетание и когда остановиться (т.е. условие оптимальности).

Как улучшить паросочетание? Простой способ

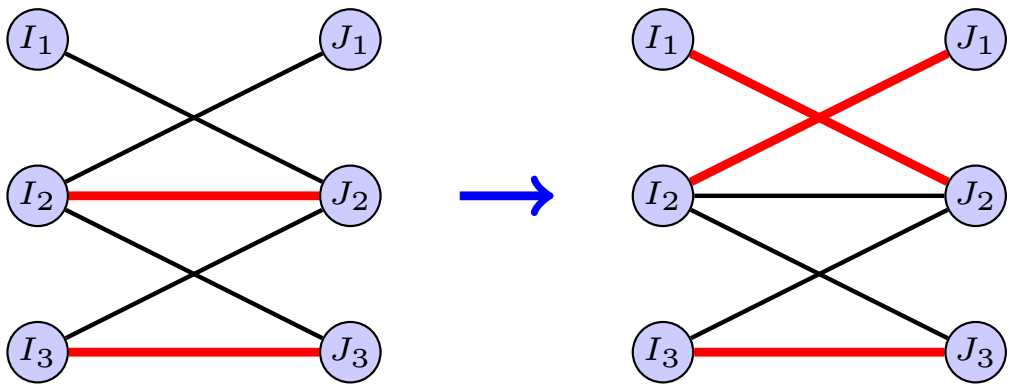
- Понятно, что мы можем увеличить паросочетание, просто назначив неназначенного сотрудника на неназначенную работу, которую он может выполнить.



- Обратите внимание, что паросочетание M'' не может быть улучшено. Фактически, M'' — это **максимальное паросочетание**, а не **наибольшее паросочетание**. Множество является **максимальным** по отношению к свойству, если оно имеет свойство, но не содержится ни в одном множестве, имеющем это свойство.

Более эффективная стратегия улучшения: Пополнение

- Обратите внимание, что ошибка заключается в выборе J_2 I_2. Таким образом, мы можем переместить I_2 из J_2 в J_1 и выбрать J_1 I_2, что увеличивает паросочетание на единицу.
- Эта операция (Пополнение) может быть реализована с помощью **пополняющего пути**.



Пополняющий путь

Определение

- Вершина называется сочетаемой относительно паросочетания M, если она является концом ребра из M.

Пополняющий путь

Определение

- Вершина называется сочетаемой относительно паросочетания M, если она является концом ребра из M.
- Вершина называется свободной (несочетаемой) относительно паросочетания M (free w.r.t. a matching M), если она не является концом ребра из M.

Пополняющий путь

Определение

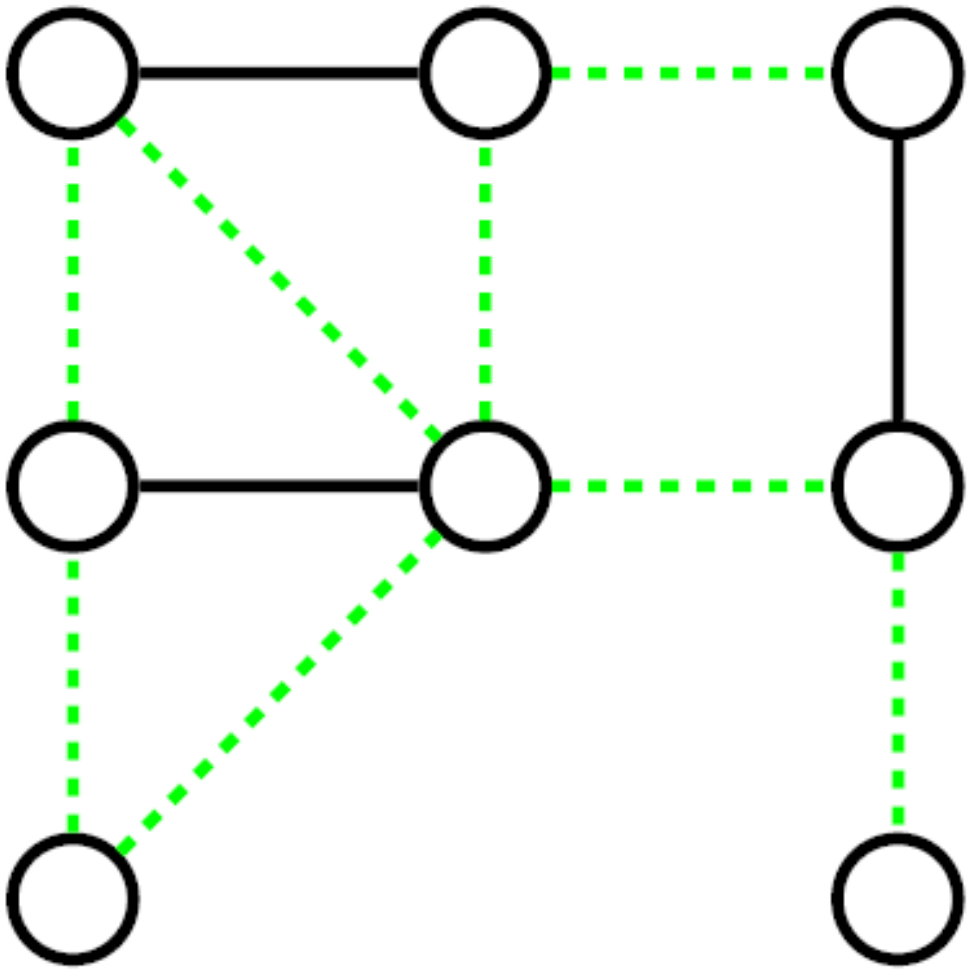
- Вершина называется сочетаемой относительно паросочетания M, если она является концом ребра из M.
- Вершина называется свободной (несочетаемой) относительно паросочетания M (free w.r.t. a matching M), если она не является концом ребра из M.
- Путь в графе называется чередующимся относительно M (alternating w.r.t. M), если в нем чередуются ребра из M и не из M.

Пополняющий путь

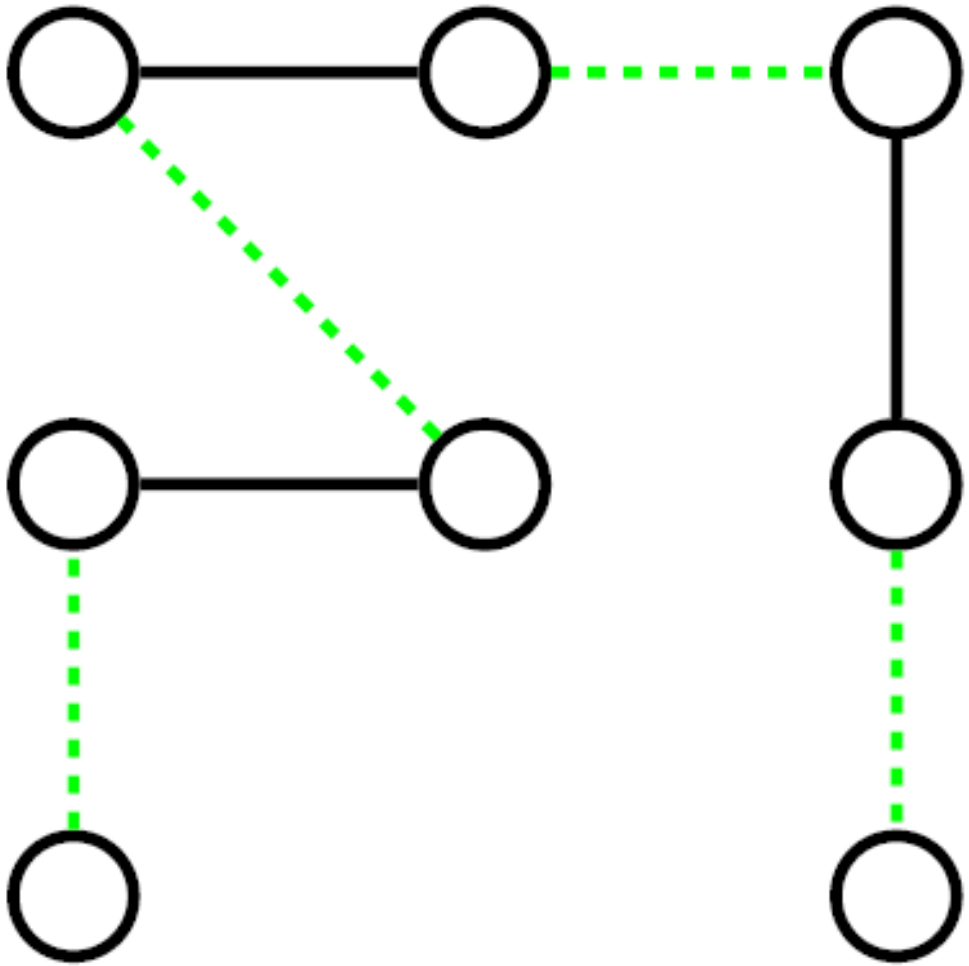
Определение

- Вершина называется сочетаемой относительно паросочетания M, если она является концом ребра из M.
- Вершина называется свободной (несочетаемой) относительно паросочетания M (free w.r.t. a matching M), если она не является концом ребра из M.
- Путь в графе называется чередующимся относительно M (alternating w.r.t. M), если в нем чередуются ребра из M и не из M.
- Путь называется пополняющим относительно M (augmenting w.r.t. M), если он является простым чередующимся путём между двумя совободными вершинами.

Пример пополняющего пути



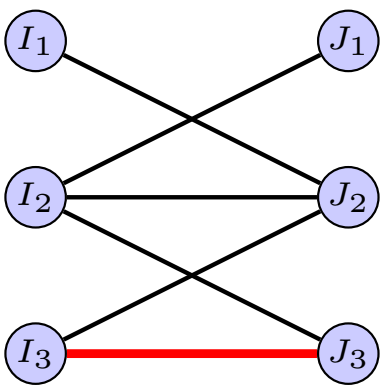
Пример пополняющего пути



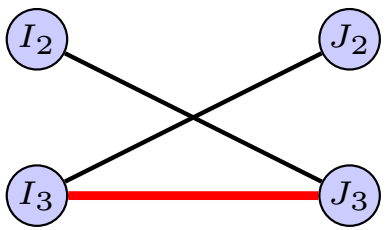
Улудшение с помощью пополняющего пути

Definition (Пополняющий путь и чередующийся путь)

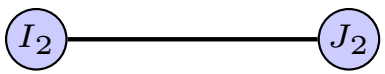
Чередующийся путь относительно паросочетания M содержит ребра поочередно из M и из $E - M$. Чередующийся путь, начинающийся и заканчивающийся несочетаемой вершиной, называется пополняющим путем, например $I_1 \to J_2 \to I_2 \to J_1$.



M



пополняющий путь 1



пополняющий путь 2

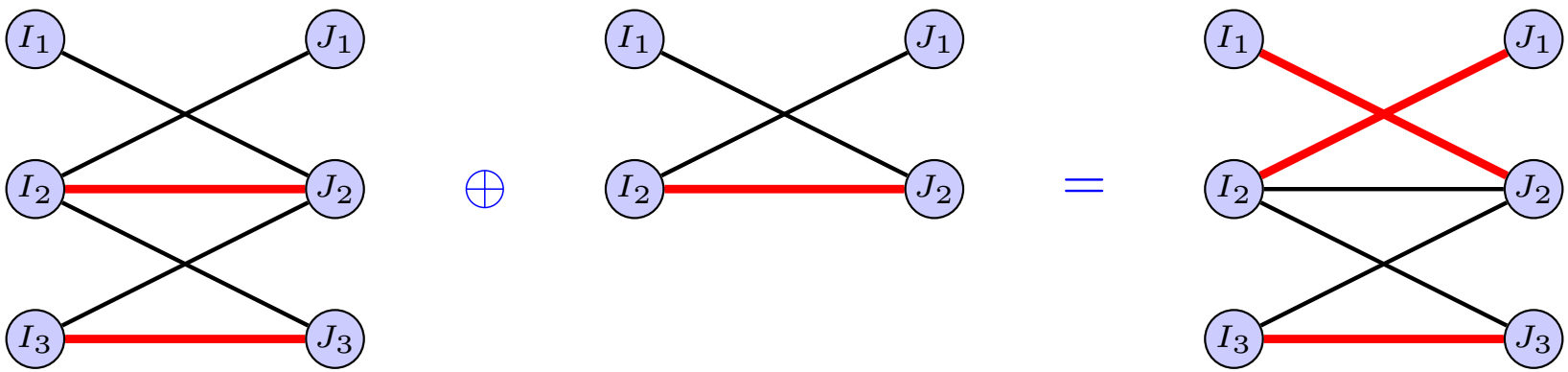
- Обратите внимание на то, что пополняющий путь относительно M имеет нечетное количество ребер, и ребер из $E - M$ на одно ребро больше, чем из M.

Улудшение M с использованием $M \oplus P$

- **Симметричная разница** паросочетания M и пополняющего пути P приводит к **паросочетанию большего размера**.

Lemma

Для пополняющего пути P относительно паросочетания M, **симметричная разность** $M \oplus P = (M \setminus P) \cup (P \setminus M)$ *меняет местами ребра в $P \cap M$ и те, что в $P \cap \overline{M}$, образуя новое паросочетание с еще одной парой, т. е. $|M \oplus P| = |M| + 1$.*

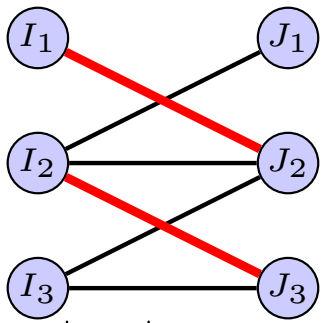


$|M| = 2$ P $|M'| = 3$

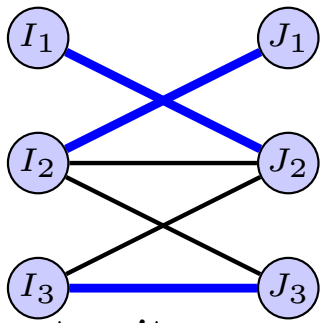
Условие оптимальности. Лемма Бержа [1957]

Theorem

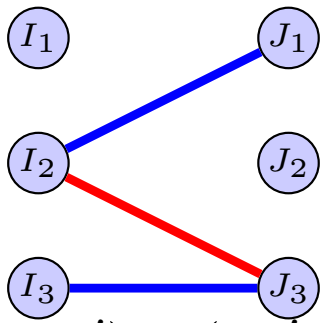
Паросочетание M в графе G является наибольшим, если нет пополняющего пути относительно M



$|M| = 2$



$|M'| = 3$



$(M - M') \cup (M' - M)$

- Ключевая идея: предположим от противного, что существует другое паросочетание M', большее, чем M. **Симметричная разность** $M \oplus M' = (M - M') \cup (M' - M)$ содержит **пополняющий путь**.
- Так что $|M'| > |M|$, по крайней мете на единицу **чередующийся путь** является **пополняющим путем** поскольку он содержит на одно ребро в M' больше, чем в M (т.е., $J_1 - I_2 - J_3 - I_3$).

Доказательство.

- От противного. Предположим, что существует другое паросочетание M' с $|M'| > |M|$. Рассмотрим граф G' с ребрами взятыми из **симметрической разности** $M \oplus M' = (M - M') \cup (M' - M)$.
- Как M так и M' — паросочетания, каждый узел v в G' инцидентен самое большее одному ребру из $M - M'$ и самое большее одному ребру из $M' - M$. Следовательно, G' состоит из связных компонент следующего типа:
  1. изолированная вершина,
  2. чередующийся путь с концевыми вершинами различного типа, или
  3. цикл четной длинны с ребрами попеременно из M и M'.
- Поскольку M' больше чем M, то должен существовать чередующийся путь имеющий ребер из M' больше, чем из M, этими ребрами можно заменить ребра пути из M. Получили противоречие.

□

Отметим, что лемма Бержа справедлива для общих графов.

Общий вид алгоритма

MAXIMUM-MATCHING(G)

1 $M \leftarrow \emptyset$
2 **repeat**
3 **if** есть дополняющий путь P относительно M
4 **then** $M \leftarrow M \oplus P$
5 **else return** M

Случай двудольных графов

- Граф называется двудольным, если его множество вершин может быть разбито на две части (доли) V_1 и V_2, так что любое ребро графа соединяет вершины разных долей.

Случай двудольных графов

- Граф называется двудольным, если его множество вершин может быть разбито на две части (доли) V_1 и V_2, так что любое ребро графа соединяет вершины разных долей.
- В двудольном графе каждый цикл имеет чётную длину.

Случай двудольных графов

- Граф называется двудольным, если его множество вершин может быть разбито на две части (доли) V_1 и V_2, так что любое ребро графа соединяет вершины разных долей.
- В двудольном графе каждый цикл имеет чётную длину.
- Задача о максимальном паросочетании в двудольном графе сводится к задаче о максимальном потоке.

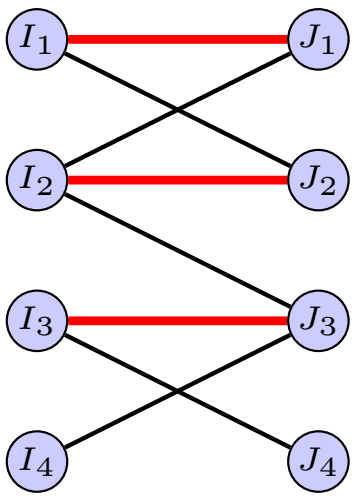
Наибольшее паросочетание в двудольных графах

- Поговорим о следующих вопросах.
  - Сведение к задаче о максимальном потоке и алгоритм Хопкровта-Карпа
  - Формулировка в форме задачи ЦЛП и двойственная задача
  - Теорема Кенига: наибольшее паросочетание и наименьшее вершинное покрытие.
  - Теорема Холла.

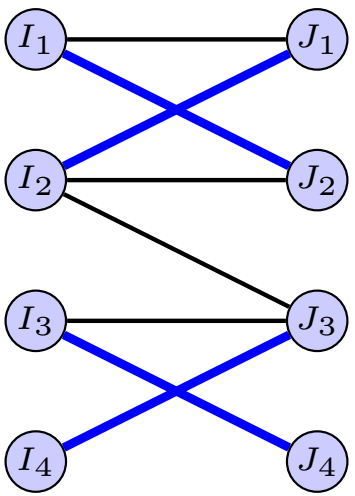
Версия Хопкровта- Тарьяна [1975]

Theorem

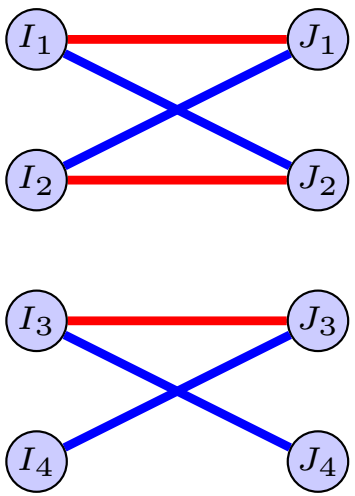
Пусть M и M' — паросочетания, $|M| = r$, $|M'| = s$, и $s > r$, тогда $M \oplus M'$ содержит по крайней мере $s - r$ **вершинно-различных пополняющих путей** *относительно M.*



$|M| = 2$



$|M'| = 4$



$M \oplus M'$

Доказательство.

- Рассмотрим граф G' с ребрами взятыми из **симметрической разности** $M \oplus M' = (M - M') \cup (M' - M)$.
- Поскольку M и M' — паросочетания, то каждый узел v в G' инцидентен самое большее с одним ребром из $M - M'$ и самое большее с одним ребром из $M' - M$. Следовательно, G' состоит из связных компонент следующего типа:
  1. изолированная вершина,
  2. чередующийся путь с концевыми вершинами различного типа, или
  3. цикл четной длинны с ребрами попеременно из M и M'.
- Для каждой компоненты $C_i = (V_i, E_i)$, определим $\delta(C_i) = |E_i \cap M'| - |E_i \cap M|$. Тогда $\delta(C_i) \in \{-1, 0, 1\}$. Отметим, что $\delta(C_i) = 1$ если и только если C_i является пополняющим путем относительно M.
- Также имеем $\sum_i \delta(C_i) = s - r$.
- Следовательно существует по меньшей мере $s - r$ пополняющих путей относительно M.

Алгоритм

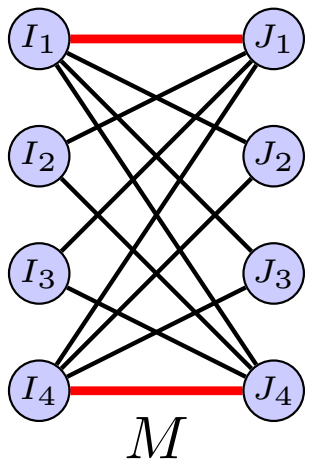
1: $\mathbf{M} = \{\}$; //исходное паросочетание является пустым;
2: **while** TRUE **do**
3: найти пополняющий путь P относительно M;
4: **if** пополняющего пути нет **then**
5: break;
6: **end if**
7: пополнить M используя P : $M = M \oplus P$;
8: **end while**
9: **return** $\mathbf{M}$;

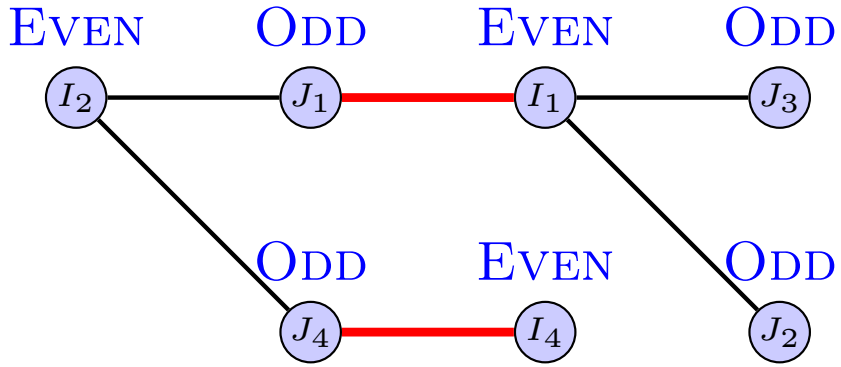
- Трудоемкость: $O(mn)$
- Поскольку:
  - $\#\texttt{WHILE} = O(n)$ так как каждая пополняющая операция увеличивает паросочетание по крайней мере на 1.
  - Поиск пополняющего пути: $O(m)$ если использовать BFS.

Вопрос: как найти пополняющий путь относительно данного паросочетания?

Метод 1: выполнить BFS в исходном двудольном графе

- Идея: начнем поиск из **свободной вершины** r, попытаемся идент-вать **все вершины, до которых существует чередующийся путь из** r. Если свободная вершина v была посещена, то мы получаем пополняющий путь из r в v.
- Эти вершины могут быть идентифицированы, используя **расширенный BFS**: когда сочетаемая вершина посещается, то пара ребер, одно из M, а другое из $E - M$, должны быть добавлены.
- Подобно BFS дереву, расширенный BFS представляет чередующиеся пути как M**-чередующее дерево** с корнем r, где каждый путь от корня до листа представляет чередующийся путь.



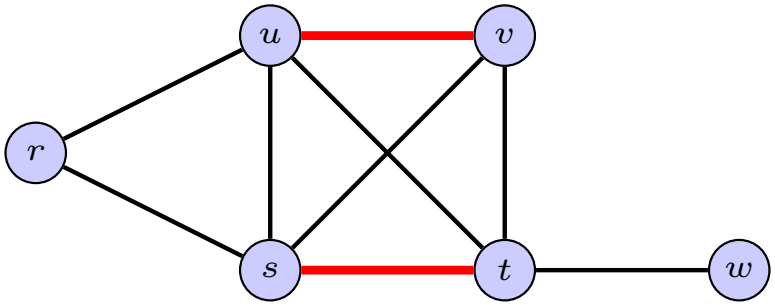


Чередующее дерево с корнем I_2

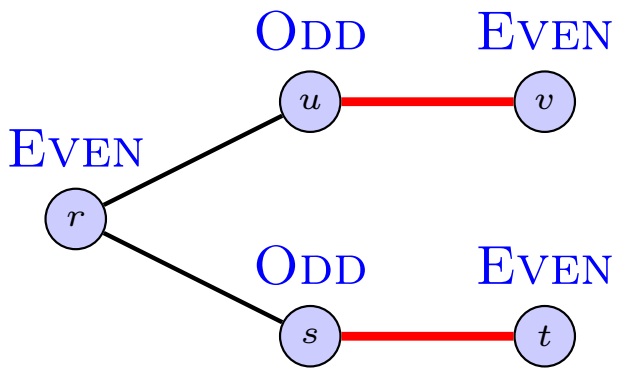
Расшириный BFS

- Метка “посещенная"в BFS расширяется: в чередующем дереве, вершины в четных листьях получают метку $\textsc{Even}$, а вершины в нечетных листьях получают метку $\textsc{Odd}$.
- Расширенный BFS выполняется за $O(m+n)$ шагов:
  - в начале все вершины непомечены.
  - Следующий шаг повторяется пока все вершины не будут исследованы. На каждом шаге, мы либо выбираем свободную непомеченную вершину r, помечаем ее меткой $\textsc{Even}$, создаем новое дерево с корнем r, или выбираем $\textsc{Even}$-вершину u, и исследуем все ребра (u,v) инцидентные ей:
    - Если v — свободная, то сообщаем о **пополняющем пути** из r в v.
    - Если v — сочетаемая (с w) но не имеет метки, то добавляем v, w, и ребро (v,w) в дерево, и помечаем v меткой $\textsc{Odd}$, w меткой $\textsc{Even}$.
    - Если v — сочетаемая и помечена меткой $\textsc{Odd}$, ничего не делаем (так как это означает, что **еще один нечетной длины путь** из r в v найден).
    - Если v — сочетаемая и помечена меткой $\textsc{Even}$, то найден **цикл нечетной длины** (называемый **цветком**).

Пример: Шаг 1



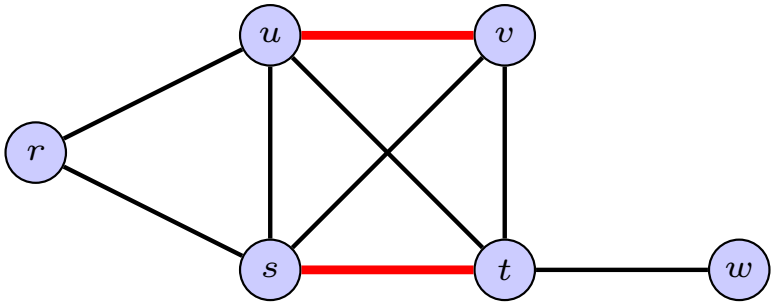
M



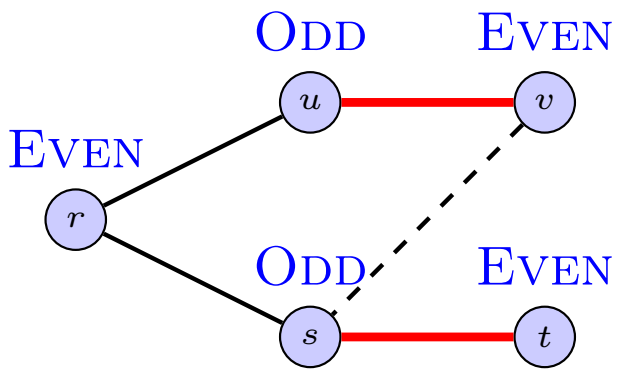
Чередующее дерево

- Стартуя из Even-вершины r, мы найдем смежную сочетаемую вершину u.
- Т.о., и u и v добавляются к дереву с метками.
- Также происходит с другой смежной вершиной s.

Пример: Шаг 2



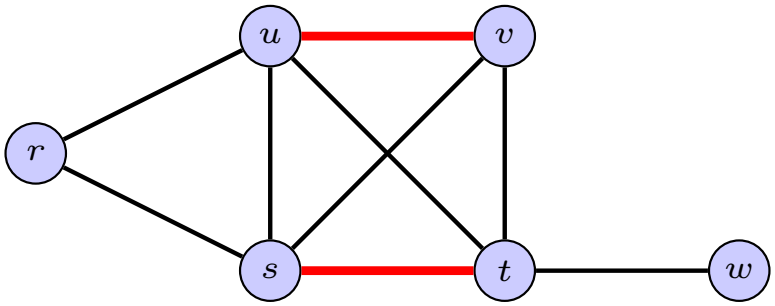
M



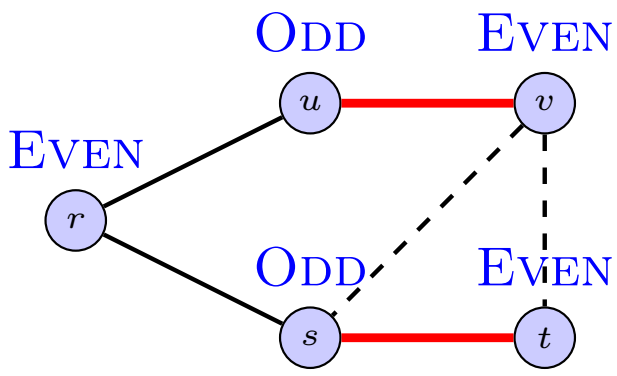
Чередующее дерево

- Стартуем из Even-вершины v, находим Odd-вершину s.
- Это означает, что найден чередующийся путь $r - s$, и найден другой нечетной длины чередующийся путь $r - u - v - s$. Т.о., ничего не нужно делать.

Пример: Шаг 3



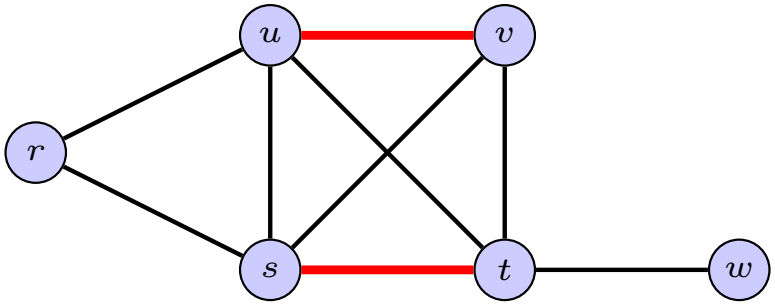
M



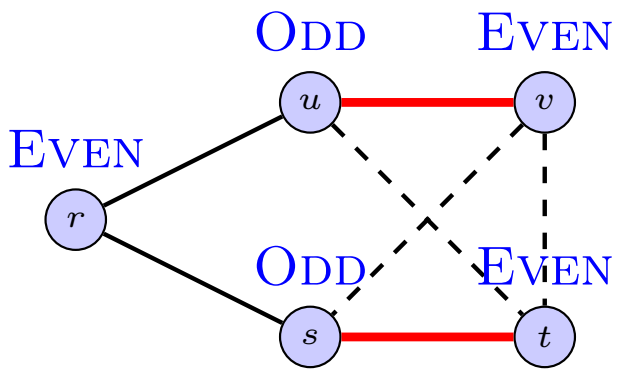
Чередующее дерево

- Стартуем из EVEN-вершины v, находим EVEN-вершину t.
- Найден нечетной длины цикл $r - u - v - t - s - r$ (называемый **цветком**).

Пример: Шаг 4



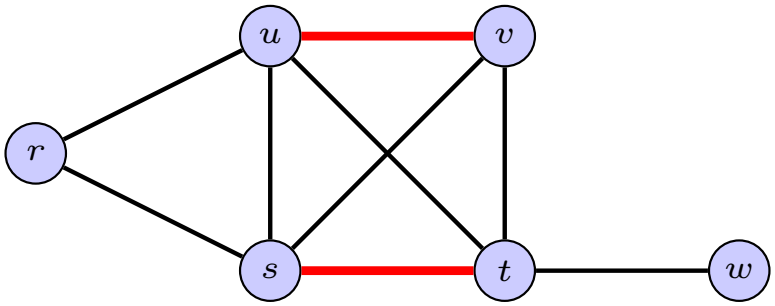
M



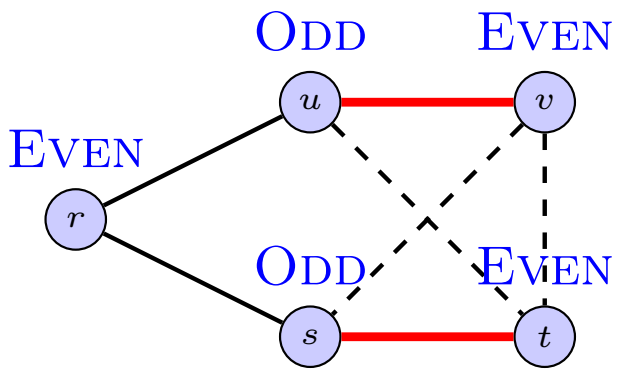
Чередующее дерево

- Стартуем из Even-вершины t, находим Odd-вершину u.
- Это означает, что найден чередующийся путь $r - u$ (нечетной длины чередующийся путь $r - s - t - u$). Т.о., ничего не нужно делать.

Пример: Шаг 5



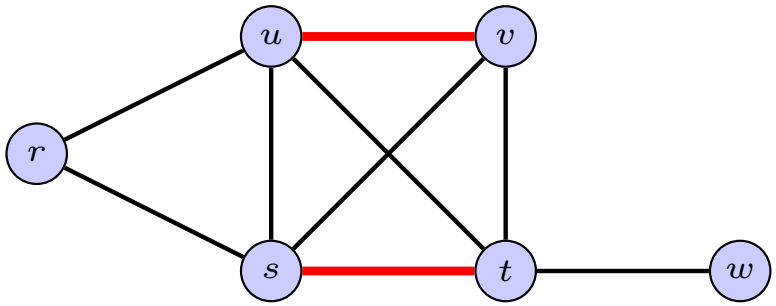
M



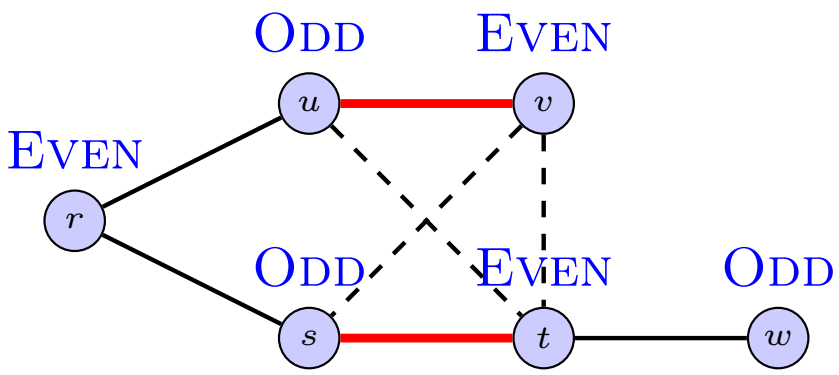
Чередующее дерево

- Стартуем из Even-вершины t, находим Even-вершину v.
- Сообщаем о нечетной длины цикле $r - u - v - t - s - r$ (найден **цветок**).

Пример: Шаг 6



M

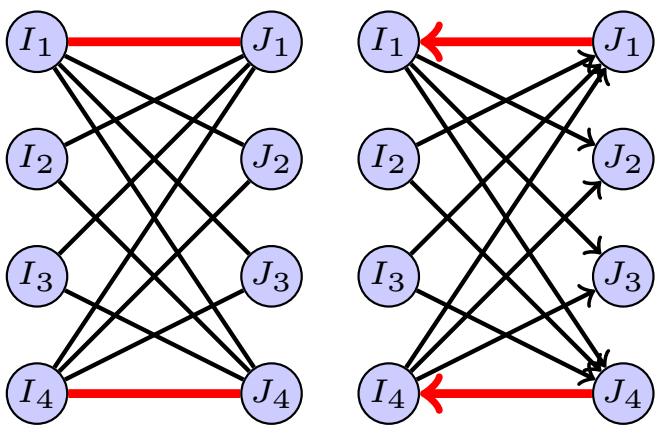


Чередующее дерево

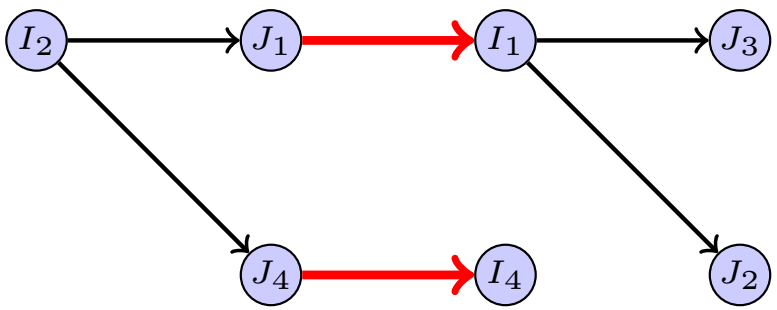
- Стартуем из Even-вершины t, найдена свободная вершина w.
- Сообщаем о чередующемся пути $r - s - t - w$.

Метод 2: выполняется стандартный BFS в ориентированном графе

- Пусть M — паросочетание в двудольном графе $G = (L \cup R, E)$. Соответствующий **орграф** G_M может быть построен назначением ребрам из M направления от R к L, и другим ребрам (из $E - M$) направления от L к R.
- Задача нахождения чередующихся путей начинающихся от свободной вершины r трансформируется в задачу нахождения достижимых вершин в G_M, что можно сделать BFS за время $O(m + n)$.



M G

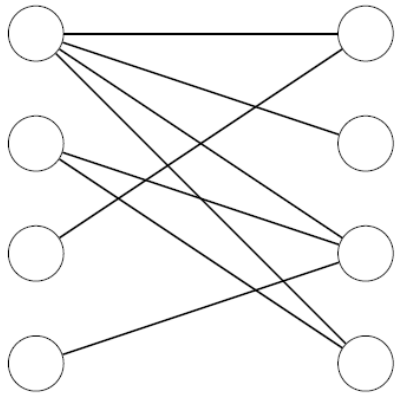


Чередующее дерево с корнем I_2

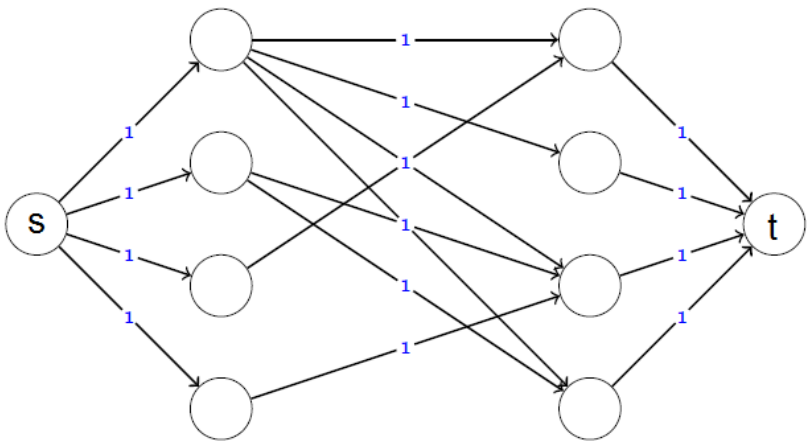
1.2 Нахождение Наибольшего паросочетания используя технику максимального потока в сети

Стратегия: поток в сети

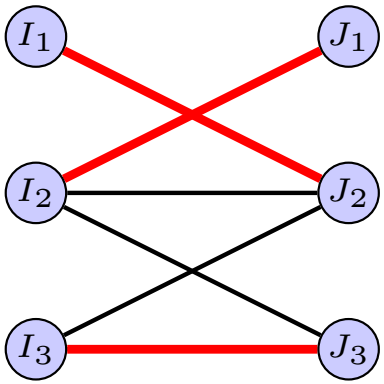
- Дан двудольный граф



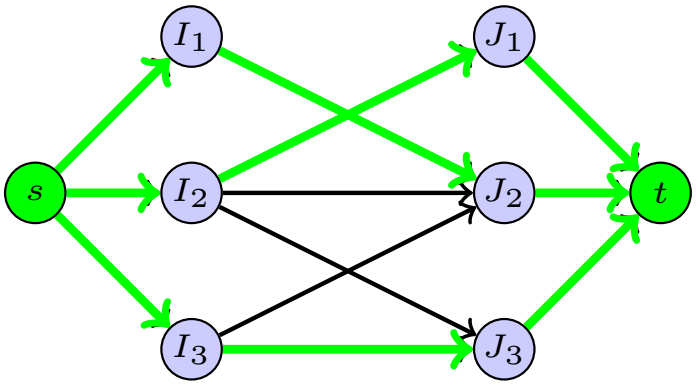
- Построим ориентированную сеть G' добавив источник s и сток t, и установив пропускные способности 1 для всех ориентированных ребер.



Стратегия: поток в сети



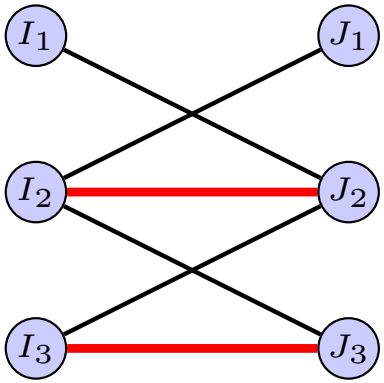
G с паросочетанием M



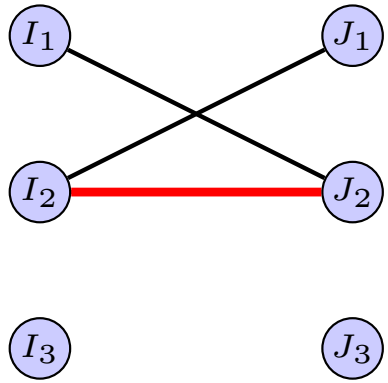
G' и поток f

- Очевидно, что любое паросочетание M соответствует потоку f величины $|M|$ (отметим, что все ребра имеют целочисленный поток, если используется алгоритм Форда-Фалкерсона). Наоборот, целочисленному потоку соответствует паросочетание M, так как ни один из узлов (за исключением s и t) не инцидентен с двумя ребрами, имеюшими поток 1.
- Т.о., наибольшее паросочетание найдено алгоритмом Форда-Фалкерсона за время $O(mn)$.

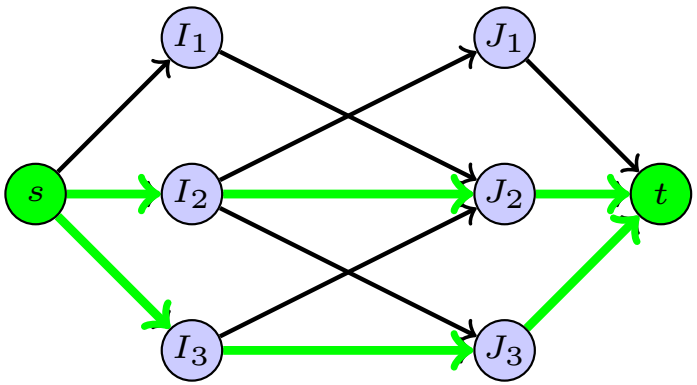
Эквивалентность



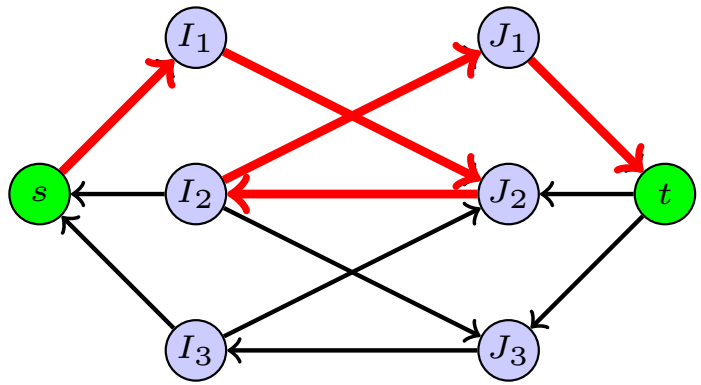
G и паросочетание M



Пополняющий путь P относительно M



G' и поток f



Пополняющий путь в остаточной сети G'_f

- Пополняющий путь $I_1 \to J_2 \to I_2 \to J_1$ относительно M соответствует увеличивающему пути в остаточной сети G'_f.

1.3 Алгоритм Хопктровта-Карпа: эффективный алгоритм для наибольшего паросочетания

Алгоритм Хопктровта-Карпа

- Идея: вместо нахождения произвольного пополняющего пути, Алгоритм Хопктровта-Карпа использует **кратчайший пополняющий путь**. Процесс использования таких путей можно разделить на **фазы**, т.е., на каждой фазе, кратчайшие пополняющие пути имеют равные длины и вершинно не пересекаются.
- Нахождение **наибольших кратчайших пополняющих путей** по существу тоже, что делает алгоритм Диница.

Алгоритм Хопктровта-Карпа [1973]

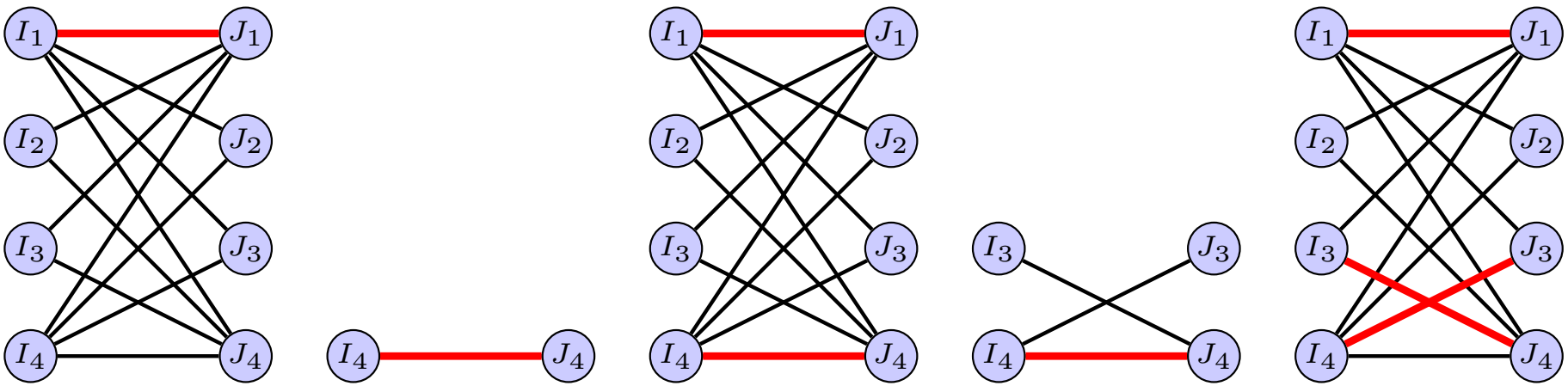
1: $\mathbf{M} = \{\}$; //исходное паросочетание;
2: **while** TRUE **do**
3: найти **максимальные кратчайшие пополняющие пути** $P = \{P_1, P_2, ..., P_k\}$ относительно M;
4: **if** ни одного пополняющего пути не найдено **then**
5: break;
6: **end if**
7: увеличить M используя P : $M = M \oplus P_1 \oplus P_2 \oplus ... \oplus P_k$;
8: **end while**
9: **return** $\mathbf{M}$;

- Трудоемкость: $O(m\sqrt{n})$.
- Важное замечание: вместо рассмотрения каждого пополняющего пути по отдельности, алгоритм рассматривает всю фазу целиком (см. **блокирующий поток** в Алгоритме Диница).

Свойство последовательных пополнений

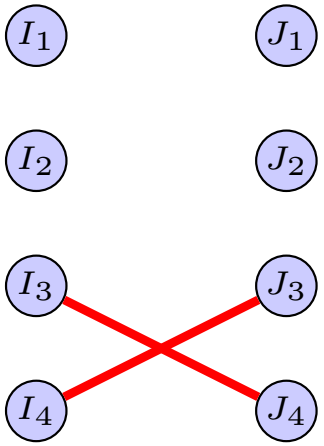
Theorem

Пусть M_0 — паросочетание, P_0 — кратчайший пополняющий путь относительно M_0, и P_1 — пополняющий путь относительно $M_1 = M_0 \oplus P_0$. Тогда $|P_1| \geq |P_0| + |P_0 \cap P_1|$.

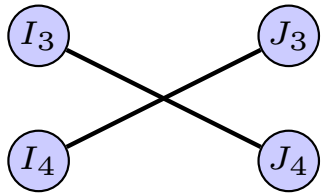


M_0 P_0 M_1 P_1 M_2

Например, $|P_0| = 1$, $|P_1| = 3$, и $|P_0 \cap P_1| = 1$.



$M_0 \oplus M_2$



$P_0 \oplus P_1$

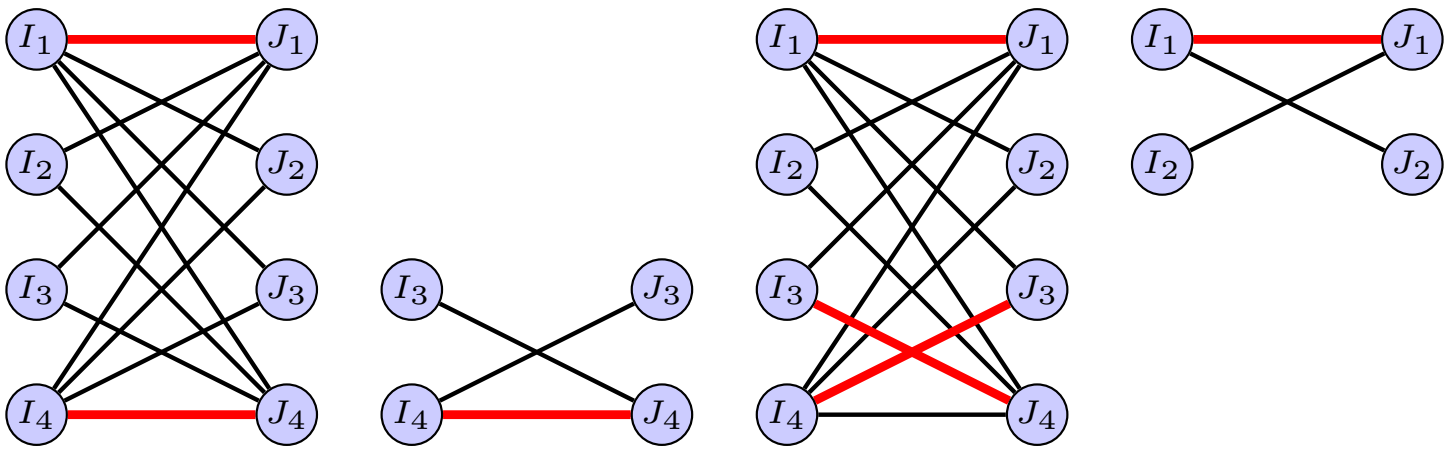
Доказательство.

- Рассмотрим $P_0 \oplus P_1$. Выполняются следующие свойства:
  - $P_0 \oplus P_1 = M_0 \oplus M_2$.
  - Поскольку $|M_2| = |M_0| + 2$, $M_0 \oplus M_2$ содержит по крайней мере 2 вершинно-различных пополняющих пути относительно M_0; обозначим их p и p'. Мы имеем $|p| \geq |P_0|$, и $|p'| \geq |P_0|$, так как P_0 является кратчайшим путем.
- Т.о., $|P_0 \oplus P_1| = |M_0 \oplus M_2| \geq |p| + |p'| \geq 2|P_0|$.
- Поскольку $|P_0 \oplus P_1| = |P_0| + |P_1| - |P_0 \cap P_1|$, получаем $|P_1| \geq |P_0| + |P_1 \cap P_0|$.

Свойство последовательных пополнений (продолжение)

Theorem

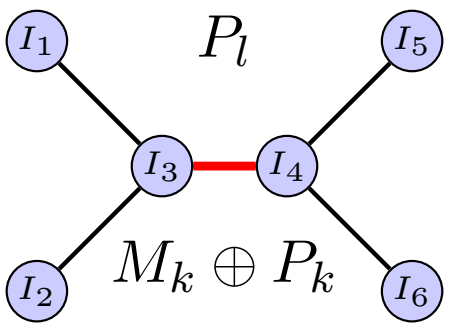
Если стартуем с паросочетания M_0, вычислим последовательность $\mathbf{M_0}, \mathbf{P_0}, \mathbf{M_1}, \mathbf{P_1}, \mathbf{M_2}, \mathbf{P_2}, ...$, *где P_i представляет* **кратчайший пополняющий путь** *относительно M_i, и $M_{i+1} = M_i \oplus P_i$. То имеем $|P_j| \geq |P_i|$ (для $j > i$). Если $|P_j| = |P_i|$, то P_j и P_i являются* **вершинно-различными**, *и оба являются пополняющими путями относительно M_i.*



M_0 P_0 M_1 P_1

Доказательство.

- От противного. Предположим, что $|P_j| = |P_i|$ (для $j > i$), но P_j и P_i не являются вершинно-различными.
- Тогда существуют k и l такие, что $i \leq k < l \leq j$, P_k и P_l не являются вершинно-различными, и для каждого m, $k < m < l$, P_m является вершинно отличным от P_k и P_l.
- Тогда P_l является пополняющим путем относительно $M_k \oplus P_k$.
- Так что $|P_l| \geq |P_k| + |P_k \cap P_l|$.
- Но $|P_l| = |P_k|$, поэтому $|P_l \cap P_k| = 0$, т.е., P_l и P_k не имеют общих ребер. Если P_l и P_k имеют общую вершину v, то они должны иметь общее ребро инцидентное с v, которое находится в $M_k \oplus P_k$. Получили противоречие.



Трудоемкость Алгоритма Hopcroft-Karp

Theorem

Пусть s — размер наибольшего паросочетания. **Число различных длин путей** $|P_0|, |P_1|, |P_2|, ..., |P_{s-1}|$ *не превосходит* $2\lfloor\sqrt{s}\rfloor + 2$.

Доказательство.

- Разделим шаги на две части:
  - Первая $r = \lfloor s - \sqrt{s}\rfloor$ шагов: после r шагов, паросочетание имеет размер $|M_r| = r$; однако, длиена пути увеличивается самое большее на $2\sqrt{s} + 1$, поскольку $|P_r| \leq 2\lfloor\frac{s-\sqrt{s}}{\sqrt{s}}\rfloor + 1 \leq 2\lfloor\sqrt{s}\rfloor + 1$. (Причина: для каждого r, имеем $|P_r| \leq 2\lfloor\frac{r}{s-r}\rfloor$, поскольку $M_s \oplus M_r$ содержит по меньшей мере $s - r$ вершинно-различных (и следовательно реберно-различных) пополняющих путей относительно M_r. Вмести эти пути содержат самое большее r ребер из M_r. Так что один из них должен иметь меньше чем $\lfloor\frac{r}{s-r}\rfloor$ ребер из M_r.)
  - Другие $\sqrt{s}$ шагов: на каждом шаге длина пути увеличивается самое большее на 1.

Трудоемкость Алгоритма Hopcroft-Karp

- Тот факт, что в последовательности длин путей не более $2\sqrt{s}+1$ различных чисел, означает, что множество пополняющих путей можно разделить не более чем на $2\sqrt{s}+1$ частей, каждая из которых содержит кратчайшие увеличивающие пути с одинаковыми длина. На каждой фазе алгоритма (шаги 3 – 7) мы работаем с соответствующей частью.
- Факт: пути в фазе вершинно-непересекающиеся, и они являются увеличивающими путями относительно паросочетания, с которого началась эта фаза.
- Таким образом, мы можем сосредоточиться на эффективной реализации **всей фазы**, т.е. нужно эффективно находить для фазы **максимальный набор увеличивающих кратчайших путей** .

Трудоемкость Алгоритма Hopcroft-Karp

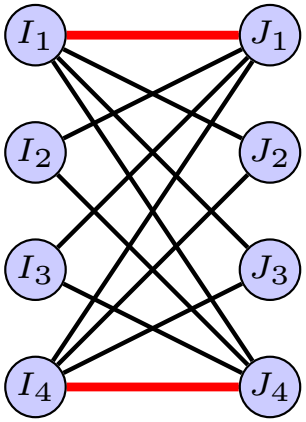
- Обратите внимание, что этот анализ работает для общих графов.
- Для двудольных графов **максимальный набор увеличивающих кратчайших путей**, образует **M-альтернирующий лес**, который можно найти с помощью методов BFS и DFS за время $O(m)$, это тоже, что и **многоуровневая сеть**, определенная в алгоритме Диница.
- Таким образом, общая трудоемкость составляет $O(m\sqrt{n})$ для двудольных графов, поскольку каждая фаза выполняется за время $O(m)$.

Алгоритм Хопкровта-Карпа: детальная версия

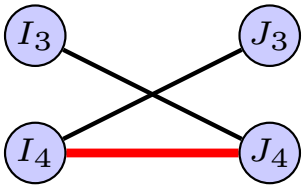
1: $\mathbf{M} = \{\}$; //исходное множество паросочетаний пусто;
2: **while** TRUE **do**
3: Построить **уровневую сеть**, выполнив BFS для графа G, где ребра в M ориентируются от доли 1 к доле 2, а ребра в $E - M$ ориентируются в обратном направлении. (Отметим, что пополняющие пути становятся ориентированными в G);
4: Найти **максимальные кратчайшие пополняющие пути** $P = \{P_1, P_2, ..., P_k\}$ относительно M, выполнив DFS по уровневой сети; (как только путь идентифицирован, он удаляется чтобы гарантировать непересечение путей по вершинам)
5: **if** ни один пополняющий путь не найден **then**
6: break;
7: **end if**
8: Увеличить M используя P, т.е., установив $M = M \oplus P_1 \oplus P_2 \oplus ... \oplus P_k$;
9: **end while**
10: **return** $\mathbf{M}$;

Трудоемкость: $O(m\sqrt{n})$.

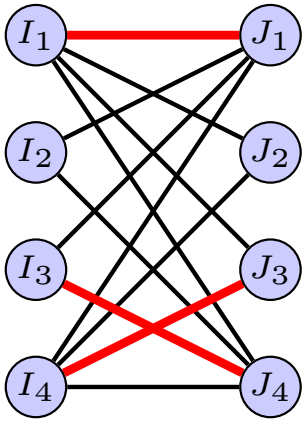
Пример: нахождения P_2 и P_3 на фазе 2



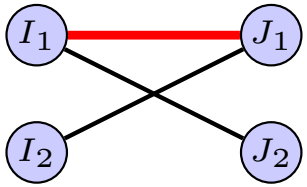
M_2



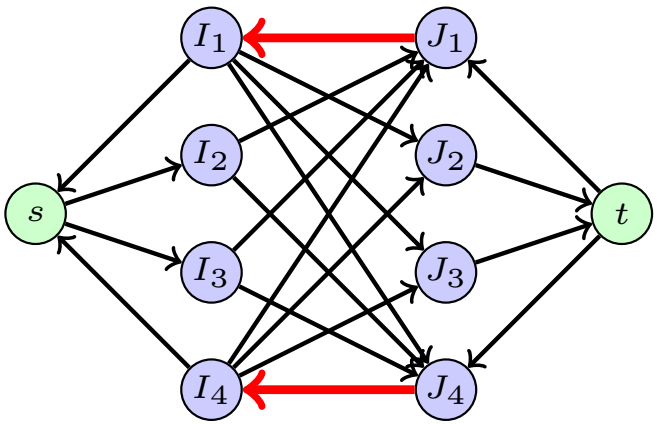
P_2



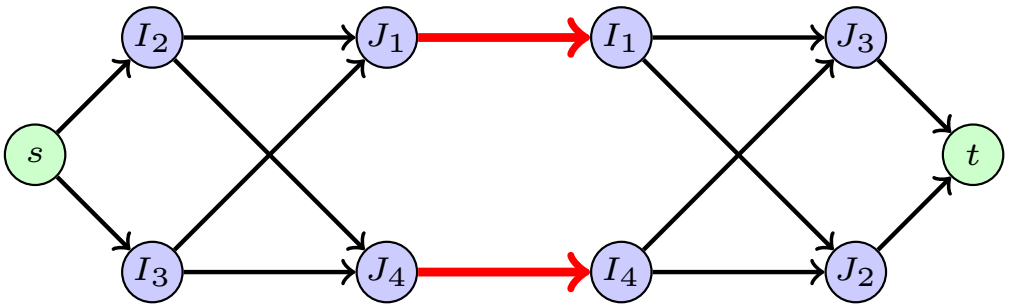
M_3



P_3



G

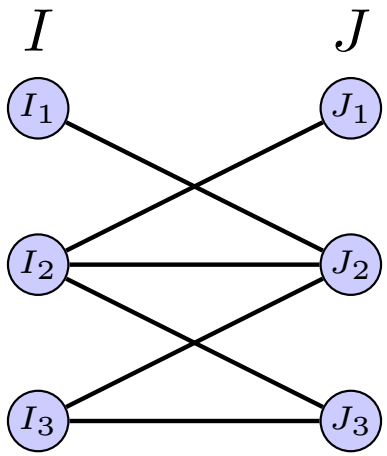


Уровневая сеть и кратчайшие $s-t$ пути

Отметим, что кратчайшие пополняющие пути относительно M_2 взаимно-однозначно соответствуют кратчайшим $s-t$ путям в G.

1.4 ЦЛП формулировка и двойственная задача

ЦЛП формулировка задачи о наибольшем паросочетании



$$
\begin{array}{lccccccccll}
\max & x_{12} & +x_{21} & +x_{22} & +x_{23} & +x_{31} & +x_{32} & +x_{33} & & & \\
s.t. & x_{12} & & & & & & & \leq 1 & \text{вершина } I_1 \\
 & & x_{21} & +x_{22} & +x_{23} & & & & \leq 1 & \text{вершина } I_2 \\
 & & & & & & x_{32} & +x_{33} & \leq 1 & \text{вершина } I_3 \\
 & & x_{21} & & & & & & \leq 1 & \text{вершина } J_1 \\
 & x_{12} & & +x_{22} & & & +x_{32} & & \leq 1 & \text{вершина } J_2 \\
 & & & & x_{23} & & & +x_{33} & \leq 1 & \text{вершина } J_3 \\
 & & & & & & & x_{ij} & = 0/1 &
\end{array}
$$

Свойство тотальной унимодулярности

- Отметим, что матрица инцидентности двудольного графа является вполне унимодулярной.
- Т.о., можно ограничения по целочисленности заменить на $1 \geq x_{ij} \geq 0$, мы получим задачу ЛП. Ее двойственная задача это Задача о вершинном покрытии. Здесь, u_i обозначает двойственную переменную для работника I_i, а v_i обозначает переменную для работы J_i.

$$
\begin{array}{lccccccccc}
\min & u_1 & +u_2 & +u_3 & +v_1 & +v_2 & +v_3 & & & \\
s.t. & u_1 & & & & +v_2 & & \geq 1 & \text{ребро } (I_1, J_2) \\
 & & u_2 & & +v_1 & & & \geq 1 & \text{ребро } (I_2, J_1) \\
 & & u_2 & & & +v_2 & & \geq 1 & \text{ребро } (I_2, J_2) \\
 & & u_2 & & & & +v_3 & \geq 1 & \text{ребро } (I_2, J_3) \\
 & & & u_3 & & +v_2 & & \geq 1 & \text{ребро } (I_2, J_2) \\
 & & u_2 & & & & +v_3 & \geq 1 & \text{ребро } (I_2, J_3) \\
 & u_1, & u_2, & u_3, & v_1, & v_2, & v_3 & \geq 0 & \\
\end{array}
$$

Двойственная задача: Наименьшее вершинное покрытие

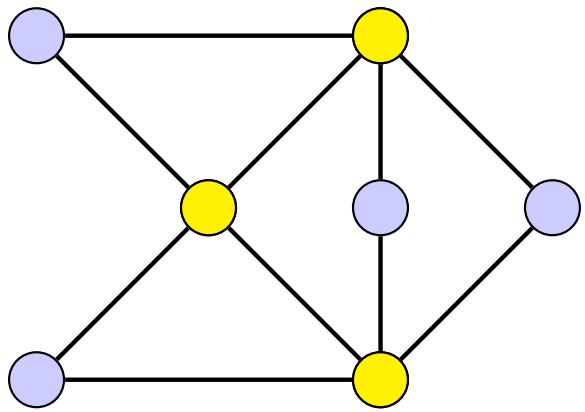
- Отметим, что матрица инцидентности двудольного графа является вполне унимодулярной. Следовательно задача ЛП имеет целочисленное решение.
- Двойственная задача — это, по сути, задача о наименьшем вершинном покрытии.

$$
\begin{array}{lccccccccl}
\min & u_1 & +u_2 & +u_3 & +v_1 & +v_2 & +v_3 & & & \\
s.t. & u_1 & & & & +v_2 & & \geq 1 & \text{ребро } (I_1, J_2) \\
 & & u_2 & & +v_1 & & & \geq 1 & \text{ребро } (I_2, J_1) \\
 & & u_2 & & & +v_2 & & \geq 1 & \text{ребро } (I_2, J_2) \\
 & & u_2 & & & & +v_3 & \geq 1 & \text{ребро } (I_2, J_3) \\
 & & & u_3 & & +v_2 & & \geq 1 & \text{ребро } (I_2, J_2) \\
 & & u_2 & & & & +v_3 & \geq 1 & \text{ребро } (I_2, J_3) \\
 & u_1, & u_2, & u_3, & v_1, & v_2, & v_3 & = 0/1 & \\
\end{array}
$$

1.5 Теорема Кёнига о наибольшем паросочетании и наименьшем вершинном покрытии

Задача о наименьшем вершинном покрытии

Вход: граф $G = (V, E)$,
Выход: Найти наименьшее множество вершин $S \subseteq V$, такое, что каждое ребро имеет по меньшей мере одну инцидентную вершину в S.



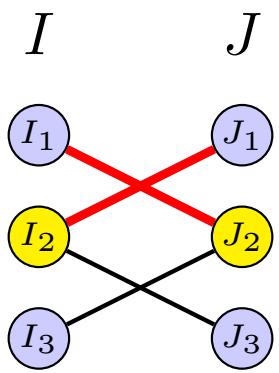
- В этом примере, три вершины покрывают все ребра.

Теорема Кёнига о наибольшем паросочетании и наименьшем вершинном покрытии [1931]

- Теорема Кёнига в теоретико-графовой формулировке:

Theorem

Для двудольного графа G, размер наибольшего паросочетания совпадает с размером наименьшего вершинного покрытия.



Теорема Кёнига о наибольшем паросочетании и наименьшем вершинном покрытии [1931]

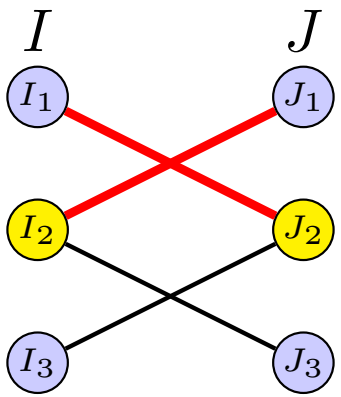
- Теорема Кёнига в матричной форме:

Theorem

Для любой 0,1-матрицы наименьшее количество строк и столбцов, которые содержат все единицы равно наибольшему количеству единиц, выбранных так, что ни в одной стоке и ни в одном столбце не выбрано более одной единицы.

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

Наибольшее паросочетание и наименьшее вершинное покрытие



- Обратите внимание, что задача MinVertexCover для общих графов является NP-трудной. Однако, в классе двудольных графов задача MinVertexCover решается эффективно.
- Фактически, эта теорема представляет собой пример двойственности в линейном программировании
- Во-первых, очевидно, что любое вершиное покрытие дает верхнюю границу для максимального паросочетания (Причина: ребра в паросочетании не имеют общих узлов; таким образом, чтобы покрыть эти ребра, нужно выбрать хотя бы один из концевых узлов каждого ребра.)

Теорема о вершинном покрытии

Теорема Кёнига

В двудольном графе размер минимального вершинного покрытия равен размеру максимального паросочетания.

Теорема о вершинном покрытии

Теорема Кёнига

В двудольном графе размер минимального вершинного покрытия равен размеру максимального паросочетания.

Доказательство

- Пусть в G построено максимальное паросочетание.

Теорема о вершинном покрытии

Теорема Кёнига

В двудольном графе размер минимального вершинного покрытия равен размеру максимального паросочетания.

Доказательство

- Пусть в G построено максимальное паросочетание.
- Ориентируем ребра паросочетания, чтобы они шли из правой доли в левую, а ребра не из паросочетания — так, чтобы они шли из левой доли в правую.

Теорема о вершинном покрытии

Теорема Кёнига

В двудольном графе размер минимального вершинного покрытия равен размеру максимального паросочетания.

Доказательство

- Пусть в G построено максимальное паросочетание.
- Ориентируем ребра паросочетания, чтобы они шли из правой доли в левую, а ребра не из паросочетания — так, чтобы они шли из левой доли в правую.
- Запустим обход в глубину из всех не насыщенных паросочетанием вершин левой доли.

Теорема о вершинном покрытии

Теорема Кёнига

В двудольном графе размер минимального вершинного покрытия равен размеру максимального паросочетания.

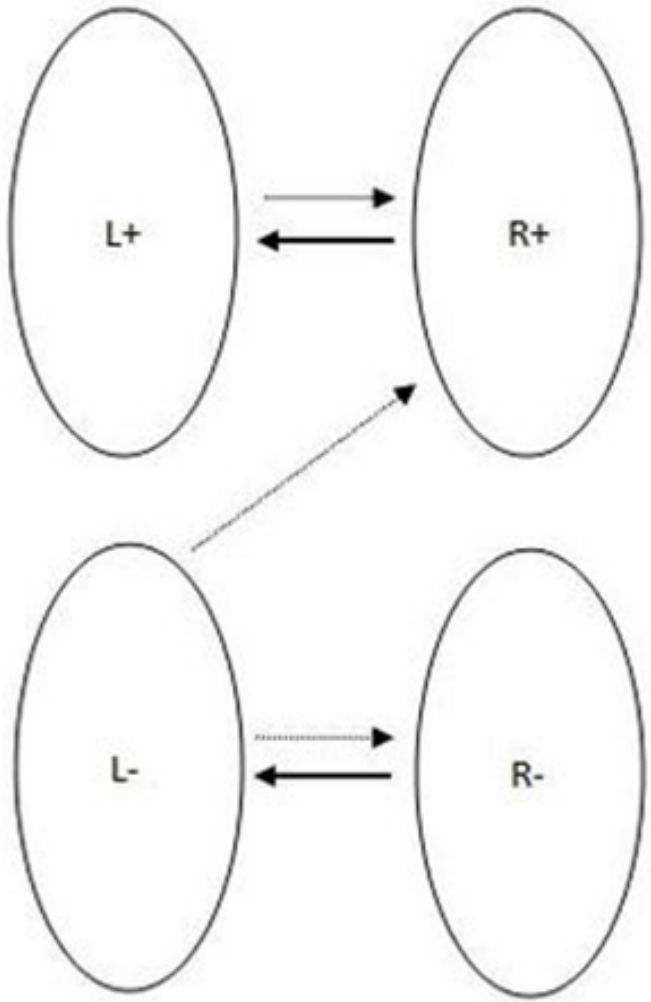
Доказательство

- Пусть в G построено максимальное паросочетание.
- Ориентируем ребра паросочетания, чтобы они шли из правой доли в левую, а ребра не из паросочетания — так, чтобы они шли из левой доли в правую.
- Запустим обход в глубину из всех не насыщенных паросочетанием вершин левой доли.
- Разобьем вершины каждой доли графа на два множества: те, которые были посещены в процессе обхода, и те, которые не были посещены в процессе обхода.

Доказательство (продолжение)

Доказательство

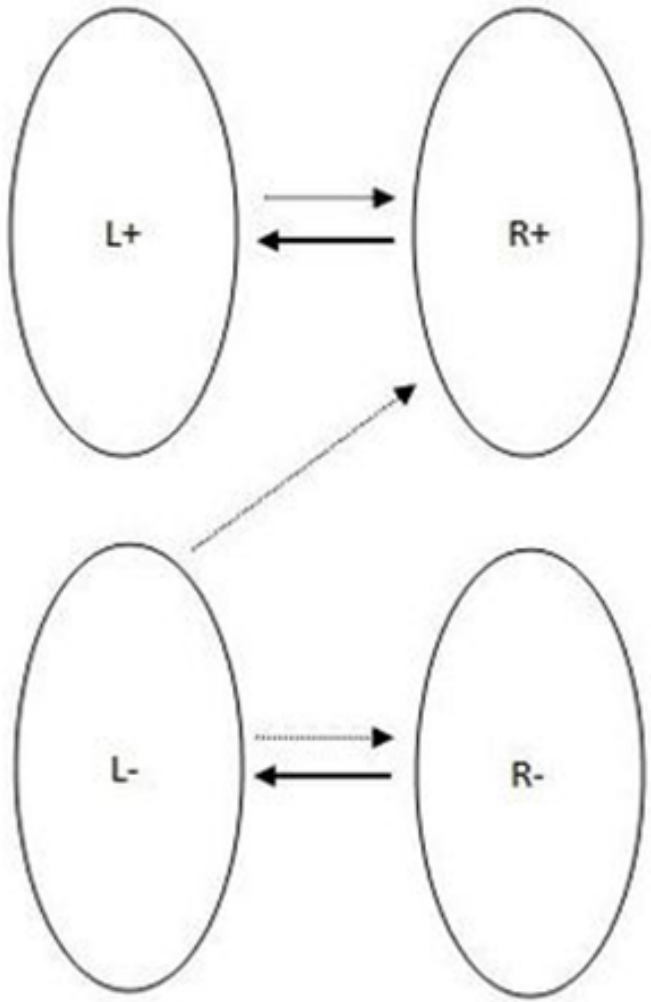
- Тогда $L = L^+ \cup L^-$, $R = R^+ \cup R^-$, где L, R — правая и левая доли соответственно, L^+, R^+ — вершины правой и левой доли, посещенные обходом, L^-, R^- — не посещенные обходом вершины.



Доказательство (продолжение)

Доказательство

- Тогда $L = L^+ \cup L^-$, $R = R^+ \cup R^-$, где L, R — правая и левая доли соответственно, L^+, R^+ — вершины правой и левой доли, посещенные обходом, L^-, R^- — не посещенные обходом вершины.
- Тогда в G могут быть следующие ребра:
  - Из вершин L^+ в вершины R^+ и из вершин R^+ в вершины L^+;
  - из вершин L^- в вершины R^- и из вершин R^- в вершины L^-;
  - из вершин L^- в вершины R^+.



Доказательство (продолжение)

Доказательство

- Очевидно, что ребер из L^+ в R^- и из R^+ в L^- быть не может.

Доказательство (продолжение)

Доказательство

- Очевидно, что ребер из L^+ в R^- и из R^+ в L^- быть не может.
- Ребер из R^- в L^+ быть не может, т.к. если такое ребро uv существует, то оно — ребро паросочетания.

Доказательство (продолжение)

Доказательство

- Очевидно, что ребер из L^+ в R^- и из R^+ в L^- быть не может.
- Ребер из R^- в L^+ быть не может, т.к. если такое ребро uv существует, то оно — ребро паросочетания.
- Тогда вершина v насыщена паросочетанием.

Доказательство (продолжение)

Доказательство

- Очевидно, что ребер из L^+ в R^- и из R^+ в L^- быть не может.
- Ребер из R^- в L^+ быть не может, т.к. если такое ребро uv существует, то оно — ребро паросочетания.
- Тогда вершина v насыщена паросочетанием.
- Но т.к. $v \in L^+$, то в нее можно дойти из какой-то ненасыщенной вершины левой доли.

Доказательство (продолжение)

Доказательство

- Очевидно, что ребер из L^+ в R^- и из R^+ в L^- быть не может.
- Ребер из R^- в L^+ быть не может, т.к. если такое ребро uv существует, то оно — ребро паросочетания.
- Тогда вершина v насыщена паросочетанием.
- Но т.к. $v \in L^+$, то в нее можно дойти из какой-то ненасыщенной вершины левой доли.
- Значит, существует ребро wv $w \in R^+$. Но тогда v инцидентны два ребра из паросочетания. Противоречие.

Доказательство (продолжение)

Доказательство

- Очевидно, что ребер из L^+ в R^- и из R^+ в L^- быть не может.
- Ребер из R^- в L^+ быть не может, т.к. если такое ребро uv существует, то оно — ребро паросочетания.
- Тогда вершина v насыщена паросочетанием.
- Но т.к. $v \in L^+$, то в нее можно дойти из какой-то ненасыщенной вершины левой доли.
- Значит, существует ребро wv $w \in R^+$. Но тогда v инцидентны два ребра из паросочетания. Противоречие.
- Заметим, что минимальным вершинным покрытием G является либо L, либо R, либо $L^- \cup R^+$.

Доказательство (продолжение)

Доказательство

- В R^+ не насыщенных паросочетанием вершин быть не может, т.к. иначе в G существует дополняющий путь, что противоречит максимальности выбранного паросочетания.

Доказательство (продолжение)

Доказательство

- В R^+ не насыщенных паросочетанием вершин быть не может, т.к. иначе в G существует дополняющий путь, что противоречит максимальности выбранного паросочетания.
- В L^- свободных вершин быть не может, т.к. все они должны находиться в L^+.

Доказательство (продолжение)

Доказательство

- В R^+ не насыщенных паросочетанием вершин быть не может, т.к. иначе в G существует дополняющий путь, что противоречит максимальности выбранного паросочетания.
- В L^- свободных вершин быть не может, т.к. все они должны находиться в L^+.
- Тогда т.к. ребер из паросочетания между R^+ и L^- нет, то каждому ребру максимального паросочетания инцидентна ровно одна вершина из $L^- \cup R^+$.

Доказательство (продолжение)

Доказательство

- В R^+ не насыщенных паросочетанием вершин быть не может, т.к. иначе в G существует дополняющий путь, что противоречит максимальности выбранного паросочетания.
- В L^- свободных вершин быть не может, т.к. все они должны находиться в L^+.
- Тогда т.к. ребер из паросочетания между R^+ и L^- нет, то каждому ребру максимального паросочетания инцидентна ровно одна вершина из $L^- \cup R^+$.
- Тогда $|L^- \cup R^+|$ равна мощности максимального паросочетания. Множество вершин $L^- \cup R^+$ является минимальным вершинным покрытием. Значит мощность максимального паросочетания равна мощности минимального вершинного покрытия.

Алгоритм построения минимального вершинного покрытия двудольного графа

1. Построить максимальное паросочетание.

Алгоритм построения минимального вершинного покрытия двудольного графа

1. Построить максимальное паросочетание.
2. Ориентировать ребра:
   - Из паросочетания — из правой доли в левую.
   - Не из паросочетания — из левой доли в правую.

Алгоритм построения минимального вершинного покрытия двудольного графа

1. Построить максимальное паросочетание.
2. Ориентировать ребра:
   - Из паросочетания — из правой доли в левую.
   - Не из паросочетания — из левой доли в правую.
3. Запустить обход в глубину из всех свободных вершин левой доли, построить множества L^+, L^-, R^+, R^-.

Алгоритм построения минимального вершинного покрытия двудольного графа

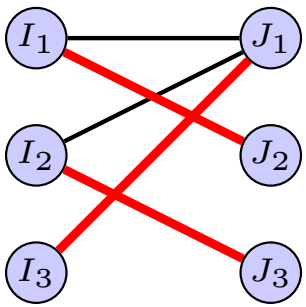
1. Построить максимальное паросочетание.
2. Ориентировать ребра:
   - Из паросочетания — из правой доли в левую.
   - Не из паросочетания — из левой доли в правую.
3. Запустить обход в глубину из всех свободных вершин левой доли, построить множества L^+, L^-, R^+, R^-.
4. В качестве результата взять $L^- \cup R^+$.

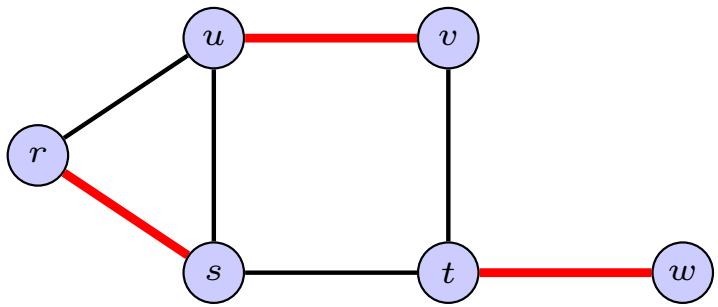
1.6 Совершенное паросочетание: теорема Холла и теорема Татта

Совершенное паросочетание

Definition (Совершенное паросочетание)

В совершенном паросочетании каждая вершина инцидентна ребру паросочетания.





- Теорема Холла устанавливает достаточное и необходимое условие существования совершенного паросочетания в двудольных графах, а теорема Татта устанавливает это условие для общих графов.

Теорема Холла [1935]

Theorem (Холла)

Двудольный граф $G = (I \cup J, E)$ имеет совершенное паросочетание тогда и только тогда, когда для любого $S \subseteq I$, $|N(S)| \geq |S|$, где $N(S)$ обозначает окружение вершин S.

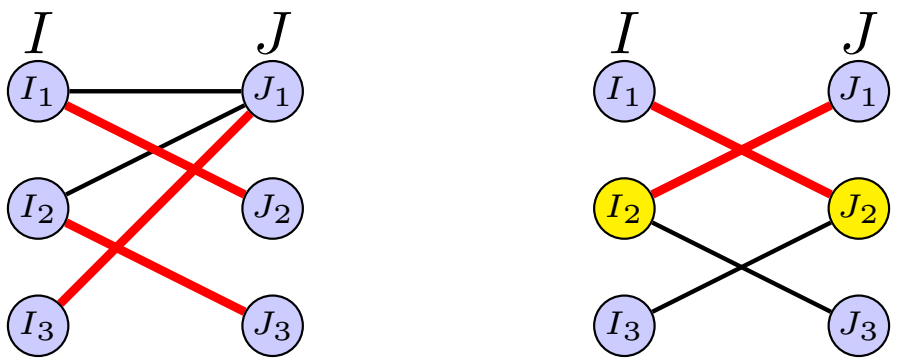
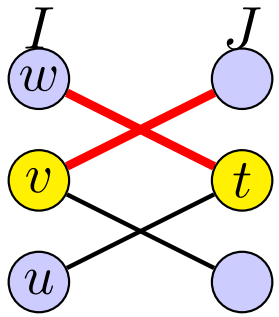


Рис.: Граф с совершенным паросочетанием (слева) и граф, не имеющий совершенного паросочетания (справа)



Доказательство.

- Здесь мы доказываем существование совершенного паросочетания только в том случае, если для любого $S \subseteq I$, $|N(S)| \geq |S|$. Предположим от противного, что максимальное паросочетание M не является совершенным, т.е. все еще существует свободная вершина $u \in I$.
- Пусть F обозначает чередующееся дерево с корнем в u, $S = I \cap F$ и $T = J \cap F$, например, $S = \{u, w\}$, и $T = \{t\}$.
- Поскольку M является максимальным паросочетанием, каждая работа из T сопоставляется с человеком из $S - \{u\}$, и наоборот. Поэтому, $|S| - 1 = |T|$.

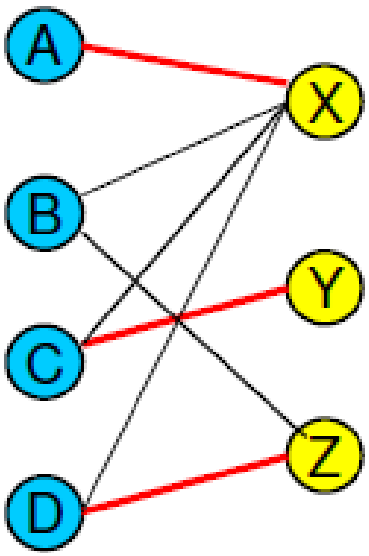
Доказательство.

- Обратите внимание, что между $S = I \cap F$ и $J - F$ нет ребра (так как эти ребра не могут быть покрыты $C = (IF) \cup (J \cap F)$, поэтому C не является наименьшим вершинным покрытием).
- Следовательно $N(S) \subseteq T$, т.е., $|N(S)| \leq |T| = |S| - 1$. Противоречие.

Полусовершенное паросочетание в двудольном графе

Definition (Полусовершенное паросочетание в двудольном графе)

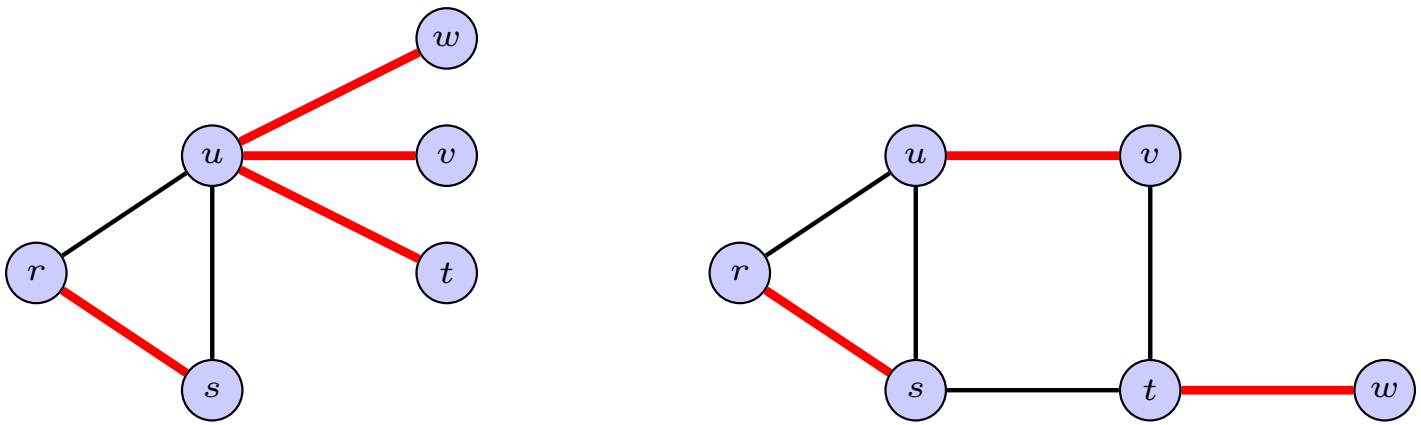
В полусовершенном паросочетании в двудольном графе каждая вершина меньшей доли инцидентна ребру паросочетания.



Полусовершенное паросочетание является максимальным паросочетанием в двудольном графе.

1.7 MaxMatching и MinEdgeCover для общих графов

ВХОД: Граф $G = (V, E)$
ВЫХОД: Минимальное подмножество ребер $S \in E$ покрывающее все вершины, т.е. каждая вершина $v \in V$ инцидентентна хотя бы одному ребру из S.

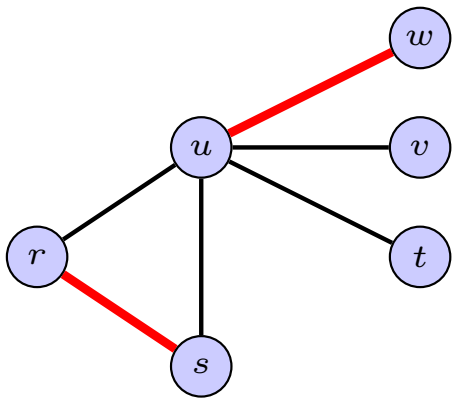


- MinVertexCover является NP-трудной, а MinEdgeCover принадлежит P.

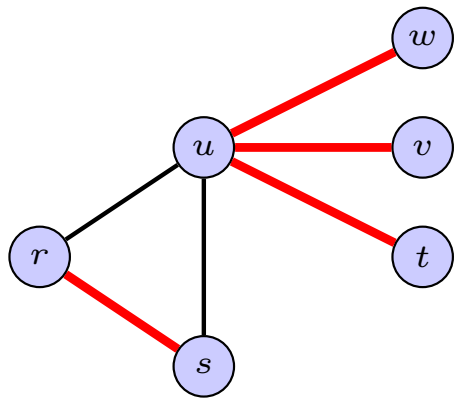
Решение MinEdgeCover

Theorem

Пусть $G = (V, E)$ — граф без изолированных вершин. Его наибольшее паросочетание M имеет размер $|M| = s$ тогда и только тогда, когда его наименьшее реберное покрытие C имеет размер $|C| = n - s$.

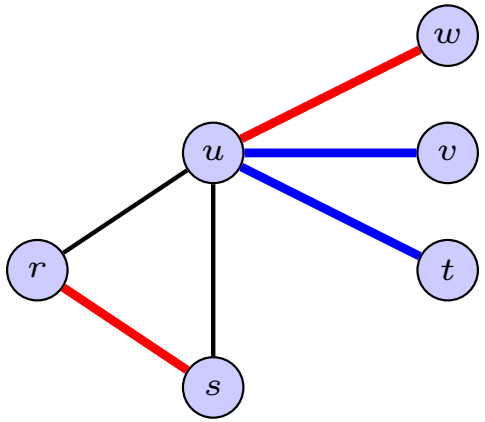


Max Matching $|M| = 2$



Min Edge Cover $|C| = 4$

- MinEdgeCover стремится покрыть все вершины, используя как можно меньше ребер. Напротив, MaxMatching стремится охватить как можно больше вершин с помощью **независимых ребер**.



Доказательство.

- Давайте построим реберное покрытие, увеличив наибольшее паросочетание.
- Рассмотрим максимальное паросочетание M с $|M| = s$. Сначала мы выбираем все ребра в M, которые покрывают $2s$ узлов.
- Затем для каждой из $n - 2s$ свободной вершины мы произвольно выбираем одно из инцидентных ей ребер.

Доказательство.

- Обратите внимание, что эти выбранные ребра различны, поскольку нет ребра, соединяющего любые две свободные вершины, скажем, v и t (в противном случае добавление ребра (v, t) приведет к большему паросочетанию).
- Таким образом, мы получаем всего $n - s$ ребер, которые покрывают все вершины.
- С другой стороны, давайте построим паросочетание посредством удаления ребер из наименьшего реберного покрытия C с $|C| = t$.
- Изначально мы устанавливаем $M = C$. Для каждой вершины $v \in V$ пусть d обозначает количество инцидентных ей ребер из M, из M удаляется $d - 1$ ребер.

Доказательство.

- Обратите внимание, что удаление $d-1$ ребер создает $d-1$ свободных вершин (относительно M). (Причина: если удаление ребра не создает свободных вершин, то C не является наименьшим реберным покрытием.)
- Пусть y — суммарное количество ребер, удаленных из C. Окончательно, M содержит $t-y$ ребер.
- M является паросочетанием, поскольку оно содержит не более 1 инцидентного ребра для каждой вершины.
- Кроме того, $|M| = t - y = n - t$. (Причина: Вначале M покрывает все вершины, и удаление y ребер создает y свободных вершин, поэтом будет $2(t-y)$ сочетаемых вершин. Т.о., получаем $n = y + 2(t-y) = 2t - y$.)

Алгоритм построения максимального паросочетания в произвольном графе

До этого были ягодки...

Определение

- Цветком относительно паросочетания M (blossom w.r.t. M) называется цикл нечётной длины с r рёбрами M и $r+1$ ребром не из M.

До этого были ягодки...

Определение

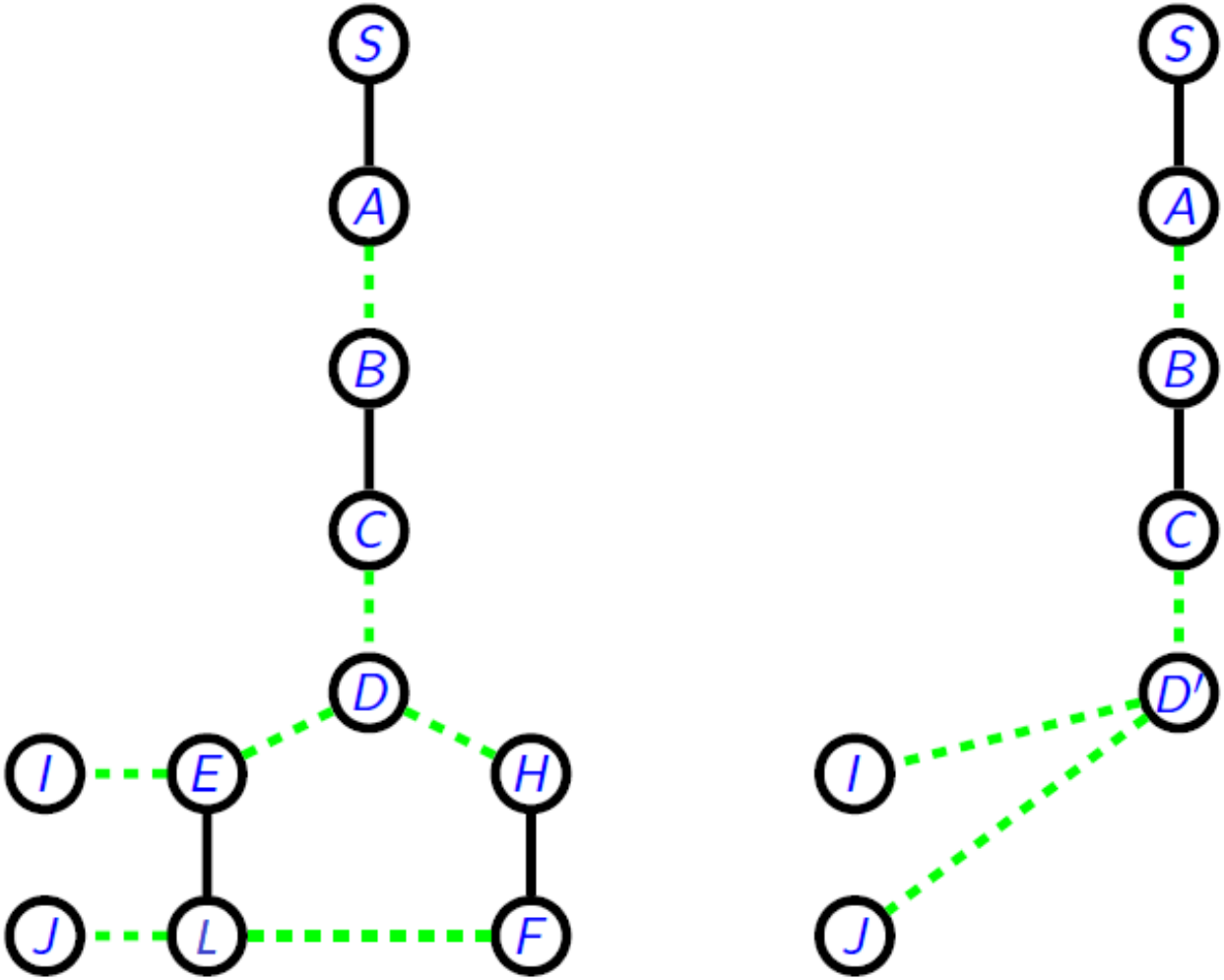
- Цветком относительно паросочетания M (blossom w.r.t. M) называется цикл нечётной длины с r рёбрами M и $r+1$ ребром не из M.
- Основанием цветка (blossom base) называется вершина цветка, в которой встречаются два ребра не из сочетания.

До этого были ягодки...

Определение

- Цветком относительно паросочетания M (blossom w.r.t. M) называется цикл нечётной длины с r рёбрами M и $r+1$ ребром не из M.
- Основанием цветка (blossom base) называется вершина цветка, в которой встречаются два ребра не из сочетания.
- Стянутым цветком (shrunken blossom) называется вершина, получаемая в результате стягивания вершин цветка. В полученном графе, эта вершина соответствует цветку в исходном графе.

Пример



Цветок (D, H, F, L, E) стягивается в вершину D'

Стягивание цветков

Теорема

Пусть M — паросочетание графа G, B — цветок в G. Предположим, что основание цветка не принадлежит M. Пусть граф G' и паросочетание M' получаются из G и M стягиванием цветка B. Тогда M' является максимальным в G' тогда и только тогда, когда M максимально в G.

Стягивание цветков

Теорема

Пусть M — паросочетание графа G, B — цветок в G. Предположим, что основание цветка не принадлежит M. Пусть граф G' и паросочетание M' получаются из G и M стягиванием цветка B. Тогда M' является максимальным в G' тогда и только тогда, когда M максимально в G.

Доказательство

- Допустим сперва, что M' не максимально в G'. Из этого следует, что в G' есть дополняющий путь P' (относительно M').

Стягивание цветков

Теорема

Пусть M — паросочетание графа G, B — цветок в G. Предположим, что основание цветка не принадлежит M. Пусть граф G' и паросочетание M' получаются из G и M стягиванием цветка B. Тогда M' является максимальным в G' тогда и только тогда, когда M максимально в G.

Доказательство

- Допустим сперва, что M' не максимально в G'. Из этого следует, что в G' есть дополняющий путь P' (относительно M').
- Если P' не содержит стянутого цветка, то он является и дополняющим путём относительно M в G.

Стягивание цветков

Теорема

Пусть M — паросочетание графа G, B — цветок в G. Предположим, что основание цветка не принадлежит M. Пусть граф G' и паросочетание M' получаются из G и M стягиванием цветка B. Тогда M' является максимальным в G' тогда и только тогда, когда M максимально в G.

Доказательство

- Допустим сперва, что M' не максимально в G'. Из этого следует, что в G' есть дополняющий путь P' (относительно M').
- Если P' не содержит стянутого цветка, то он является и дополняющим путём относительно M в G.
- Допустим, что P' содержит стянутый цветок B.

Стягивание цветков

Теорема

Пусть M — паросочетание графа G, B — цветок в G. Предположим, что основание цветка не принадлежит M. Пусть граф G' и паросочетание M' получаются из G и M стягиванием цветка B. Тогда M' является максимальным в G' тогда и только тогда, когда M максимально в G.

Доказательство

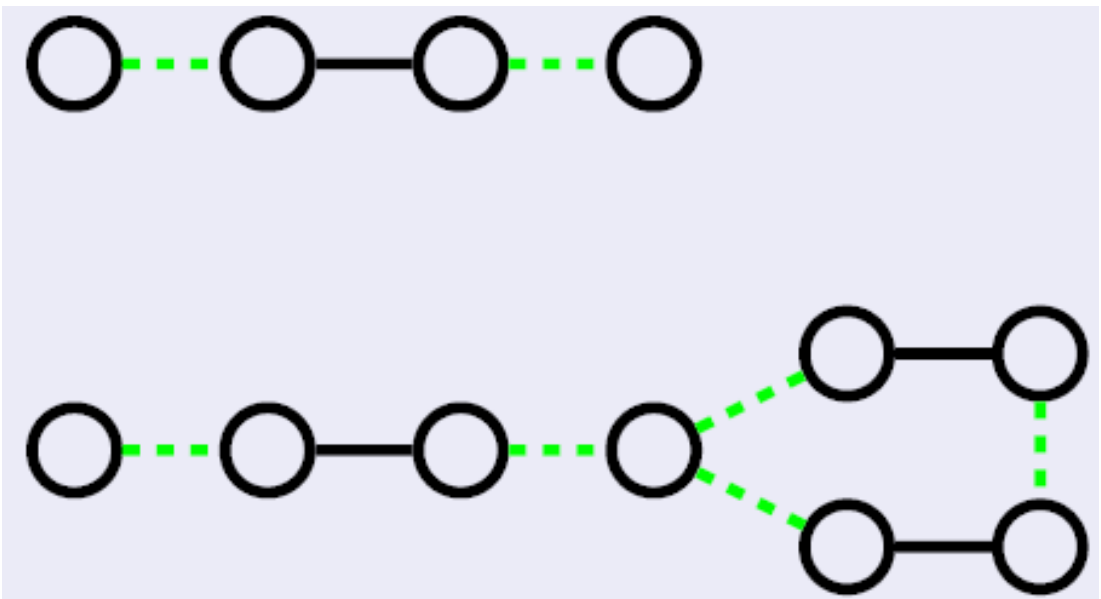
- Допустим сперва, что M' не максимально в G'. Из этого следует, что в G' есть дополняющий путь P' (относительно M').
- Если P' не содержит стянутого цветка, то он является и дополняющим путём относительно M в G.
- Допустим, что P' содержит стянутый цветок B.
- Поскольку стянутый цветок B не пересекается с паросочетанием M', он является конечной вершиной пути P'.

Раскрытие цветка

Доказательство

- Легко видеть, что при раскрытии цветка путь P' можно дополнить до дополняющего в G.

Пример



Продолжение доказательства

Доказательство

- Допустим теперь, что M не является максимальным в G.

Продолжение доказательства

Доказательство

- Допустим теперь, что M не является максимальным в G.
- Значит, в G есть дополняющий путь P относительно M.

Продолжение доказательства

Доказательство

- Допустим теперь, что M не является максимальным в G.
- Значит, в G есть дополняющий путь P относительно M.
- Считаем, что P пересекает цветок B (иначе всё и так ясно).

Продолжение доказательства

Доказательство

- Допустим теперь, что M не является максимальным в G.
- Значит, в G есть дополняющий путь P относительно M.
- Считаем, что P пересекает цветок B (иначе всё и так ясно).
- Поскольку цветок содержит ровно одну вершину не из паросочетания, хотя бы один из концов P (назовём его w) не лежит в цветке.

Продолжение доказательства

Доказательство

- Допустим теперь, что M не является максимальным в G.
- Значит, в G есть дополняющий путь P относительно M.
- Считаем, что P пересекает цветок B (иначе всё и так ясно).
- Поскольку цветок содержит ровно одну вершину не из паросочетания, хотя бы один из концов P (назовём его w) не лежит в цветке.
- Тогда путь от w до первого пересечения с цветком является дополняющим в G'.

Общие идеи алгоритма

- Мы модифицируем поиск в ширину так, чтобы он находил цветки. Каждый обнаруженный цветок стягивается в вершину, после чего поиск продолжается.

Общие идеи алгоритма

- Мы модифицируем поиск в ширину так, чтобы он находил цветки. Каждый обнаруженный цветок стягивается в вершину, после чего поиск продолжается.
- Ясно, что дополняющий путь в стянутом графе легко развернуть в дополняющий путь в исходном графе.

Общие идеи алгоритма

- Мы модифицируем поиск в ширину так, чтобы он находил цветки. Каждый обнаруженный цветок стягивается в вершину, после чего поиск продолжается.
- Ясно, что дополняющий путь в стянутом графе легко развернуть в дополняющий путь в исходном графе.
- Обозначим через U множество вершин не из паросочетания M.

Общие идеи алгоритма

- Мы модифицируем поиск в ширину так, чтобы он находил цветки. Каждый обнаруженный цветок стягивается в вершину, после чего поиск продолжается.
- Ясно, что дополняющий путь в стянутом графе легко развернуть в дополняющий путь в исходном графе.
- Обозначим через U множество вершин не из паросочетания M.
- Будем строить лес F (последовательно добавляя рёбра из паросочетания и не из паросочетания), содержащий по компоненте на каждую вершину U.

Общие идеи алгоритма

- Мы модифицируем поиск в ширину так, чтобы он находил цветки. Каждый обнаруженный цветок стягивается в вершину, после чего поиск продолжается.
- Ясно, что дополняющий путь в стянутом графе легко развернуть в дополняющий путь в исходном графе.
- Обозначим через U множество вершин не из паросочетания M.
- Будем строить лес F (последовательно добавляя рёбра из паросочетания и не из паросочетания), содержащий по компоненте на каждую вершину U.
- Ребра паросочетания, таким образом, будут на чётном расстоянии от вершин U.

Общие идеи алгоритма

- Мы модифицируем поиск в ширину так, чтобы он находил цветки. Каждый обнаруженный цветок стягивается в вершину, после чего поиск продолжается.
- Ясно, что дополняющий путь в стянутом графе легко развернуть в дополняющий путь в исходном графе.
- Обозначим через U множество вершин не из паросочетания M.
- Будем строить лес F (последовательно добавляя рёбра из паросочетания и не из паросочетания), содержащий по компоненте на каждую вершину U.
- Ребра паросочетания, таким образом, будут на чётном расстоянии от вершин U.
- Вершины, находящиеся на чётном расстоянии, будут иметь степень 2 (одно ребро из паросочетания, второе — нет).

Общие идеи алгоритма

- Мы модифицируем поиск в ширину так, чтобы он находил цветки. Каждый обнаруженный цветок стягивается в вершину, после чего поиск продолжается.
- Ясно, что дополняющий путь в стянутом графе легко развернуть в дополняющий путь в исходном графе.
- Обозначим через U множество вершин не из паросочетания M.
- Будем строить лес F (последовательно добавляя рёбра из паросочетания и не из паросочетания), содержащий по компоненте на каждую вершину U.
- Ребра паросочетания, таким образом, будут на чётном расстоянии от вершин U.
- Вершины, находящиеся на чётном расстоянии, будут иметь степень 2 (одно ребро из паросочетания, второе — нет).
- Назовём такие вершины внутренними, оставшиеся — внешними.

Описание алгоритма

Рассмотрим множество соседей внешних вершин. Возможны четыре случая:

Описание алгоритма

Рассмотрим множество соседей внешних вершин. Возможны четыре случая:

1. **Внешняя вершина x соединена с вершиной y не из F.** Добавить рёбра (x, y) и $(y, z) \in M$ в F.

Описание алгоритма

Рассмотрим множество соседей внешних вершин. Возможны четыре случая:

1. **Внешняя вершина x соединена с вершиной y не из F.** Добавить рёбра (x, y) и $(y, z) \in M$ в F.
2. **Соединены две внешние вершины из разных компонент.** Корни этих компонет соединены дополняющим путём.

Описание алгоритма

Рассмотрим множество соседей внешних вершин. Возможны четыре случая:

1. **Внешняя вершина x соединена с вершиной y не из F.** Добавить рёбра (x, y) и $(y, z) \in M$ в F.
2. **Соединены две внешние вершины из разных компонент.** Корни этих компонет соединены дополняющим путём.
3. **Соединены две внешние вершины x и y из одной компоненты.** Рассмотрим цикл C компоненты, содержащий вершины x, y, а также путь, ведущий из корня компоненты в этот цикл. Изменить рёбра сочетания вдоль этого пути (размер сочетания при этом не изменится) и стянуть найденный цветок.

Описание алгоритма

Рассмотрим множество соседей внешних вершин. Возможны четыре случая:

1. **Внешняя вершина x соединена с вершиной y не из F.** Добавить рёбра (x, y) и $(y, z) \in M$ в F.
2. **Соединены две внешние вершины из разных компонент.** Корни этих компонет соединены дополняющим путём.
3. **Соединены две внешние вершины x и y из одной компоненты.** Рассмотрим цикл C компоненты, содержащий вершины x, y, а также путь, ведущий из корня компоненты в этот цикл. Изменить рёбра сочетания вдоль этого пути (размер сочетания при этом не изменится) и стянуть найденный цветок.
4. **Каждая внешняя вершина соединена только со внутренними вершинами.** Значит, текущее паросочетание максимально.

Пояснение к последнему шагу

- Почему же паросочетание является максимальным, если каждая внешняя вершина соединена только со внутренними?

Пояснение к последнему шагу

- Почему же паросочетание является максимальным, если каждая внешняя вершина соединена только со внутренними?
- Пусть F содержит p внутренних вершин и q внешних.

Пояснение к последнему шагу

- Почему же паросочетание является максимальным, если каждая внешняя вершина соединена только со внутренними?
- Пусть F содержит p внутренних вершин и q внешних.
- Ясно, что $q - p = |U|$.

Пояснение к последнему шагу

- Почему же паросочетание является максимальным, если каждая внешняя вершина соединена только со внутренними?
- Пусть F содержит p внутренних вершин и q внешних.
- Ясно, что $q - p = |U|$.
- Итак, внешних вершин на $|U|$ больше, чем внутренних.

Пояснение к последнему шагу

- Почему же паросочетание является максимальным, если каждая внешняя вершина соединена только со внутренними?
- Пусть F содержит p внутренних вершин и q внешних.
- Ясно, что $q - p = |U|$.
- Итак, внешних вершин на $|U|$ больше, чем внутренних.
- Но каждая внешняя вершина может сочетаться только со внутренней. Значит, хотя бы для $|U|$ из них не найдется пары.

Пояснение к последнему шагу

- Почему же паросочетание является максимальным, если каждая внешняя вершина соединена только со внутренними?
- Пусть F содержит p внутренних вершин и q внешних.
- Ясно, что $q - p = |U|$.
- Итак, внешних вершин на $|U|$ больше, чем внутренних.
- Но каждая внешняя вершина может сочетаться только со внутренней. Значит, хотя бы для $|U|$ из них не найдется пары.
- Поскольку в текущем паросочетании пары нет как раз у $|U|$ вершин, то оно максимально.

Анализ времени работы

Лемма

Время работы представленного алгоритма есть $O(|V|^4)$.

Анализ времени работы

Лемма

Время работы представленного алгоритма есть $O(|V|^4)$.

Доказательство

- На каждом шаге мы либо увеличиваем размер F, либо уменьшаем размер G, либо находим дополняющий путь, либо заканчиваем работу.

Анализ времени работы

Лемма

Время работы представленного алгоритма есть $O(|V|^4)$.

Доказательство

- На каждом шаге мы либо увеличиваем размер F, либо уменьшаем размер G, либо находим дополняющий путь, либо заканчиваем работу.
- Дополняющий путь находится не более $|V|$ раз.

Анализ времени работы

Лемма

Время работы представленного алгоритма есть $O(|V|^4)$.

Доказательство

- На каждом шаге мы либо увеличиваем размер F, либо уменьшаем размер G, либо находим дополняющий путь, либо заканчиваем работу.
- Дополняющий путь находится не более $|V|$ раз.
- Стягивается не более $|V|$ цветков.

Анализ времени работы

Лемма

Время работы представленного алгоритма есть $O(|V|^4)$.

Доказательство

- На каждом шаге мы либо увеличиваем размер F, либо уменьшаем размер G, либо находим дополняющий путь, либо заканчиваем работу.
- Дополняющий путь находится не более $|V|$ раз.
- Стягивается не более $|V|$ цветков.
- Нахождение цветка или дополняющего пути требует времени $O(|E|)$.

Анализ времени работы

Лемма

Время работы представленного алгоритма есть $O(|V|^4)$.

Доказательство

- На каждом шаге мы либо увеличиваем размер F, либо уменьшаем размер G, либо находим дополняющий путь, либо заканчиваем работу.
- Дополняющий путь находится не более $|V|$ раз.
- Стягивается не более $|V|$ цветков.
- Нахождение цветка или дополняющего пути требует времени $O(|E|)$.
- Итого, $O(|V|^2|E|)$.

Покрытие путями ориентированного ациклического графа

Задача сводимая к задаче о максимальном паросочетании в двудольном графе

Покрытие путями ориентированного ациклического графа

Дан ориентированный ациклический граф G. Требуется покрыть его наименьшим числом путей, т.е. найти наименьшее по мощности множество непересекающихся по вершинам простых путей, таких, что каждая вершина принадлежит какому-либо пути.

Задача сводимая к задаче о максимальном паросочетании в двудольном графе

Покрытие путями ориентированного ациклического графа

Дан ориентированный ациклический граф G. Требуется покрыть его наименьшим числом путей, т.е. найти наименьшее по мощности множество непересекающихся по вершинам простых путей, таких, что каждая вершина принадлежит какому-либо пути.

Сведение к задаче о паросочетании в двудольном графе

Пусть дан орграф G с n вершинами.

Задача сводимая к задаче о максимальном паросочетании в двудольном графе

Покрытие путями ориентированного ациклического графа

Дан ориентированный ациклический граф G. Требуется покрыть его наименьшим числом путей, т.е. найти наименьшее по мощности множество непересекающихся по вершинам простых путей, таких, что каждая вершина принадлежит какому-либо пути.

Сведение к задаче о паросочетании в двудольном графе

Пусть дан орграф G с n вершинами.

1. Построим соответствующий ему двудольный граф H следующим образом: в каждой доле графа H будет по n вершин, обозначим их через a_i и b_i соответственно.

Задача сводимая к задаче о максимальном паросочетании в двудольном графе

Покрытие путями ориентированного ациклического графа

Дан ориентированный ациклический граф G. Требуется покрыть его наименьшим числом путей, т.е. найти наименьшее по мощности множество непересекающихся по вершинам простых путей, таких, что каждая вершина принадлежит какому-либо пути.

Сведение к задаче о паросочетании в двудольном графе
Пусть дан орграф G с n вершинами.

1. Построим соответствующий ему двудольный граф H следующим образом: в каждой доле графа H будет по n вершин, обозначим их через a_i и b_i соответственно.
2. Для каждого ребра (i, j) исходного графа G проведём соответствующее ребро (a_i, b_j).

Корретность сведения

- Каждому ребру G соответствует одно ребро H, и наоборот.

Корретность сведения

- Каждому ребру G соответствует одно ребро H, и наоборот.
- Если мы рассмотрим в G любой путь $P = (v_1, v_2, \ldots, v_k)$, то ему ставится в соответствие набор рёбер $(a_{v1}, b_{v2}), (a_{v2}, b_{v3}), \ldots, (a_{vk-1}, b_{vk})$ (это паросочетание в H).

Корретность сведения

- Каждому ребру G соответствует одно ребро H, и наоборот.
- Если мы рассмотрим в G любой путь $P = (v_1, v_2, \ldots, v_k)$, то ему ставится в соответствие набор рёбер $(a_{v1}, b_{v2}), (a_{v2}, b_{v3}), \ldots, (a_{vk-1}, b_{vk})$ (это паросочетание в H).
- Добавим «обратные» рёбра, т.е. образуем граф H' из графа H добавлением рёбер вида (b_i, a_i), $i = 1, \ldots, N$.

Корретность сведения

- Каждому ребру G соответствует одно ребро H, и наоборот.
- Если мы рассмотрим в G любой путь $P = (v_1, v_2, \ldots, v_k)$, то ему ставится в соответствие набор рёбер $(a_{v1}, b_{v2}), (a_{v2}, b_{v3}), \ldots, (a_{vk-1}, b_{vk})$ (это паросочетание в H).
- Добавим «обратные» рёбра, т.е. образуем граф H' из графа H добавлением рёбер вида (b_i, a_i), $i = 1, \ldots, N$.
- Тогда пути $P = (v_1, v_2, \ldots, v_k)$ в графе G будет соответствовать путь $Q' = (a_{v1}, b_{v2}), (b_{v2}, a_{v2}), (a_{v2}, b_{v3}), \ldots, (a_{vk-1}, b_{vk})$ в графе H'.

Корретность сведения

- Каждому ребру G соответствует одно ребро H, и наоборот.
- Если мы рассмотрим в G любой путь $P = (v_1, v_2, \ldots, v_k)$, то ему ставится в соответствие набор рёбер $(a_{v1}, b_{v2}), (a_{v2}, b_{v3}), \ldots, (a_{vk-1}, b_{vk})$ (это паросочетание в H).
- Добавим «обратные» рёбра, т.е. образуем граф H' из графа H добавлением рёбер вида (b_i, a_i), $i = 1, \ldots, N$.
- Тогда пути $P = (v_1, v_2, \ldots, v_k)$ в графе G будет соответствовать путь $Q' = (a_{v1}, b_{v2}), (b_{v2}, a_{v2}), (a_{v2}, b_{v3}), \ldots, (a_{vk-1}, b_{vk})$ в графе H'.
- Обратно, рассмотрим любой путь Q' в графе H', начинающийся в первой доле и заканчивающийся во второй доле.

Корретность сведения

- Каждому ребру G соответствует одно ребро H, и наоборот.
- Если мы рассмотрим в G любой путь $P = (v_1, v_2, \ldots, v_k)$, то ему ставится в соответствие набор рёбер $(a_{v1}, b_{v2}), (a_{v2}, b_{v3}), \ldots, (a_{vk-1}, b_{vk})$ (это паросочетание в H).
- Добавим «обратные» рёбра, т.е. образуем граф H' из графа H добавлением рёбер вида (b_i, a_i), $i = 1, \ldots, N$.
- Тогда пути $P = (v_1, v_2, \ldots, v_k)$ в графе G будет соответствовать путь $Q' = (a_{v1}, b_{v2}), (b_{v2}, a_{v2}), (a_{v2}, b_{v3}), \ldots, (a_{vk-1}, b_{vk})$ в графе H'.
- Обратно, рассмотрим любой путь Q' в графе H', начинающийся в первой доле и заканчивающийся во второй доле.
- Очевидно, Q' снова будет иметь вид $Q' = (a_{v1}, b_{v2}), (a_{v2}, b_{v3}), \ldots, (a_{vk-1}, b_{vk})$, и ему можно поставить в соответствие в графе G путь $P = (v_1, v_2, \ldots, v_k)$.

Замечание

- Здесь есть одна тонкость: v_1 могло совпадать с v_k, поэтому путь P получился бы циклом.

Замечание

- Здесь есть одна тонкость: v_1 могло совпадать с v_k, поэтому путь P получился бы циклом.
- Однако по условию граф G ациклический, поэтому это вообще невозможно (это единственное место, где используется ацикличность графа G;

Замечание

- Здесь есть одна тонкость: v_1 могло совпадать с v_k, поэтому путь P получился бы циклом.
- Однако по условию граф G ациклический, поэтому это вообще невозможно (это единственное место, где используется ацикличность графа G;
- тем не менее, на циклические графы описываемый здесь метод вообще нельзя обобщить).

Что мы узнали из этой лекции?

- Основным моментом в поиске максимального паросочетания является поиск дополняющих путей.

Что мы узнали из этой лекции?

- Основным моментом в поиске максимального паросочетания является поиск дополняющих путей.
- Искать дополняющие пути в двудольном графе гораздо легче.

Что мы узнали из этой лекции?

- Основным моментом в поиске максимального паросочетания является поиск дополняющих путей.
- Искать дополняющие пути в двудольном графе гораздо легче.
- В двудольном графе размер максимального паросочетания совпадает с размером минимального вершинного покрытия.

Что мы узнали из этой лекции?

- Основным моментом в поиске максимального паросочетания является поиск дополняющих путей.
- Искать дополняющие пути в двудольном графе гораздо легче.
- В двудольном графе размер максимального паросочетания совпадает с размером минимального вершинного покрытия.
- В общем случае максимальное паросочетание находится за время $O(|V|^4)$.

Что мы узнали из этой лекции?

- Основным моментом в поиске максимального паросочетания является поиск дополняющих путей.
- Искать дополняющие пути в двудольном графе гораздо легче.
- В двудольном графе размер максимального паросочетания совпадает с размером минимального вершинного покрытия.
- В общем случае максимальное паросочетание находится за время $O(|V|^4)$.
- Задача о покрытии путями ориентированного ациклического графа сводится к задаче о паросочетании в двудольном графе.

Спасибо за внимание!