

Графовые модели

Судаков Владимир Анатольевич,
доктор технических наук,
Главный научный сотрудник,
Руководитель магистерской программы

sudakov@ws-dss.com

Литература

- Л. Э. Сукар. Вероятностные графовые модели. Принципы и приложения
<https://dmkpress.com/catalog/computer/data/978-5-97060-874-6/>
- Рассел Стюарт, Норвиг Питер. Искусственный интеллект. Современный подход (AIMA-2)
- Джоэл Грас. Data Science. Наука о данных с нуля <https://bhv.ru/product/nauka-o-danniyh-s-nulya-per-s-angl-2-e-izd/>

Содержание

- Основы теории графов
- Анализ социальных сетей и большие данные
- Основы теории вероятностей
- Графовые вероятностные модели
 - Байесовские классификаторы
 - Скрытые марковские модели
 - Байесовские сети: представление и вывод
 - Байесовские сети: обучение
 - Динамические и временные байесовские сети
 - Марковские процессы принятия решений
 - Реляционные вероятностные графические модели
 - Причинно-следственные графические модели
- Графы принятия решений
- Модели системной динамики

Программное обеспечение

- Git  GitHub

- Python



- Jupyter Notebook



- Библиотеки:

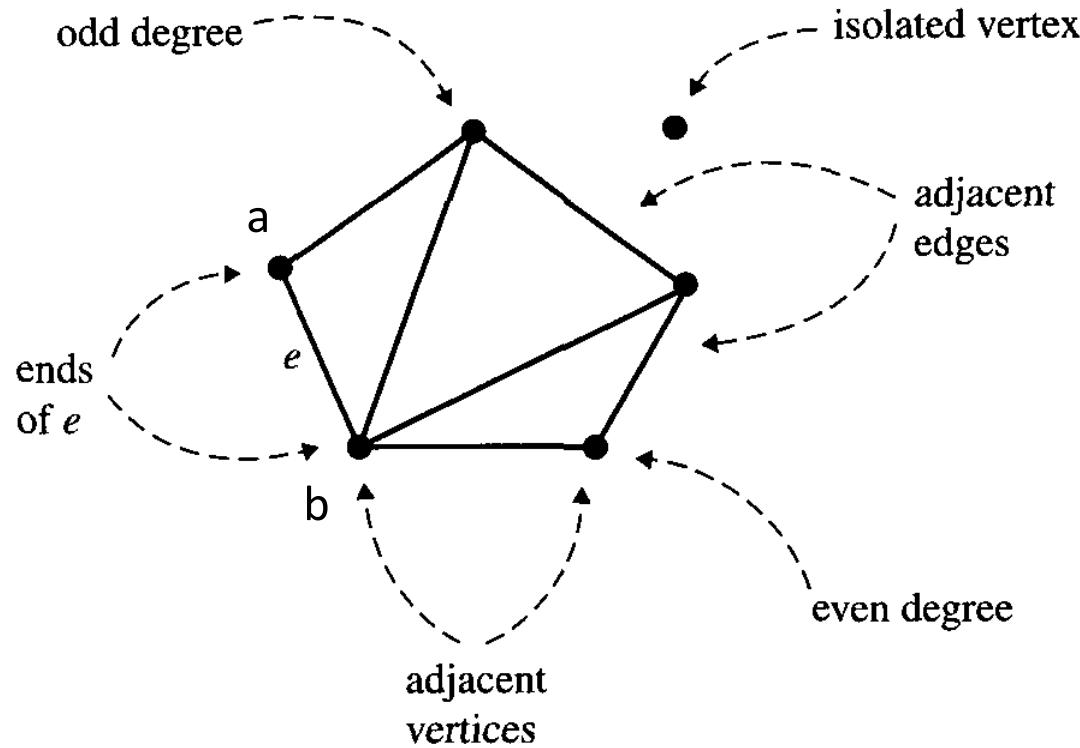
- networkX
- pgmpy



Определение графа

- Граф $G=(V,E)$ состоит из конечного непустого множества V (элементы – вершины) и конечного множества E пар различных элементов V , называемых ребрами графа G .
- Вершины ребер могут быть:
 - упорядоченными (ориентированный граф);
или
 - неупорядоченными (неориентированный граф).

Названия элементов графа



$$e=(a,b)$$

Ребро e инцидентно вершинам a и b .

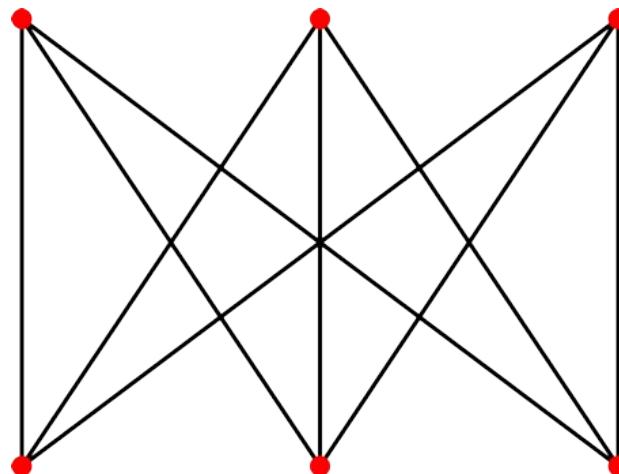
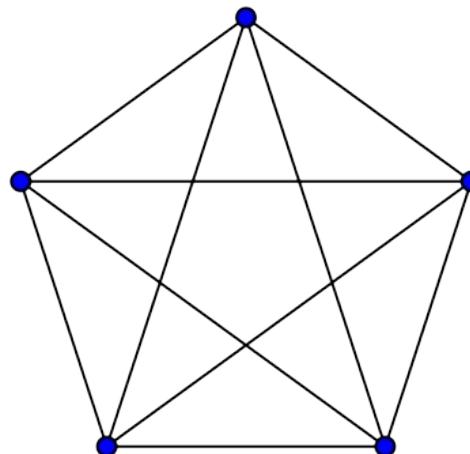
степень вершины $\text{degree}(v)$ – это число инцидентных ей ребер

Определения некоторых «особых» графов

- Граф у которого каждая пара вершин соединена ребром, называется *полным* (*насыщенным*) графом.
- Граф, степени вершин которого равны, называется *регулярным* графом.
- Если множество вершин графа можно разбить на 2 подмножества V_1 и V_2 (биразбиение), такие что для любого ребра $e=(v_1, v_2)$ $v_1 \in V_1$, а $v_2 \in V_2$, то граф называется *двудольным*

Планарный граф

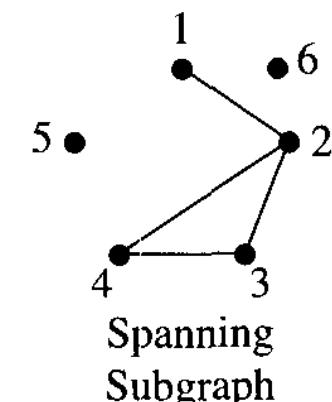
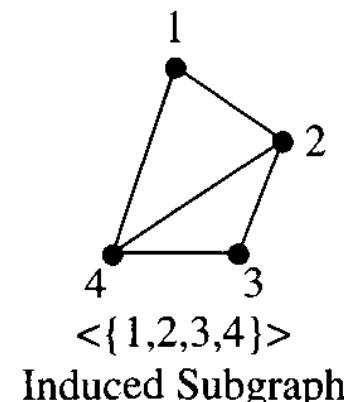
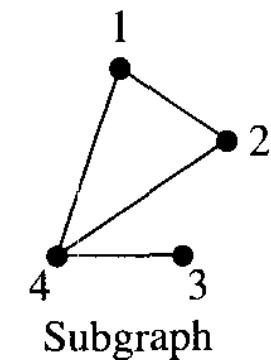
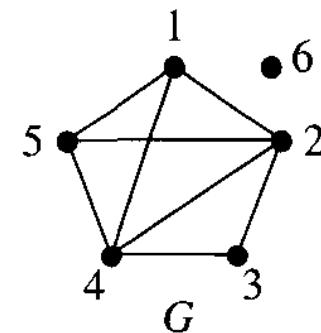
- *Планарный граф* — граф, который может быть изображён на плоскости без пересечения ребер. Области, на которые граф разбивает плоскость, называются его **гранями**.
- Примеры непланарных графов:



- Задача: Есть три дома и три колодца. Можно ли так проложить дорожки между домами и колодцами, чтобы от каждого дома к каждому колодцу вела дорожка, и никакие две дорожки не пересекались бы.

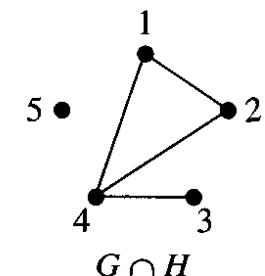
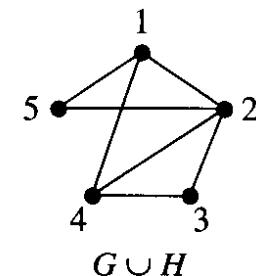
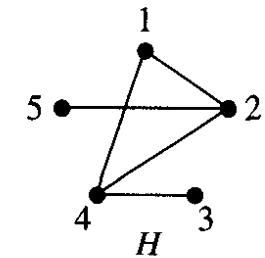
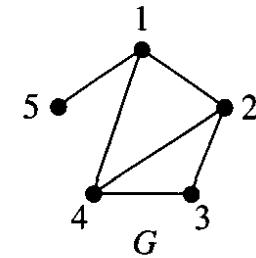
Подграф

- Граф $H=(V_1, E_1)$ называется подграфом графа $G=(V, E)$, если $V_1 \subseteq V$ и $E_1 \subseteq E$ и оба конца любого ребра $e \in E_1$ принадлежат V_1 .
- H – оставный подграф если $V=V_1$.
- H – индуцированный подграф, если E_1 состоит из всех ребер G , оба конца которых принадлежат V_1 .



Операции на графах

- $G = (V_1, E_1)$
- $H = (V_2, E_2)$
- $G \cup H = (V_1 \cup V_2, E_1 \cup E_2)$
- $G \cap H = (V_1 \cap V_2, E_1 \cap E_2)$

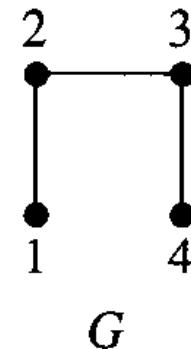


Дополнение графа

- Операция **дополнения графа $G=(X,F)$ по отображению** до графа $H=(Y,P)$: $G \subseteq H$, называется такой граф $G_H=(Y,T)$, для которого для любых $y \in Y$, $T_y = P_y \setminus F_y$
- Граф G_H задается на всем множестве вершин графа H , т.е. на Y и включает только те ребра, которые есть в H , но отсутствуют в G .
- Универсальный граф – это граф, для которого все рассматриваемые в данном круге задач графы являются подграфами (это полный граф).

Дополнение графа

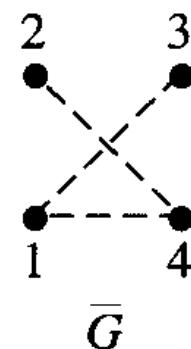
- Дополнение графа G по отображению до универсального графа обозначается как $G \underline{\quad}$



- Разность графов

$$Q = G \setminus H = G \cap H$$

$\underline{\quad}$



Задачи о рукопожатиях

- На встрече выпускники ВУЗа пожимали друг-другу руки. Можно ли сказать что-нибудь о числе рукопожатий? Могут ли все пожать разное кол-во рук?
- Дан граф G , в котором более одной вершины. Степени по крайней мере 2-х вершин равны.
- В любом графе число нечетных вершин четно

Степени по крайней мере двух вершин в графе равны

Доказательство.

По индукции.

База. $n_0=2$ – очевидно.

Шаг индукции. $n>=n_0$, Пусть в графе с n вершинами условие выполняется, добавим вершину v , если она изолирована, то степени вершин не изменятся, а если нет то $1<=\deg(v)<=n$, значит в графе не более n различных степеней вершин, но вершин всего $n+1$, то согласно принципу Дерихле по крайней мере у двух вершин степени совпадают.

Степени вершин

$$G = (V, E), \sum_{v \in V} \deg(v) = 2|E|$$

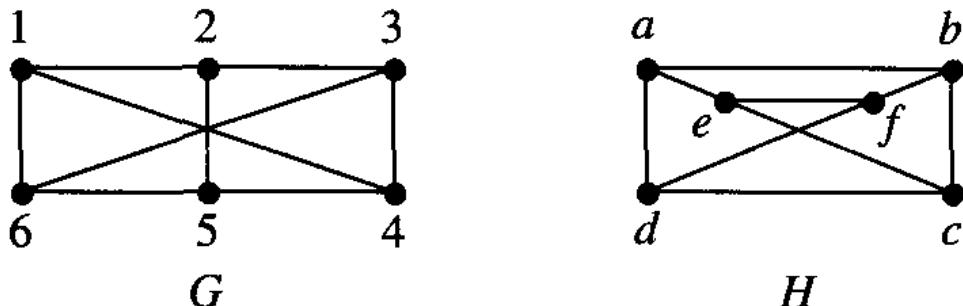
Теорема. В любом графе число нечетных вершин четно

$$\sum_{v \in V} \deg(v) = \sum_{v \text{ чет}} \deg(v) + \sum_{v \text{ нечет}} \deg(v)$$
$$\sum_{v \text{ нечет}} \deg(v) = 2|E| - \sum_{v \text{ чет}} \deg(v)$$

Изоморфизм графов

Графы $G=(V_G, E_G)$ и $H=(V_H, E_H)$ называются изоморфными, если существует биекция между множествами вершин графов

$F: V_G \rightarrow V_H$ такая, что $(u, v) \in E_G$ тогда и только тогда, когда $(F(u), F(v)) \in E_H$.



$$F: V(G) \rightarrow V(H)$$

$$1 \longrightarrow a$$

$$2 \longrightarrow b$$

$$3 \longrightarrow f$$

$$4 \longrightarrow e$$

$$5 \longrightarrow c$$

$$6 \longrightarrow d$$

$$E(G) \longrightarrow E(H)$$

$$(1, 2) \longrightarrow (F(1), F(2)) = (a, b)$$

$$(2, 3) \longrightarrow (F(2), F(3)) = (b, f)$$

$$(3, 4) \longrightarrow (F(3), F(4)) = (f, e)$$

$$(4, 5) \longrightarrow (F(4), F(5)) = (e, c)$$

$$(5, 6) \longrightarrow (F(5), F(6)) = (c, d)$$

$$(6, 1) \longrightarrow (F(6), F(1)) = (d, a)$$

$$(2, 5) \longrightarrow (F(2), F(5)) = (b, c)$$

$$(1, 4) \longrightarrow (F(1), F(4)) = (a, e)$$

$$(3, 6) \longrightarrow (F(3), F(6)) = (f, d)$$

Аналитические способы задания графа

1) $G=(X, U)$

$$X = \{x_1, x_2, x_3, x_4\}$$

$$U = \{(x_1, x_1), (x_1, x_2), (x_1, x_3), (x_1, x_4), (x_2, x_3), (x_3, x_2), (x_3, x_4)\}$$

2) $G=(X, F)$

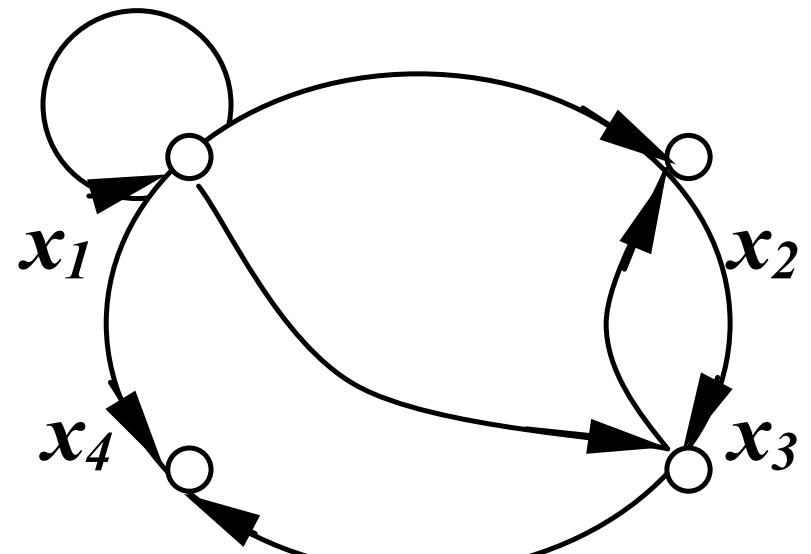
$$X = \{x_1, x_2, x_3, x_4, x_5\}$$

$$Fx_1 = \{x_1, x_2, x_3, x_4\}$$

$$Fx_2 = \{x_3\}$$

$$Fx_3 = \{x_2, x_4\}$$

$$Fx_4 = \emptyset$$



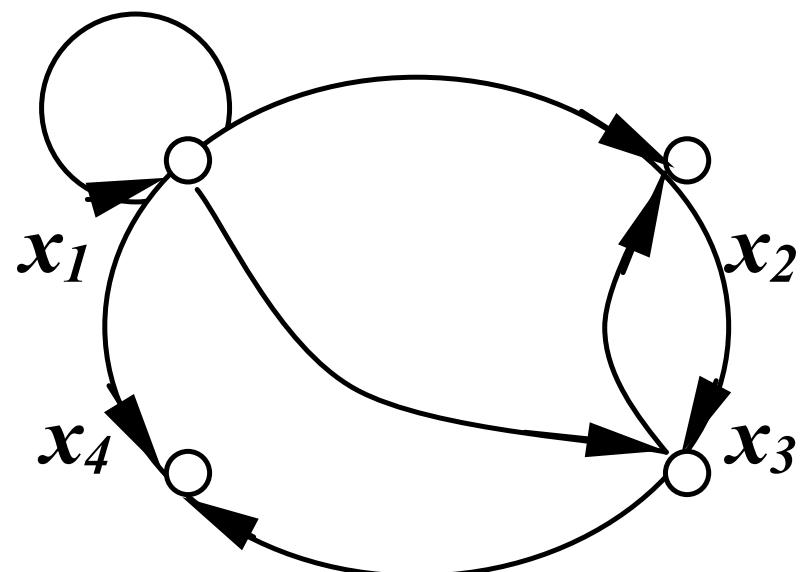
Матричные способы задания графов

- Матрица смежности R

$$R = \{r_{ij}\} \quad i, j = 1..n$$

$r_{i,j} = \begin{cases} 1, & \text{если } x_j \in Fx_i \text{ (т.е. есть дуга } x_i, x_j) \\ 0, & \text{если } x_j \notin Fx_i \text{ (т.е. дуга } x_i, x_j \text{ – отсутствует)} \end{cases}$

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



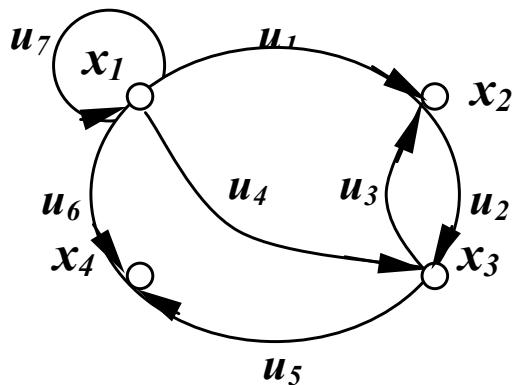
Матрица инцидентности

прямоугольная матрица $A=\{a_{ik}\}$, строки которой поставлены в соответствие вершинам графа, а столбцы есть его ребра. Тогда элементы матрицы a_{ik} будут равны:

$$a_{ik} = \begin{cases} 0 & \text{если вершина } x_i \text{ не является граничной точкой для ребра } u_k \\ 1 & \text{если дуга } u_k \text{ заходит в вершину } x_i \\ -1 & \text{если дуга } u_k \text{ исходит из вершины } x_i \end{cases}$$

В соответствии с этим в каждом столбце матрицы инциденций будет либо один ненулевой элемент (+1), если это ребро – петля, либо два ненулевых элемента (+1,-1), если ребро – дуга.

Пример:



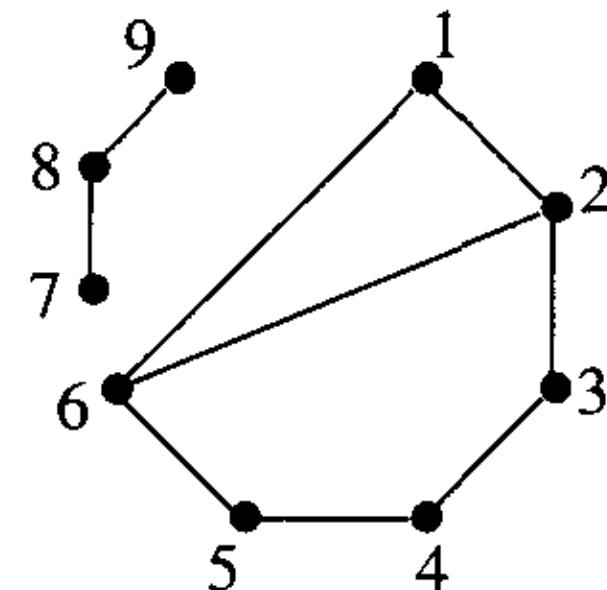
Матрица А будет иметь вид:

|||

	u ₁	u ₂	u ₃	u ₄	u ₅	u ₆	u ₇
x ₁	-1	0	0	-1	0	-1	1
x ₂	1	-1	1	0	0	0	0
x ₃	0	1	-1	1	-1	0	0
x ₄	0	0	0	0	1	1	0

Список смежности

Vertices	List of Adjacencies		
1	2	6	
2	1	6	3
3	2	4	
4	3	5	
5	4	6	
6	1	5	2
7	8		
8	7	9	
9	8		



Путь в графе

- Путем (или ориентированным маршрутом) в ориентированном графе называется последовательность дуг, в которой конечная вершина всякой дуги (отличной от последней) является начальной вершиной следующей.
- Путь имеет начальную и конечную вершины.
- Длина пути – число дуг, составляющих путь.
- Простой путь – путь, в котором никакая дуга не встречается дважды.
- Элементарный путь – путь, в котором никакая вершина не встречается дважды.
- Контур – путь, у которого конечная и начальная вершины совпадают (может быть простой и элементарный).
- Цепь (маршрут) – последовательность ребер неориентированного графа , когда любые два соседних ребра в этой последовательности – смежные. Цепь имеет две граничные вершины, каждой из которых инцидентно только одно ребро. Цепь может быть простой (каждое ребро встречается только один раз) и элементарной (каждая вершина встречается только один раз).
- Цикл – это замкнутая цепь конечной длины, у которой граничные вершины совпадают. Цикл может быть простой и элементарный.

Алгоритм поиска в глубину (Depth-first search, DFS)

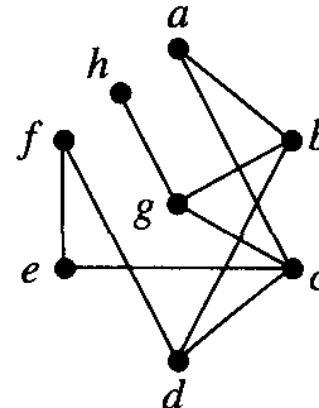
```

used_vertices = []

def dfs(graph, vertex, used_vertices)
    used_vertices << vertex
    graph.adjacent_vertices(vertex).each do |v|
        if not used_vertices.include?
            puts "#{vertex}-#{v}"
            dfs(graph, v,
used_vertices)
        end
    end
end

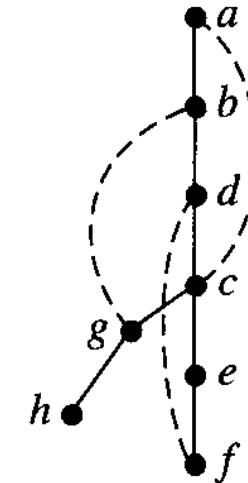
dfs(graph, initial_vertex, used_vertices)
puts "result:"
puts used_vertices

```



Vertices	Lists of Adjacencies
a	b c
b	a d g
c	a d e g
d	c b f
e	c f
f	e d
g	b c h
h	g

Depth First Search
beginning at a



— edge to unvisited vertex (tree edge)
--- edge to previously visited vertex

Алгоритм поиска в ширину (Breadth-first search, BFS)

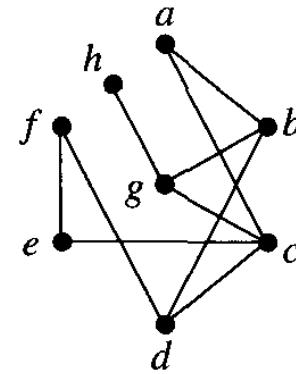
```

used_vertices = []
queue = [initial_vertex]

until queue.empty? do
    vertex = queue.shift
    used_vertices << vertex
    graph.adjacent_vertices(vertex).each do |v|
        if not used_vertices.include?
            v and
                not queue.include? v
                puts "#{vertex}-#{v}"
                queue << v
        end
    end
end

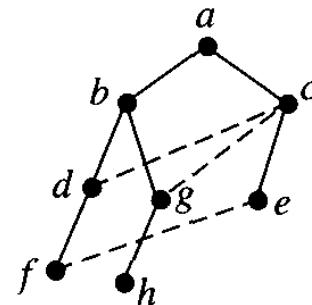
puts "result:"
puts used_vertices

```

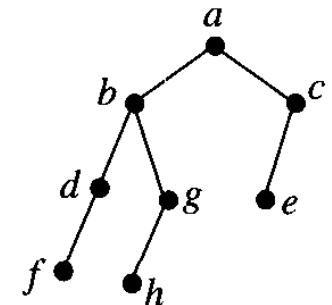


Vertices	Lists of Adjacencies
a	b c
b	a d g
c	a d e g
d	c b f
e	c f
f	e d
g	b c h
h	g

Breadth First Search
Beginning at a



Breadth First
Search Tree



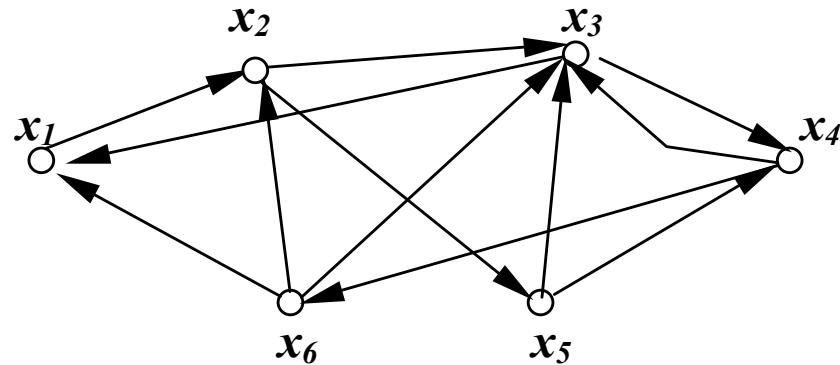
Пути в графе

- **Гамильтонов путь в графе** – это элементарный путь, проходящий через все вершины графа, т.е. путь, проходящий через все вершины в графе, причем каждая вершина встречается только один раз. Если этот путь замкнут, то говорят о Гамильтоновом контуре (Гамильтоновом цикле), а граф – Гамильтонов.
- **Эйлеров путь в графе** – простой путь в графе, проходящий через все дуги графа, т.е. путь, проходящий через все дуги графа, причем каждая дуга встречается в пути только один раз. Если этот путь замкнут, то говорят об Эйлеровом контуре, а граф – Эйлеров.

Поиск гамильтонова пути (ГП)

- Метод перебора - аналог алгоритма поиска в глубину (алгоритм Робертса и Флореса):
 1. $S=[x_i]$ - начальная вершина пути
 2. Если $|S|$ =число вершин в графе, то ГП найден, **завершить работу.**
 3. Если есть смежные с последней вершиной пути *не пройденные* вершины, то поместить в S следующую смежную не пройденную вершину с минимальным номером и переход на п.2
 4. Если $|S|=1$, то ГП нет, **завершить работу.**
 5. Удалить из S последнюю вершину и переход на п.3.
- Нет строгого простого правила проверки существования ГП.

Пример поиска ГП



1	2
2	3 5
3	1 4
4	3 6
5	3 4
6	1 2 3

1. $S=[x_1]$ К ней добавляем вершину x_2
2. $S=[x_1, x_2]$ Добавляем x_3
3. $S=[x_1, x_2, x_3]$ Добавляем x_4
4. $S=[x_1, x_2, x_3, x_4]$ Добавляем x_6
5. $S=[x_1, x_2, x_3, x_4, x_6]$ Дальше добавлять из строки, соответствующей вершине x_6 ничего нельзя, т.к. все ее элементы уже входят в S . Возвращаемся на предыдущий шаг
6. $S=[x_1, x_2, x_3, x_4]$ Кроме x_6 добавить нечего. Возвращаемся
7. $S=[x_1, x_2, x_3]$ Кроме x_4 добавить нечего. Возвращаемся
8. $S=[x_1, x_2]$ Добавляем x_5
9. $S=[x_1, x_2, x_5]$ Добавляем x_3
10. $S=[x_1, x_2, x_5, x_3]$ Добавляем x_4
11. $S=[x_1, x_2, x_5, x_3, x_4]$ Добавляем x_6
12. $S=[x_1, x_2, x_5, x_3, x_4, x_6]$ Гамильтонов путь.

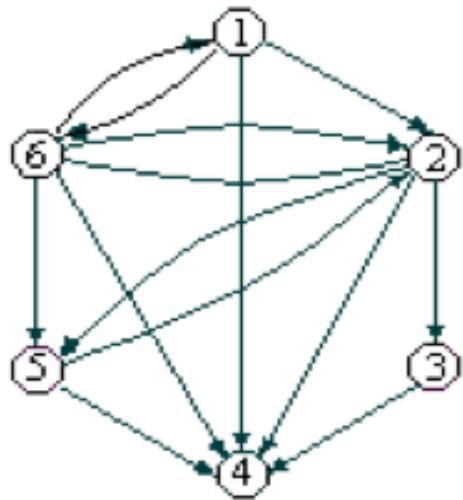
Дуга (x_6, x_1) дает Гамильтонов цикл.

Сокращение перебора при поиске ГП

Алгоритм Фаулкса

1. Построить матрицу R^1 - достижимости не более чем за один шаг. Матрица R^1 может быть получена из матрицы смежности R в результате подстановки единиц на главной диагонали.
2. Вычислять $R^N = \underbrace{R^1 \otimes R^1 \otimes \dots \otimes R^1}_N$ $r_{ij}^N = \bigvee_{k=1}^n r_{ik}^{N-1} \cdot r_{kj}$
3. Удалять начальные и конечные вершины. Вершина является начальной вершиной ГП, если во всей строке матрицы стоят единицы, а во всем столбце (за исключением их пересечения) - нули. Вершина является конечной вершиной ГП, то во всей строке матрицы стоят нули (за исключением их пересечения), а во всем столбце - единицы.
4. Прекратить вычисления когда $R^n = R^{n-1}$.

Пример применения алгоритма Фаулкса



$$R^1 = A_3$$

	A_1	A_2	A_3	A_4	A_5	A_6
A_1	1	1	0	1	0	1
A_2	0	1	1	1	1	1
A_3	0	0	1	1	0	0
A_4	0	0	0	1	0	0
A_5	0	1	0	1	1	0
A_6	1	1	0	1	1	1

$$(R^1)' = A_2$$

	A_1	A_2	A_3	A_5	A_6
A_1	1	1	0	0	1
A_2	0	1	1	1	1
A_3	0	0	1	0	0
A_5	0	1	0	1	0
A_6	1	1	0	1	1

Пример применения алгоритма Фаулкса (2)

$$(R^2)' = \begin{array}{c|ccccc} & A_1 & A_2 & A_3 & A_5 & A_6 \\ \hline A_1 & 1 & 1 & 1 & 1 & 1 \\ A_2 & 1 & 1 & 1 & 1 & 1 \\ A_3 & 0 & 0 & 1 & 0 & 0 \\ A_5 & 0 & 1 & 1 & 1 & 1 \\ A_6 & 1 & 1 & 1 & 1 & 1 \end{array} \quad \begin{array}{c|ccccc} & A_1 & A_2 & A_3 & A_5 & A_6 \\ \hline A_1 & 1 & 1 & 1 & 1 & 1 \\ A_2 & 1 & 1 & 1 & 1 & 1 \\ A_5 & 0 & 1 & 1 & 1 & 1 \\ A_6 & 1 & 1 & 1 & 1 & 1 \end{array} \quad (R^3)' = \begin{array}{c|ccccc} & A_1 & A_2 & A_5 & A_6 \\ \hline A_1 & 1 & 1 & 1 & 1 \\ A_2 & 1 & 1 & 1 & 1 \\ A_5 & 1 & 1 & 1 & 1 \\ A_6 & 1 & 1 & 1 & 1 \end{array}$$

$$R^b = A_3 \begin{array}{c|cccccc} & A_1 & A_2 & A_3 & A_4 & A_5 & A_6 \\ \hline A_1 & 1 & 1 & 1 & 1 & 1 & 1 \\ A_2 & 1 & 1 & 1 & 1 & 1 & 1 \\ A_3 & 0 & 0 & 1 & 1 & 0 & 0 \\ A_4 & 0 & 0 & 0 & 1 & 0 & 0 \\ A_5 & 1 & 1 & 1 & 1 & 1 & 1 \\ A_6 & 1 & 1 & 1 & 1 & 1 & 1 \end{array} \quad R^e = A_5 \begin{array}{c|ccccc|cc} & A_{\textcolor{blue}{1}} & A_{\textcolor{blue}{2}} & A_{\textcolor{teal}{5}} & A_{\textcolor{teal}{6}} & A_{\textcolor{teal}{3}} & A_{\textcolor{teal}{4}} \\ \hline A_{\textcolor{blue}{1}} & 1 & 1 & 1 & 1 & 1 & 1 \\ A_{\textcolor{blue}{2}} & 1 & 1 & 1 & 1 & 1 & 1 \\ A_{\textcolor{teal}{5}} & 1 & 1 & 1 & 1 & 1 & 1 \\ A_{\textcolor{teal}{6}} & 1 & 1 & 1 & 1 & 1 & 1 \\ A_{\textcolor{teal}{3}} & 0 & 0 & 0 & 0 & 1 & 1 \\ A_{\textcolor{teal}{4}} & 0 & 0 & 0 & 0 & 0 & 1 \end{array}$$

Теорема Эйлера

- для неориентированных графов: для того чтобы на графе существовал замкнутый эйлеров путь (ЭП), необходимо и достаточно, чтобы граф был связным и степени всех его вершин были четными.
- для ориентированных графов: для того чтобы на ориентированном графе существовал замкнутый ЭП, необходимо и достаточно, чтобы граф был связным и для каждой его вершины полустепень исхода равнялась полустепени захода.

Алгоритм поиска ЭП

0. Задается начальный шаг итерации $k = 0$. Задается начальная (она же и конечная в случае замкнутого ЭП) вершина ЭП V_0 . Все ребра графа считаются непомеченными. Число непомеченных ребер S равно числу ребер графа.

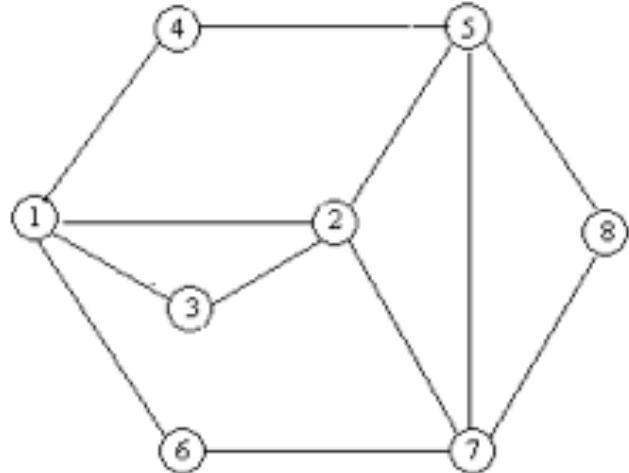
1. $k=k+1$. Определяются все вершины, непосредственно связанные непомеченными ребрами с вершиной V_{k-1} .
2. Определяется число S_k непомеченных ребер, инцидентных вершине V_{k-1} .

Если $S_k = 1$, то вторая вершина, инцидентная этому ребру, становится очередной вершиной V_k ЭП. Это ребро помечается. Возвращаемся на п. 1.

Если $S_k > 1$, то среди вершин, смежных V_{k-1} , выбирается вершина с наименьшим номером k , которая становится очередной вершиной V_k ЭП. Ребро, соединяющее эту вершину с вершиной V_{k-1} , помечается. Возвращаемся на п. 1.

3. $S_k = 0$. Для вершины, принадлежащей полученному таким образом пути и имеющей непомеченные инцидентные ребра, строится цикл по непомеченным ребрам. Определение новой вершины этого цикла начинается с п. 1.
4. Построенный таким образом новый цикл объединяется с ранее построенным в вершине, с которой начиналось построение нового цикла.
5. Если в цикле есть еще вершины, имеющие непомеченные инцидентные ребра, то возвращаемся на п. 3. Если таких ребер нет, то ЭП найден.

Пример поиска ЭП



0. $k = 0$. Выберем начальную вершину с номером 3, т.е.

$V_0 = 3$. Число непомеченных ребер $S = 12$.

1. $k = 1$. С вершиной $V_0 = 3$ связаны вершины 1, 2.

2. Число непомеченных ребер для 3-й вершины $S_1 = 2$.

Следующей вершиной ЭП является вершина с номером 1.

Ребро 3-1 помечается. Возвращаемся к п. 1.

1. $k = 2$. С вершиной 1 связаны вершины 2, 4, 6.

2. $S_2 = 3$. Вершина 2. Помечается ребро 1-2.

Возвращаемся к п. 1.

1. $k = 3$. С вершиной 2 связаны вершины 3, 5, 7.

2. $S_3 = 3$. Вершина 3. Помечается ребро 2-3.

Возвращаемся к п. 1.

1. $k = 4$. С вершиной 3 связаны вершины 1, 2.

2. $S_4 = 0$. Вершин таких нет. Следовательно, получен цикл 3-1-2-3, который еще не является ЭП.

Пример поиска ЭП (2)

3. Вершины 1 и 2, входящие в этот цикл, имеют непомеченные инцидентные ребра. Рассмотрим вершину 1 в качестве начальной точки следующего цикла $A_4 = 1$.

1. $k = 5$. Вершины 4, 6.

2. $S_5 = 2$. Вершина 4. Ребро 1-4.

1. $k = 6$. Вершина 5.

2. $S_6 = 1$. Вершина 5. Ребро 4-5.

1. $k = 7$. Вершины 2, 7, 8.

2. $S_7 = 3$. Вершина 2. Ребро 5-2.

1. $k = 8$. Вершина 7.

2. $S_8 = 1$. Вершина 7. Ребро 2-7.

1. $k = 9$. Вершины 5, 6, 8.

2. $S_9 = 3$. Вершина 5. Ребро 7-5.

1. $k = 10$. Вершина 8.

2. $S_{10} = 1$. Вершина 8. Ребро 5-8.

1. $k = 11$. Вершина 7.

2. $S_{11} = 1$. Вершина 7. Ребро 8-7.

1. $k = 12$. Вершина 6.

2. $S_{12} = 1$. Вершина 6. Ребро 7-6.

1. $k = 13$. Вершина 1.

2. $S_{13} = 1$. Вершина 1. Ребро 6-1.

1. $k = 14$. Вершин, связанных непомеченными ребрами, нет.

3. $S_{14} = 0$. В найденных циклах нет вершин, имеющих инцидентные непомеченные ребра.

4. Для нахождения ЭП необходимо объединить два найденных цикла 3-1-2-3 и 1-4-5-2-7-5-8-7-6-1 в вершине 1. Результатом объединения будет путь: 3-1-4-5-2-7-5-8-7-6-1-2-3.

5. Так как в этом пути нет вершин с непомеченными ребрами, то найденный путь является эйлеровым.

Расстояния в графе

В теории графов расстоянием между двумя вершинами графа называется число рёбер в кратчайшем пути (также называемым геодезической графа). Расстояние в графе называется также геодезическим расстоянием.

Эксцентриситетом $\epsilon(v)$ вершины v называется наибольшее геодезическое расстояние между v и любой другой вершиной графа. То есть расстояние до самой дальней от v вершины графа.

$$\epsilon(v) = \max_{u \in V} d(v, u)$$

Радиусом r графа называется минимальный эксцентриситет среди всех вершин графа

$$r = \min_{v \in V} \epsilon(v).$$

Диаметром d графа называется максимальный эксцентриситет среди всех вершин графа. Таким образом, d – это наибольшее расстояние между всеми парами вершин графа

$$d = \max_{v \in V} \epsilon(v)$$

Чтобы найти диаметр графа сначала находят кратчайшие пути между всеми парами вершин. Наибольшая длина кратчайшего пути есть диаметр графа.

Центральной вершиной графа радиусом r называется вершина, эксцентриситет которой равен r . То есть вершина, на которой достигается радиус

$$\epsilon(v) = r.$$

Периферийной вершиной графа диаметра d называется вершина, эксцентриситет которой равен d

$$\epsilon(v) = d.$$

Определение связности графа

- Неориентированный граф называется **связным**, если для любой пары вершин $x_i, x_j \in X$ существует хотя бы один маршрут, их соединяющий. В противном случае граф несвязан. Несвязный граф может быть единственным образом разбит на группы взаимосвязанных вершин (**на связные компоненты**) таким образом, что не существует маршрута, соединяющей какие-либо из вершин различных компонент.

Определение связности графа с помощью элементов алгоритма Фаулкса

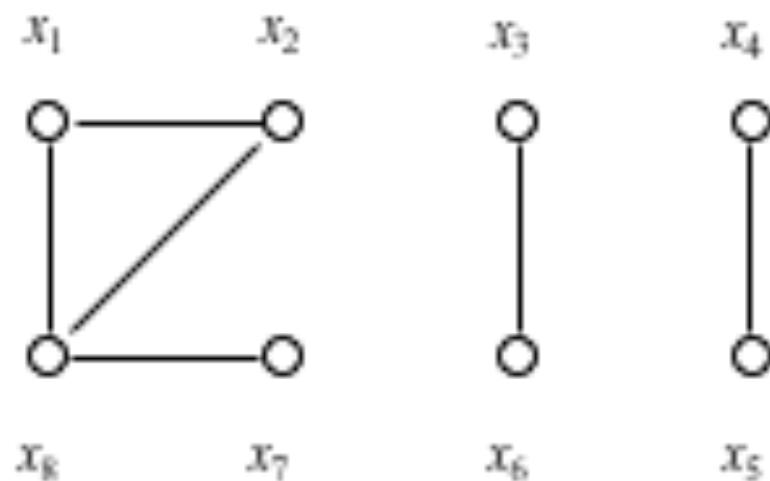
- Пусть дана матрица смежности R неориентированного графа G (матрица R симметрична относительно главной диагонали), имеющего n вершин
- На основании матрицы смежности R получим матрицу достижимости не более чем за один шаг R^1
- Найдем матрицу R^N путем булевского перемножения матриц ($R^N = R^{N-1} \otimes R^1$) до тех пор, пока не достигнем $R^N = R^{N-1}$.
- Необходимо в некоторой i - строке матрицы R^N найти элементы $r_{ij}^N = 1$, $i, j = 1, \dots, n$. Номера столбцов j , где $r_{ij} = 1$, и будут номерами вершин, входящих в связный подграф, включающий в себя i -ю вершину.
Рассматривая последовательно все несовпадающие строки матрицы можно выявить все t связных подграфов исходного графа G .

Пример проверки связности

$$R^1 = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 \end{pmatrix} \quad R^2 = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{pmatrix}$$

$$R^2 = R^3$$

$$G_1 = \{1, 2, 7, 8\}; \quad G_2 = \{3, 6\}; \\ G_3 = \{4, 5\}$$



Связность ориентированного графа

- орграф **сильно связан**, если существует хотя бы один путь как из $x_i \rightarrow x_j$, так и $x_j \rightarrow x_i$, т.е. любые две вершины взаимно достижимы
- орграф **слабо связан**, если для любой пары $(x_i, x_j) \in X$ существует маршрут, их соединяющий, но не для всех пар существует путь, их соединяющий
- орграф **связан односторонне**, если для любой пары $(x_i, x_j) \in X$ существует хотя бы один путь, их соединяющий, из $x_i \rightarrow x_j$ или $x_j \rightarrow x_i$ (или оба одновременно).

Матрицы L и Q

Матрица достижимости L:

$$l_{ik} = \begin{cases} 1 & \text{если вершина } x_k \text{ достижима из вершины } x_i; \\ 0 & \text{если вершина } x_k \text{ не достижима из вершины } x_i; \end{cases}$$

L=

$$R^N = \underbrace{R^1 \otimes R^1 \otimes \dots \otimes R^1}_N \quad N: R^N = R^{N-1} \quad N \leq n-1$$

Матрица контрдостижимости Q :

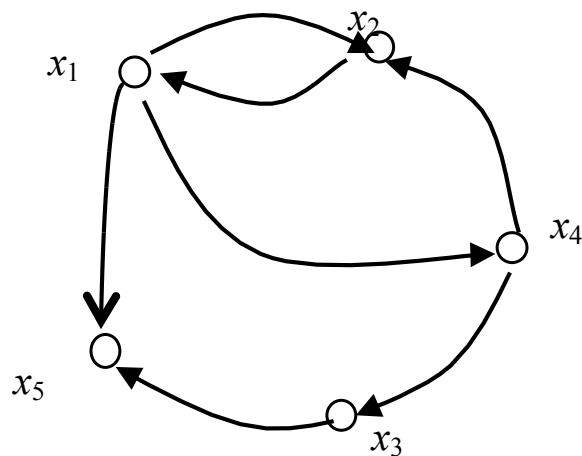
$$q_{ik} = \begin{cases} 1 & \text{если вершина } x_i \text{ достижима из вершины } x_k; \\ 0 & \text{если вершина } x_i \text{ не достижима из вершины } x_k; \end{cases}$$

$$Q = L^T$$

Сильные компоненты графа

- Под **сильной компонентой (СК)** графа G будем понимать подграф G' графа G , являющийся сильно связным графом, который не содержится в любом другом сильном подграфе графа G .
- СК содержит максимальное возможно число вершин
- Сильные компоненты произвольного графа G могут быть найдены в результате поэлементного логического перемножения матриц L и Q . Номера столбцов в i -й строке, для которых элемент матрицы равен 1, соответствуют номерам вершин, входящих в данную сильную компоненту.

Пример поиска сильных компонент

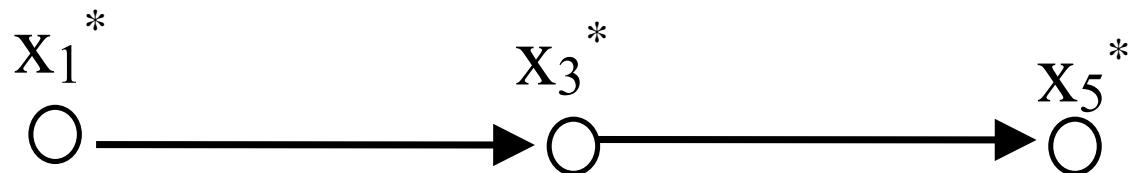


$$R^1 = \begin{pmatrix} 11011 \\ 11000 \\ 00101 \\ 00110 \\ 00001 \end{pmatrix}, \quad L = \begin{pmatrix} 11111 \\ 11111 \\ 00101 \\ 11111 \\ 00001 \end{pmatrix}, \quad Q = \begin{pmatrix} 11010 \\ 11010 \\ 11110 \\ 11010 \\ 11111 \end{pmatrix}. \quad L \cap Q = \begin{pmatrix} 11010 \\ 11010 \\ 00100 \\ 11010 \\ 00001 \end{pmatrix}.$$

$$CK_{x_1} = \{x_1, x_2, x_4\}, \quad CK_{x_2} = \{x_3\}, \quad CK_{x_3} = \{x_5\}$$

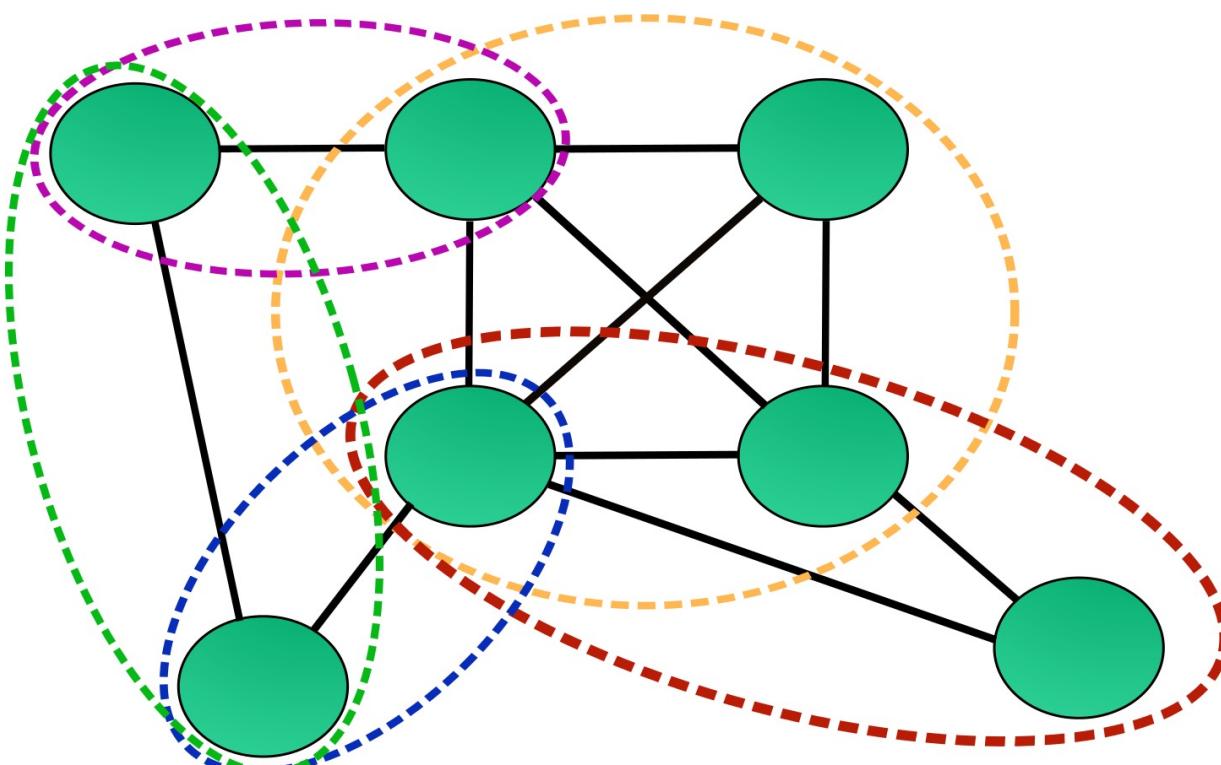
Конденсация графа

На сильных компонентах, как на вершинах, может быть построен новый граф $G^* = (X^*, F^*)$ – который называется **конденсацией** графа G . Каждая его вершина отображает множество вершин, соответствующих одной из сильных компонент исходного графа G , а дуга (x_ξ^*, x_η^*) существует в G^* тогда и только тогда, когда в графе G существует дуга (x_i, x_j) такая, что $x_i \in CKx_\xi$, а $x_j \in CKx_\eta$.



Клика

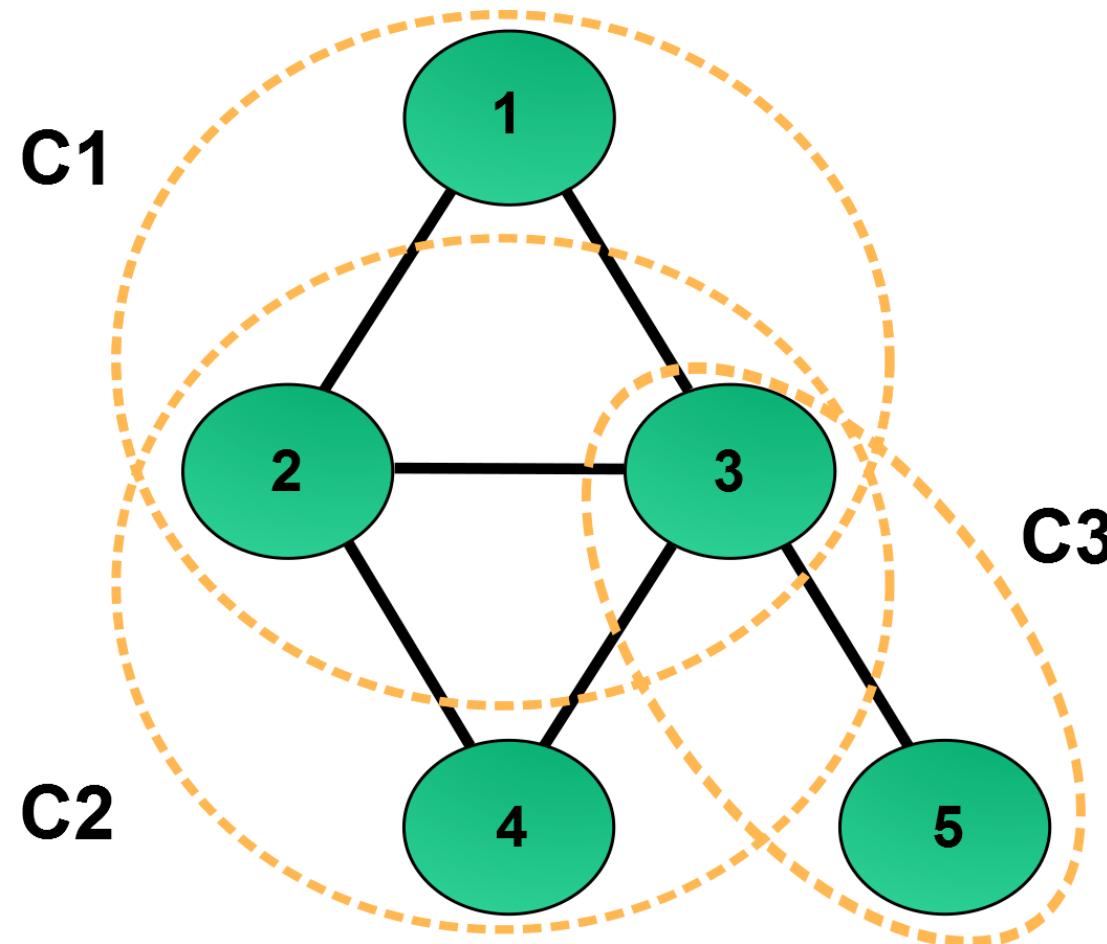
Клика (C) – это подмножество графа G , такое, что представляет собой максимальное полное множество, т. е. в графе G не существует другого полного множества, которое содержит C .



Упорядочивание в графе

- Порядок узлов в графе состоит в присвоении каждой вершине целого числа.
- Дан граф $G = (V, E)$ с n вершинами, тогда
- $\alpha = [V_1, V_2, \dots, V_n]$ — порядок графа; V_i находится перед V_j в соответствии с этим порядком, если $i < j$
- Порядок α графа $G = (V, E)$ является совершенным порядком, если все смежные вершины каждой вершины V_i , находящиеся перед V_i , согласно этому порядку, полностью связаны

Совершенный порядок

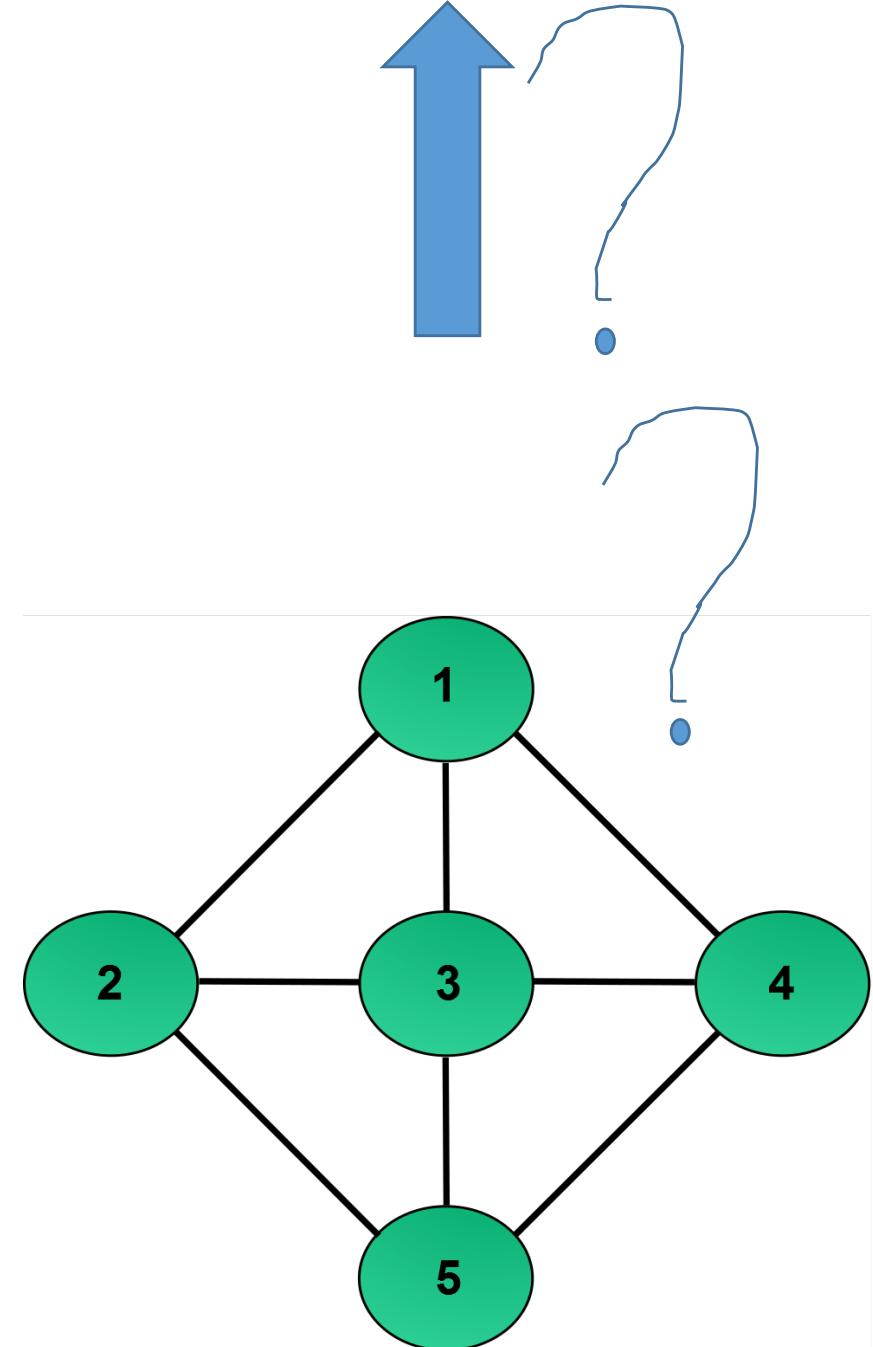


Порядок клик

- Аналогично порядку узлов мы можем определить порядок клик, $\beta = [C_1, C_2, \dots, C_p]$
- Порядок β клик имеет свойство бегущего пересечения (древесной декомпозиции), если все общие узлы каждой клики C_i с предыдущими кликами в соответствии с этим порядком содержатся в клике C_j ; C_j является родителем C_i
- Возможно, что клика имеет более одного родителя

Триангулированный граф

- Граф G называется триангулированным, если каждый простой контур длины больше трех в G имеет хорду
- Хорда — это ребро, соединяющее две вершины контура и не являющееся частью этого контура.
- Условием достижения идеального упорядочения вершин и упорядочения клик, удовлетворяющего свойству бегущего пересечения, является триангуляция графа.



Maximum cardinality search

Если граф триангулирован, следующий алгоритм гарантирует идеальное упорядочение:

1. Выберите любую вершину из V и назначьте ей номер 1.
2. ПОКА не все вершины в G пронумерованы:

Из всех непомеченных вершин выбрать ту, для которой смежная вершина имеет наибольший номер, и присвоить выбранной вершине следующий номер. Неоднозначности разрешаются произвольно.

Упорядочивание клик

- После полного упорядочивания всех вершин легко пронумеровать клики так, чтобы их порядок соответствовал свойству древесной де- композиции.
- Для этого клики нумеруются в обратном порядке.
- Пусть существует r клик, тогда клике, содержащей узел с наибольшим номером, присваивается номер r . Клика, включающая узел со следующим наибольшим номером, получает номер $r - 1$ и т.д.

Дополнение графа до триангулированного

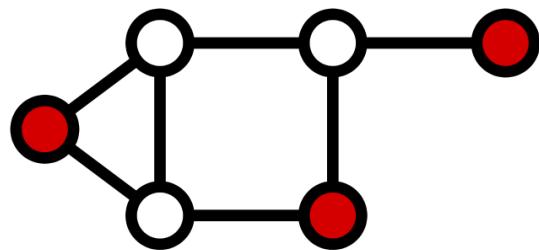
- ① Order the vertices V with maximum cardinality search:
 V_1, V_2, \dots, V_n .
- ② FOR $i = n$ TO $i = 1$
 - ① For node V_i , select all its adjacent nodes V_j such that $j > i$. Call this set of nodes A_i .
 - ② Let V_m be the node with largest number in A_i .
 - ③ Add an arc from V_i to V_k if $k > i$, $k < m$ and $V_k \notin A_i$.

Базовое и доминирующее множества

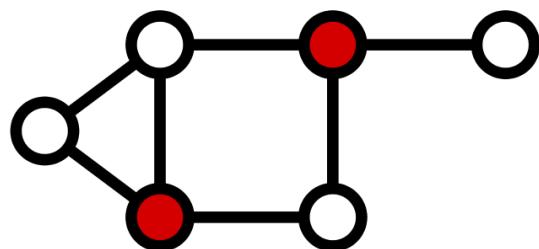
- **Базовым множеством (БМ)** графа G называется такое множество B вершин этого графа, из которых достижима любая вершина графа и не существует его подмножества, обладающего таким же свойством достижимости.
- **Доминирующим множеством (ДМ)** вершин $S \subseteq X$ графа $G=(X,F)$ называется такое множество вершин, что для каждой вершины $x_j \notin S$ существует дуга (путь длиной 1), идущая от некоторой вершины множества S к данной вершине x_j и не существует его подмножества, обладающего таким же свойством.
- Для поиска БМ/ДМ нужно в матрице L/R^1 найти такой набор строк, чтобы в каждом столбце была хотя бы одна 1. И из этого набора строк нельзя удалить ни одной строки без нарушения данного условия.

Поиск доминирующих множеств

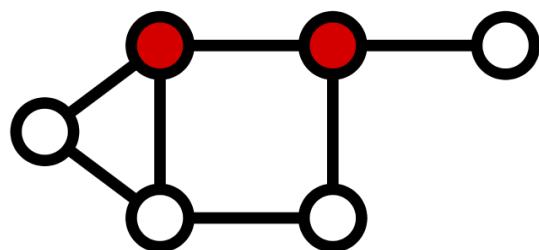
(a)



(b)



(c)

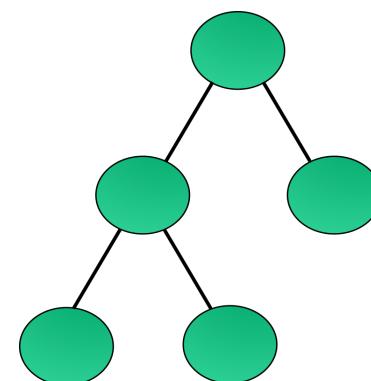


Неориентированное дерево

— это связный граф, не имеющий простых циклов.

В неориентированном дереве есть два класса вершин или узлов:

- (i) конечные узлы степени один (листья);
- (ii) внутренние узлы со степенью больше единицы



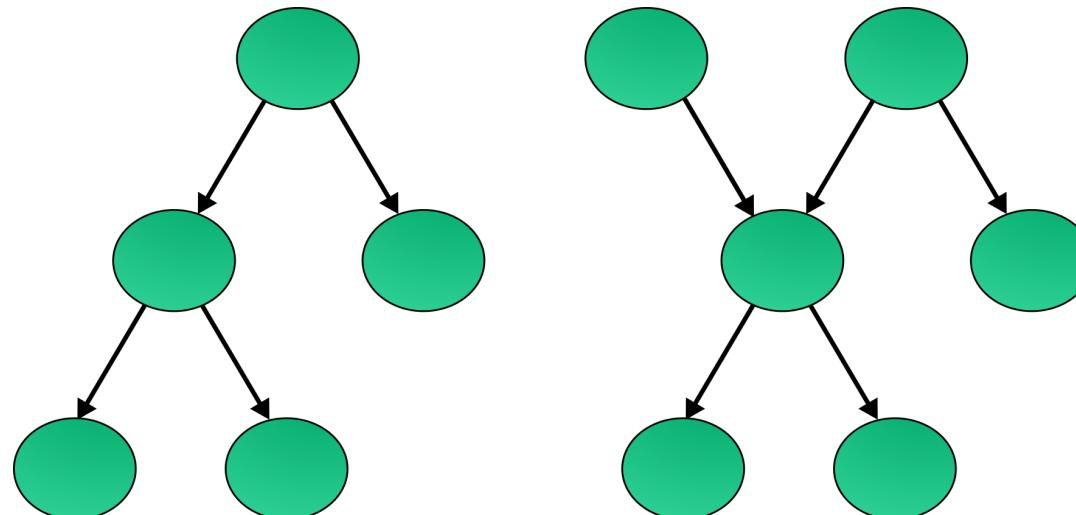
Свойства

- Между каждой парой вершин существует простой путь.
- Количество вершин, $|V|$, равно количеству ребер, $|E|$ плюс один: $|V| = |E| + 1$.
- Дерево с двумя или более вершинами имеет как минимум два листовых узла.

Направленное дерево

— это связный ориентированный граф, между каждой парой вершин которого существует не более одного пути.

- Корневое дерево имеет одну вершину с нулевой степенью входа (корневой узел), а остальные вершины имеют степень один
- Полидерево может иметь более одного узла с нулевой степенью (корнями) и некоторые узлы (ноль или более) со степенью больше единицы.



Определения

- Корень: узел с нулевой степенью входа.
- Лист: узел с нулевой степенью исхода.
- Внутренний узел: узел, степень исхода которого больше нуля.
- Родитель/Ребенок: если есть направленная дуга от A до B, A является родителем B и B является дочерним элементом A.
- Братья: два или более узла, которые имеют одного и того же родителя.
- Предки/Потомки: если существует дуга из A в B, A является предком B, а B является потомком A.
- Поддерево с корнем A: поддерево с вершиной A в качестве корня.
- Поддерево A: поддерево с дочерним элементом A в качестве корня.
- K-арное дерево: дерево, в котором каждый внутренний узел имеет не более K детей. Это обычное дерево, если каждый внутренний узел имеет K потомков.
- Двоичное дерево: дерево, в котором каждый внутренний узел имеет не более двух детей.

Центральность в графе

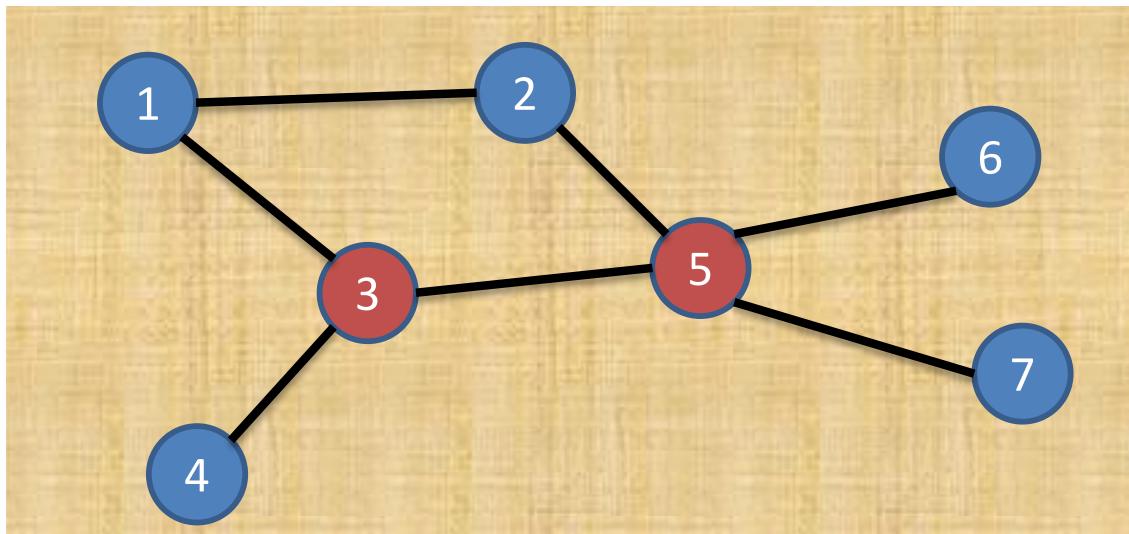
Центральность вершин в графе – это вектор, сопоставляющей каждой вершине графа некоторое число (индекс).

Наиболее распространенные индексы:

- Степенная центральность (degree centrality);
- Центральность по близости (closeness centrality);
- Центральность по посредничеству (betweenness centrality);
- Центральность по собственному вектору (eigenvector centrality);
- Центральность PageRank.

Центральность по близости

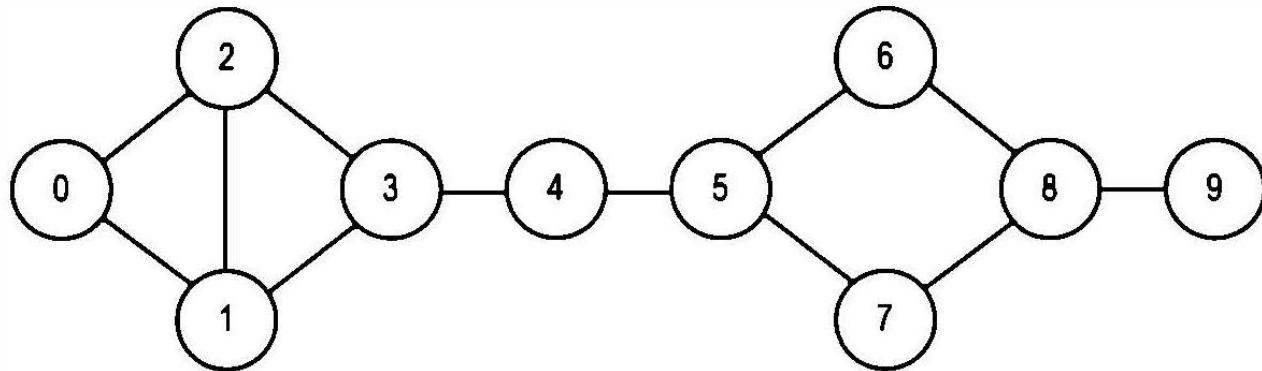
Вершина, находящаяся ближе всех к другим вершинам сети, является наиболее центральной



$$C_i = \frac{1}{\sum_j d_{ij}} \quad C_i = \sum_j \frac{1}{d_{ij}}$$

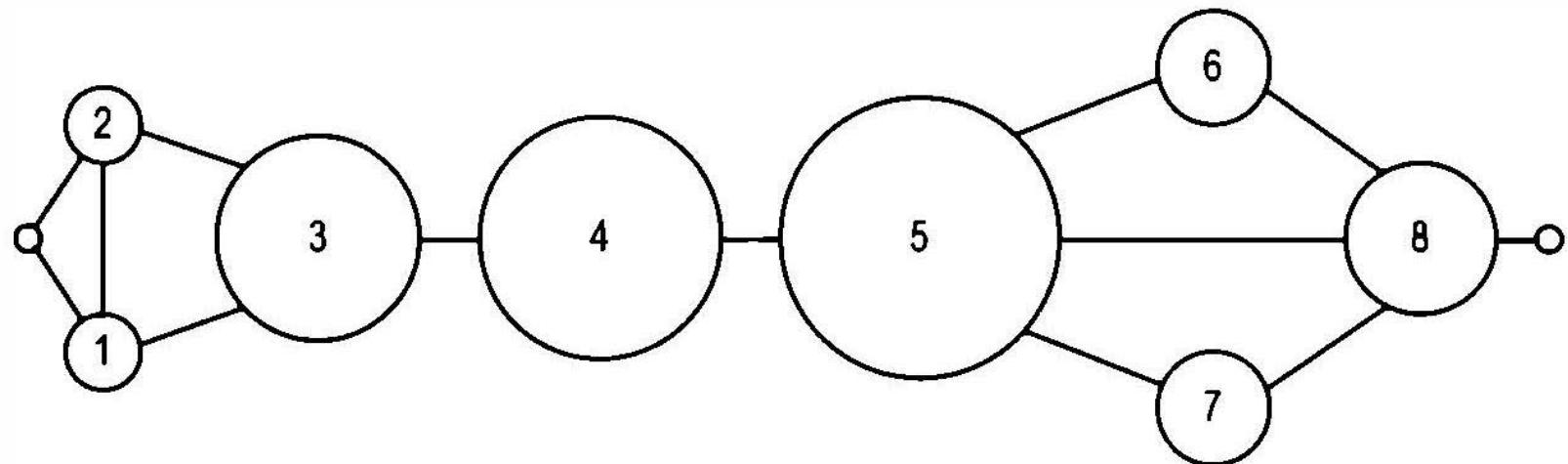
Центральность по посредничеству

Вершина, через которую проходит наибольшее число кратчайших путей, является наиболее центральной.



$$C_i = \sum_{jk} \frac{w_{jk}(i)}{w_{jk}}$$

Центральность по посредничеству



Центральность по собственному значению

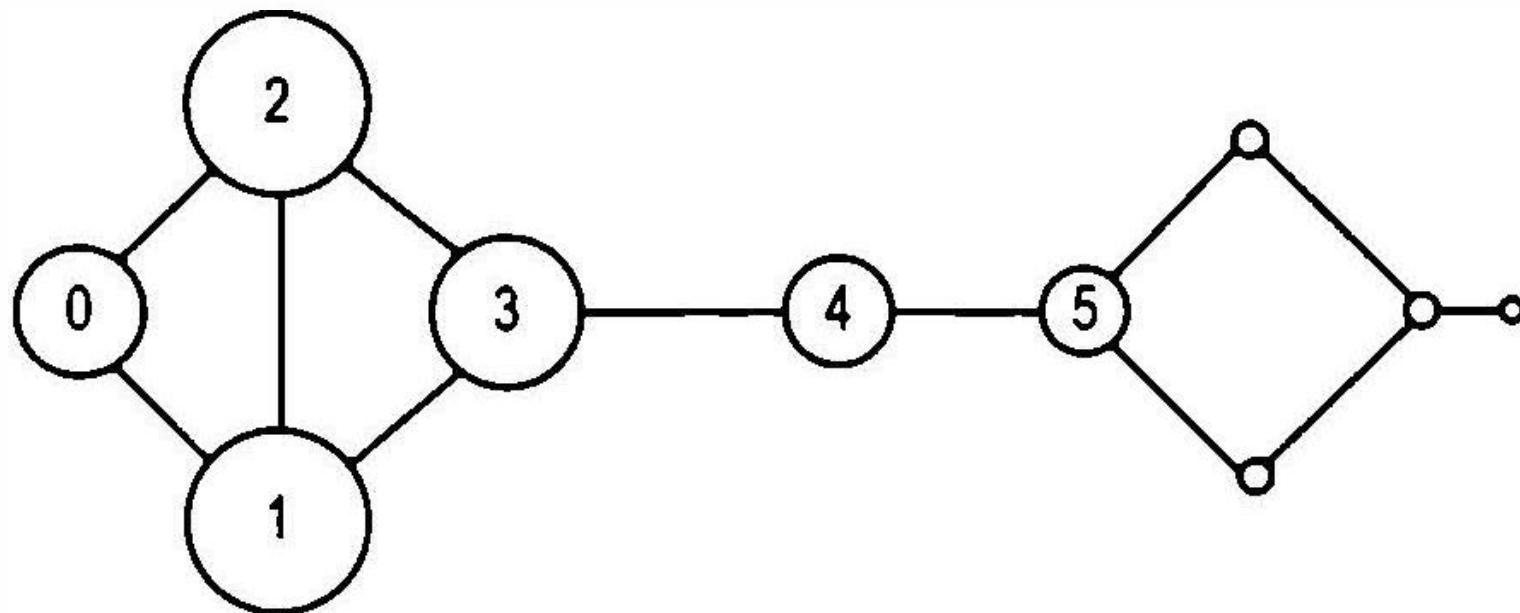
Центральность вершины i зависит от центральностей соседей вершины i .

$$x_i = \frac{1}{\lambda} \sum_{j \in F_i} x_j = \frac{1}{\lambda} \sum_j a_{ij} x_j$$

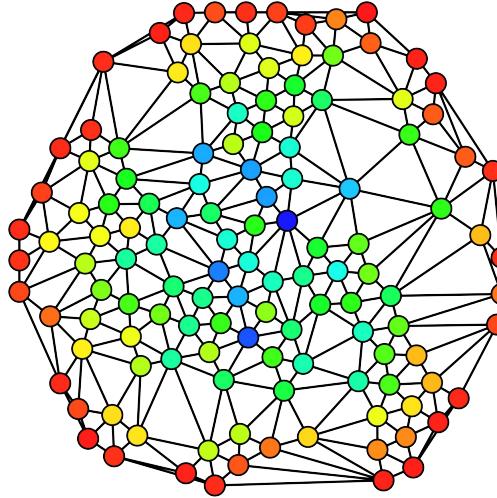
$$\lambda x = Ax$$

- Выбирается собственный вектор, соответствующий максимальному собственному значению.
- Данная центральность учитывает дальние взаимодействия.
- Наиболее центральными считаются вершины, которые сами указывают на сильные вершины.

Центральность по собственному значению



Задача



- Давайте соберем информацию о друзьях и друзьях Ваших друзей из VK по всем членам Вашей группы
- Построить граф дружбы. Все отношения дружбы между найденными профилями должны быть в графе
- Оценить центральность членов Ваше группы: по посредничеству, по близости, собственного вектора
- Посчитайте диаметр и радиус графа, найдите центральные и периферийные вершины
- Найдите клики в графе.
- Является ли график триангулированным? Если ответ Нет, то сделайте его таким и найдите клики снова.
- Визуализировать график, по возможности красиво и информативно
- Вывести имена людей с максимальными центральностями.

Определение

Графовая вероятностная модель — это вероятностная модель, в которой в виде графа представлены зависимости между случайными величинами. Вершины графа соответствуют случайным переменным, а рёбра — непосредственным вероятностным взаимосвязям между случайными величинами.

Могут работать с малыми выборками

Применение

- извлечение информации
- распознавание речи
- компьютерное зрение
- диагностика болезней
- диагностика промышленного оборудования
- оценка систем безопасности
- прогнозирование отказов

Некоторые сведения из теории вероятностей

Для независимых событий:

$$P(AB) = P(A) P(B)$$

Для зависимых событий:

$$P(AB) = P(A) P(B/A)$$

$$P(AB) = P(B) P(A/B)$$

Формула Байеса:

$$P(A/B) = \frac{P(A)P(B/A)}{P(B)}$$

Теорема Байеса для классификации

$$P(y/x_1, x_2, \dots, x_n) = \frac{P(y)P(x_1, x_2, \dots, x_n/y)}{P(x_1, x_2, \dots, x_n)}$$

$$\hat{y} = \arg \max_y P(y/x_1, x_2, \dots, x_n)$$

Наивное предположение:

$$P(x_i/y, x_1, \dots, x_{i-1}, x_{i+1}, x_n) = P(x_i/y)$$

$$P(y/x_1, x_2, \dots, x_n) = \frac{P(y) \prod_{i=1}^n P(x_i/y)}{P(x_1, x_2, \dots, x_n)}$$

$$\hat{y} = \arg \max_y P(y) \prod_{i=1}^n P(x_i/y)$$

Задача

Создайте наивный классификатор для данных из таблицы 1

Какие метрики оценки качества можно использовать?

Посчитайте эти метрики

Таблица 1

Прогноз	Температура	Влажность	Ветер	Игра
Солнечно	Высокая	Высокая	Ложь	Нет
Солнечно	Высокая	Высокая	Истина	Нет
Облачно	Высокая	Высокая	Ложь	Да
Дождь	Средняя	Высокая	Ложь	Да
Дождь	Низкая	Нормальная	Ложь	Да
Дождь	Низкая	Нормальная	Истина	Нет
Облачно	Низкая	Нормальная	Истина	Да
Солнечно	Средняя	Высокая	Ложь	Нет
Солнечно	Низкая	Нормальная	Ложь	Да
Дождь	Средняя	Нормальная	Ложь	Да
Солнечно	Средняя	Нормальная	Истина	Да
Облачно	Средняя	Высокая	Истина	Да
Облачно	Высокая	Нормальная	Ложь	Да
Дождь	Средняя	Высокая	Истина	Нет

Проблема и путь решения

Независимость признаков часто не выполняется

$$P(x_i/y, x_1, \dots, x_{i-1}, x_{i+1}, x_n) \neq P(x_i/y)$$

Путь решения - рассчитывать условные вероятности значений признака и от y и от x , которые на него влияют.

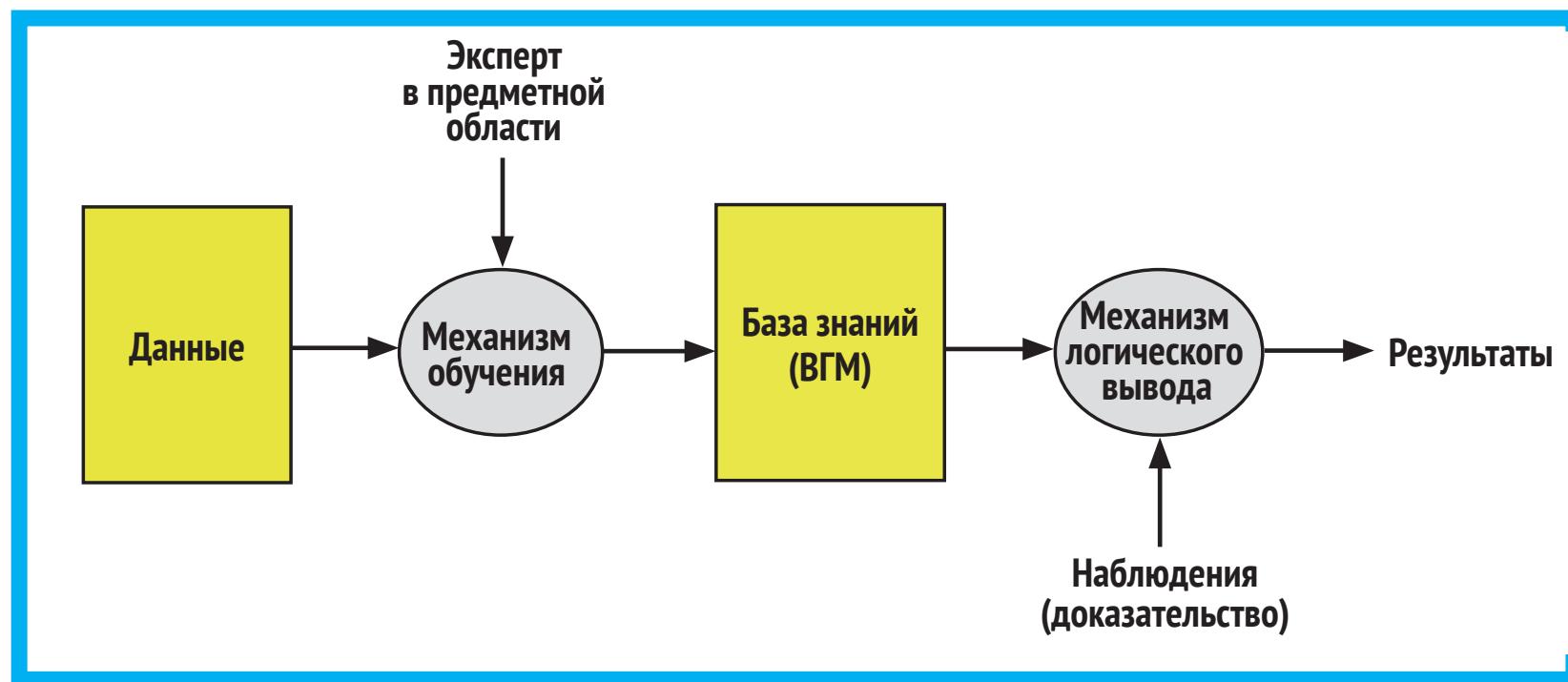
Для этого нужно построить граф. В нем:

- вершины – это признаки
- дуги/ребра отражают влияние признаков друг на друга

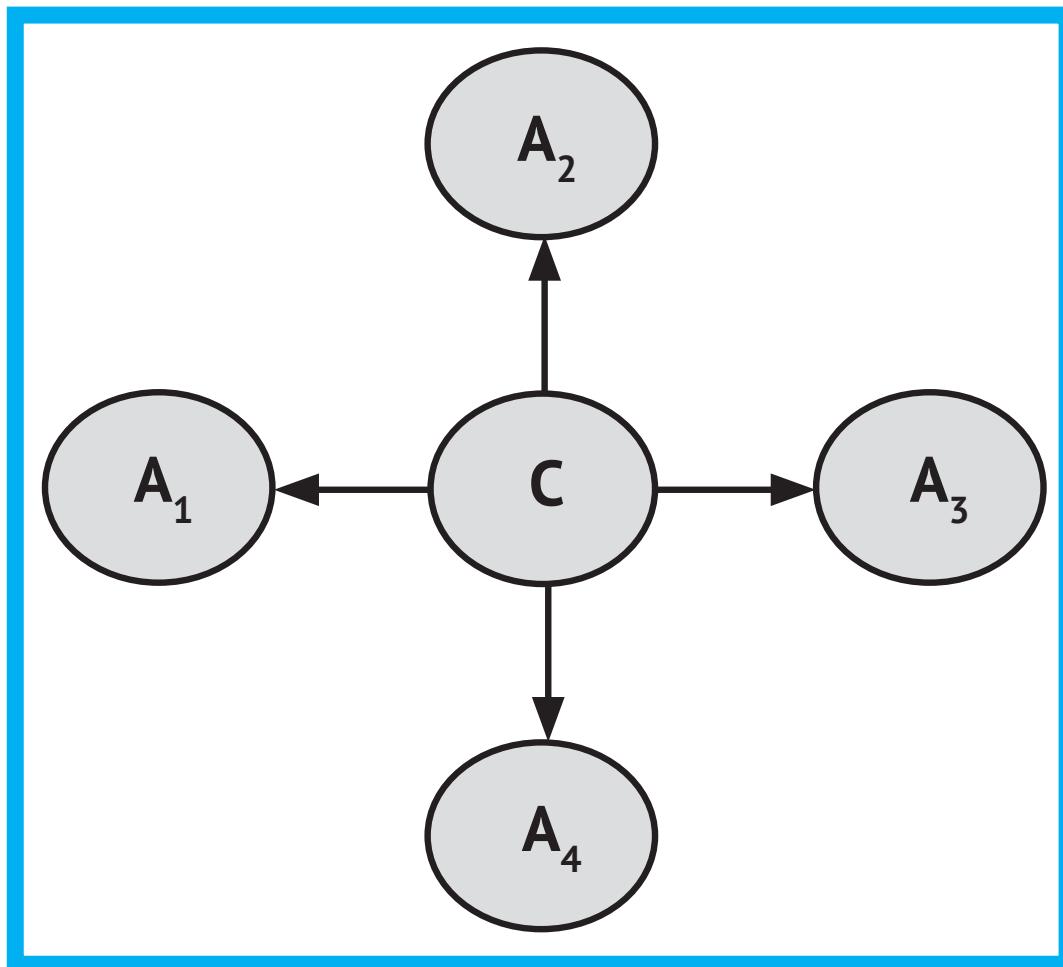
Типы моделей

Тип модели	Направленный / Ненаправленный граф	Статическая Динамическая	/	Вероятностная / Для принятия решений
Байесовские классификаторы	Напр./ Ненапр.	Статическая		Вероятностная
Марковские цепи	Напр.	Динамическая		Вероятностная
Скрытые марковские модели	Напр.	Динамическая		Вероятностная
Марковские случайные поля	Ненапр.	Статическая		Вероятностная
Байесовские сети	Напр.	Статическая		Вероятностная
Динамические байесовские сети	Напр.	Динамическая		Вероятностная
Диаграммы влияния	Напр.	Статическая		Для принятия решений
Марковские процессы принятия решений (МППР)	Напр.	Динамическая		Для принятия решений
Частично наблюдаемые МППР	Напр.	Динамическая		Для принятия решений

Представление, логический вывод и обучение

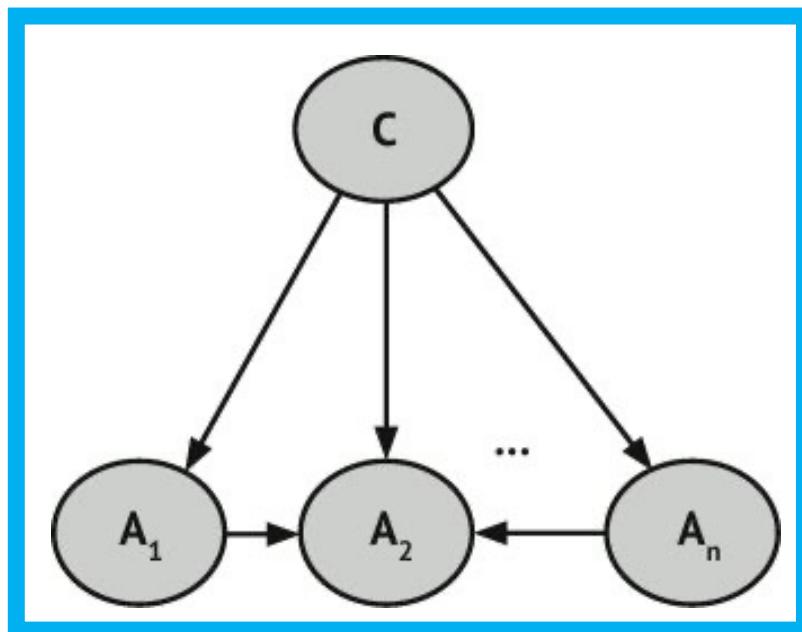


Визуализация наивного байесовского классификатора

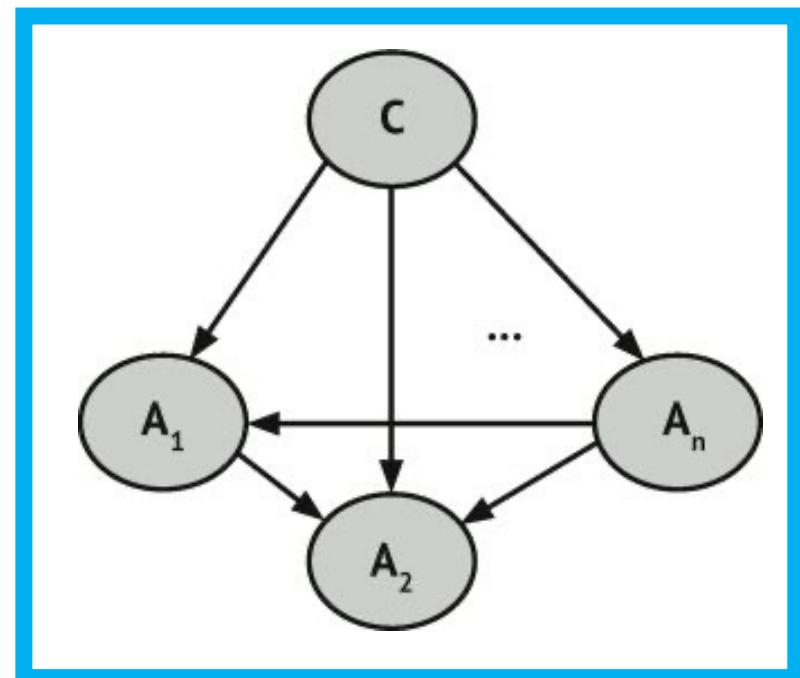


Другие классификаторы

TAN (Tree augmented Bayesian classifier - байесовский классификатор, дополненный деревом)



BAN (Bayesian Network augmented Bayesian classifier - байесовский классификатор, дополненный байесовской сетью)



Расчет вероятностей в TAN и BAN

Апостериорную вероятность для переменной класса можно получить тем же способом, что и для наивного байесовского классификатора.

Каждый атрибут зависит не только от класса, но и от других атрибутов в соответствии со структурой графа.

Необходимо учитывать условную вероятность каждого атрибута по отношению к классу и к своим родительским атрибутам:

$$P(C|A_1, A_2, \dots, A_n)$$

$$= \frac{P(C)P(A_1|C, Pa(A_1))P(A_2|C, Pa(A_2)) \dots P(A_n|C, Pa(A_n))}{P(A)}$$

Задача

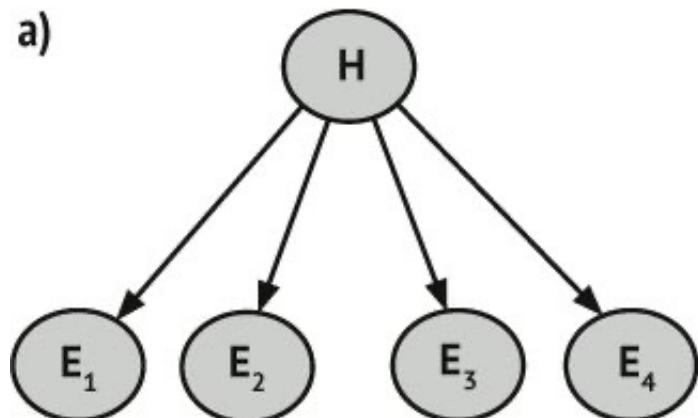
Рассмотрим модель ТАН со следующей структурой зависимостей атрибутов:

- прогноз погоды —► температура
- прогноз погоды —► влажность
- температура —► ветер

Используя набор данных таблицы прошлой задачи, составить таблицы условных вероятностей для этой модели ТАН

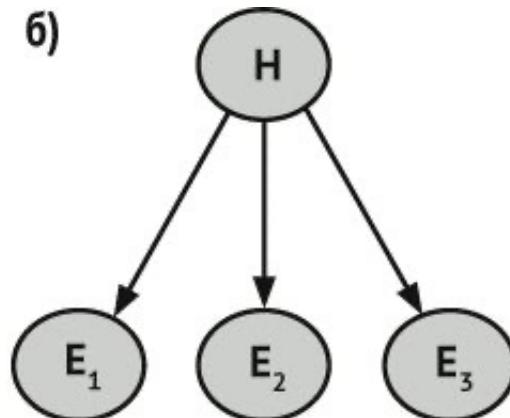
Улучшение структуры

а)



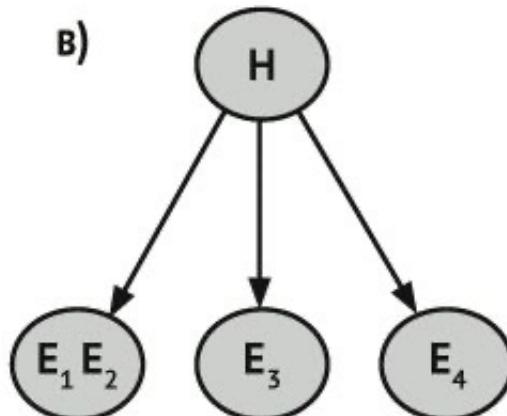
Исходная структура

б)



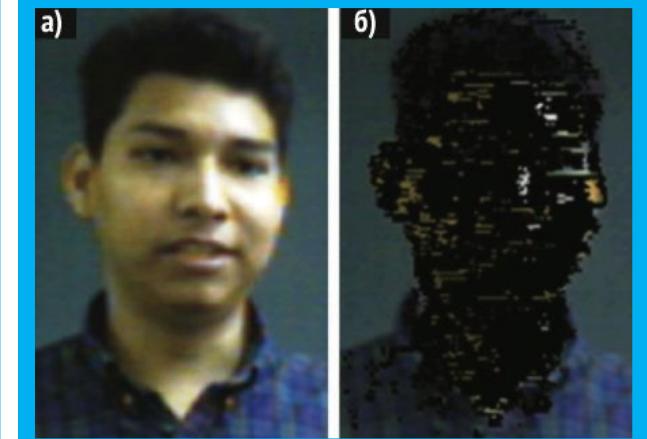
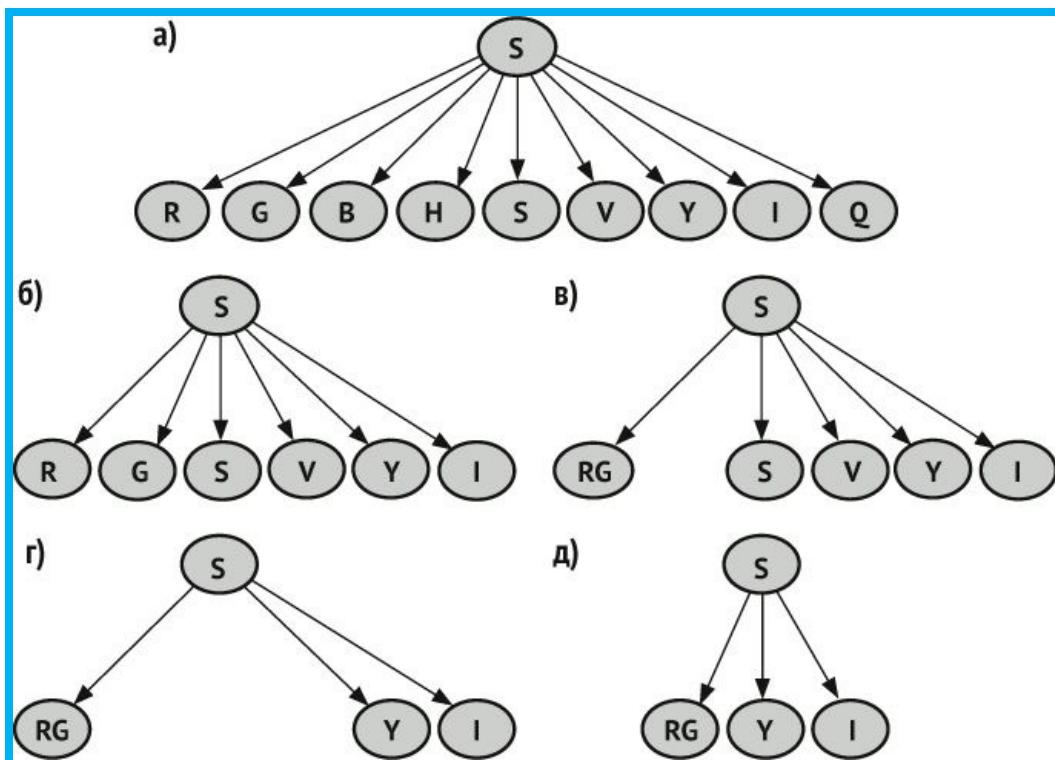
Исключение E₄

в)

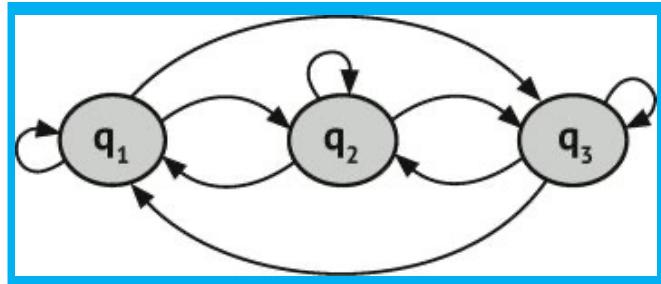


Объединение E₁, E₂

Определение кожи человека



Марковские цепи



Солнечно	Облачно	Дождь
0.2	0.5	0.3

	Солнечно	Облачно	Дождь
Солнечно	0.8	0.1	0.1
Облачно	0.2	0.6	0.2
Дождь	0.3	0.3	0.4

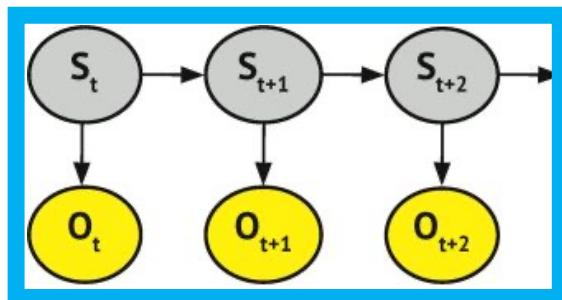
Марковская цепь позволяет ответить на вопросы:

Какова вероятность некоторой конкретной последовательности состояний?

Какова вероятность того, что цепь остается в определенном состоянии в течение некоторого интервала времени?

Каково ожидаемое время, в течение которого цепь будет оставаться в определенном состоянии?

Скрытые Марковские цепи



Распознавание жеста Стоп



Итоги

- Вероятностные графовые модели позволяют учесть экспертное мнение о взаимосвязи параметров
- Вероятностные графовые модели позволяют работать с малыми выборками
- Наивный байесовский классификатор не всегда применим в силу зависимости показателей
- Марковские цепи позволяют прогнозировать поведение динамических систем

Спасибо за внимание!

Пишите: sudakov@ws-dss.com