

# Кратчайшие пути в графе

Лекция 10

# Определения

Пусть  $G = (V, E)$  – ориентированный граф.

Поставим в соответствие каждому ребру  $e \in E$  в графе  $G$  неотрицательную стоимость  $c(e)$ .

$c: E \rightarrow R^+$  - *функция стоимости*

**Стоимость пути** определяется как сумма стоимостей ребер, образующих его.

**Задача о нахождении кратчайшего пути**

состоит в нахождении для каждой пары узлов  $(v, w)$  пути наименьшей стоимости.

*Нахождение кратчайшего пути  
из одного источника*

## *Алгоритм Дейкстры*

**Идея алгоритма Дейкстры** нахождения кратчайших путей из одной вершины во все другие состоит в следующем.

1. Строится множество  $S$ , содержащее вершины графа, кратчайшие расстояния до которых от источника известны.
2. На каждом шаге добавляется тот из оставшихся узлов, кратчайшее расстояние до которого меньше всех других оставшихся узлов.

# Алгоритм Дейкстры

Вход:

$G = (V, E)$  – ориентированный граф,

$v_0 \in V$  – источник,

$c : E \rightarrow R^+$  – функция стоимости ребер графа  $G$ .

Полагаем, что

$$c(v_i, v_j) = +\infty, \text{ если } (v_i, v_j) \notin E$$

$$c(v, v) = 0.$$

Выход:

для всех вершин  $v \in V$  наименьшая сумма стоимостей ребер из  $E$ , взятая по всем путям, идущим из  $v_0$  в  $v$ .

# Алгоритм Дейкстры

Метод:

Строим такое множество  $S \subseteq V$ , что кратчайший путь из источника в каждый узел  $v \in S$  целиком лежит в  $S$ .

Массив  $D[v]$  содержит стоимость текущего кратчайшего пути из  $v_0$  в  $v$ .

# Алгоритм Дейкстры - $O(n^2)$

{

$S \leftarrow \{v_0\};$

$D[v_0] \leftarrow 0;$

для всех  $v \in V \setminus \{v_0\}$  выполнить:  $D[v] = c(v_0, v);$

пока  $S \neq V$  выполнять:

{

выбрать узел  $w \in V \setminus S$ , для которого

$D[w]$  принимает наименьшее значение;

добавить  $w$  к  $S$ ;

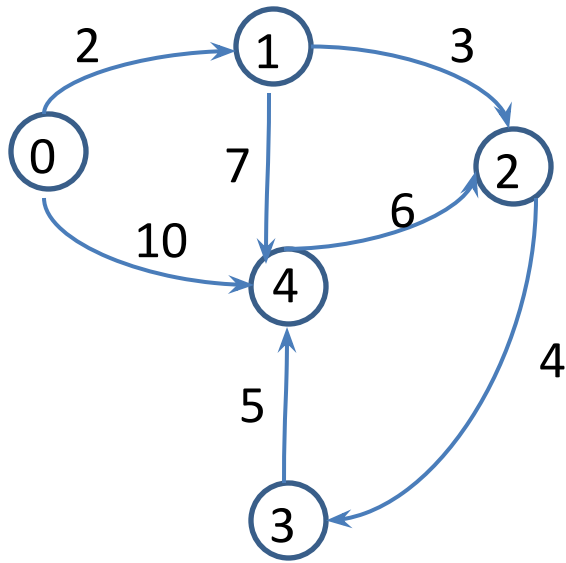
для всех  $v \in V \setminus S$  выполнить

$D[v] = \min(D[v], D[w] + c(w, v));$

}

}

# Алгоритм Дейкстры. Пример



№	S	w	D[w]	D[1]	D[2]	D[3]	D[4]
0	{0}	-	-	2	$+\infty$	$+\infty$	10
1	{0, 1}	1	2	2	5	$+\infty$	9
2	{0, 1, 2}	2	5	2	5	9	9
3	{0, 1, 2, 3}	3	9	2	5	9	9
4	{0, 1, 2, 3, 4}	4	9	2	5	9	9



# Алгоритм Беллмана-Форда

## Задача:

Для заданного взвешенного графа  $G=(V, E)$  найти кратчайшие пути из заданной вершины  $s$  до всех остальных вершин.

В случае, когда в графе  $G$  содержатся отрицательные циклы, достижимые из  $s$ , сообщить, что кратчайших путей не существует.

# Алгоритм Беллмана-Форда

Количество путей длины  $k$  рёбер можно найти с помощью метода *динамического программирования*.

Пусть  $d[k][u]$  — количество путей длины  $k$  рёбер, заканчивающихся в вершине  $u$ . Тогда

$$d[k][u] = \sum_{v:(v,u) \in E} d[k-1][v]$$

Аналогично посчитаем *пути кратчайшей длины*.

Пусть  $s$  — стартовая вершина. Тогда

$$d[k][u] = \min_{v:(v,u) \in E} (d[k-1][v] + c(v, u))$$

при этом  $d[0][s] = 0$ ,  $d[0][u] = \infty$ .

## Алгоритм Беллмана-Форда

*bool* FordBellman(*s*):

for  $v \in V$

$d[v] \leftarrow \infty$

$d[s] \leftarrow 0$

for  $i \leftarrow 1$  to  $|V|-1$

    for  $(u, v) \in E$

        if  $d[v] > d[u] + c(u, v)$      //  $c(u, v)$  — вес ребра  $uv$

$d[v] \leftarrow d[u] + c(u, v)$

$p[v] \leftarrow u$              //  $u$  — это предок  $v$

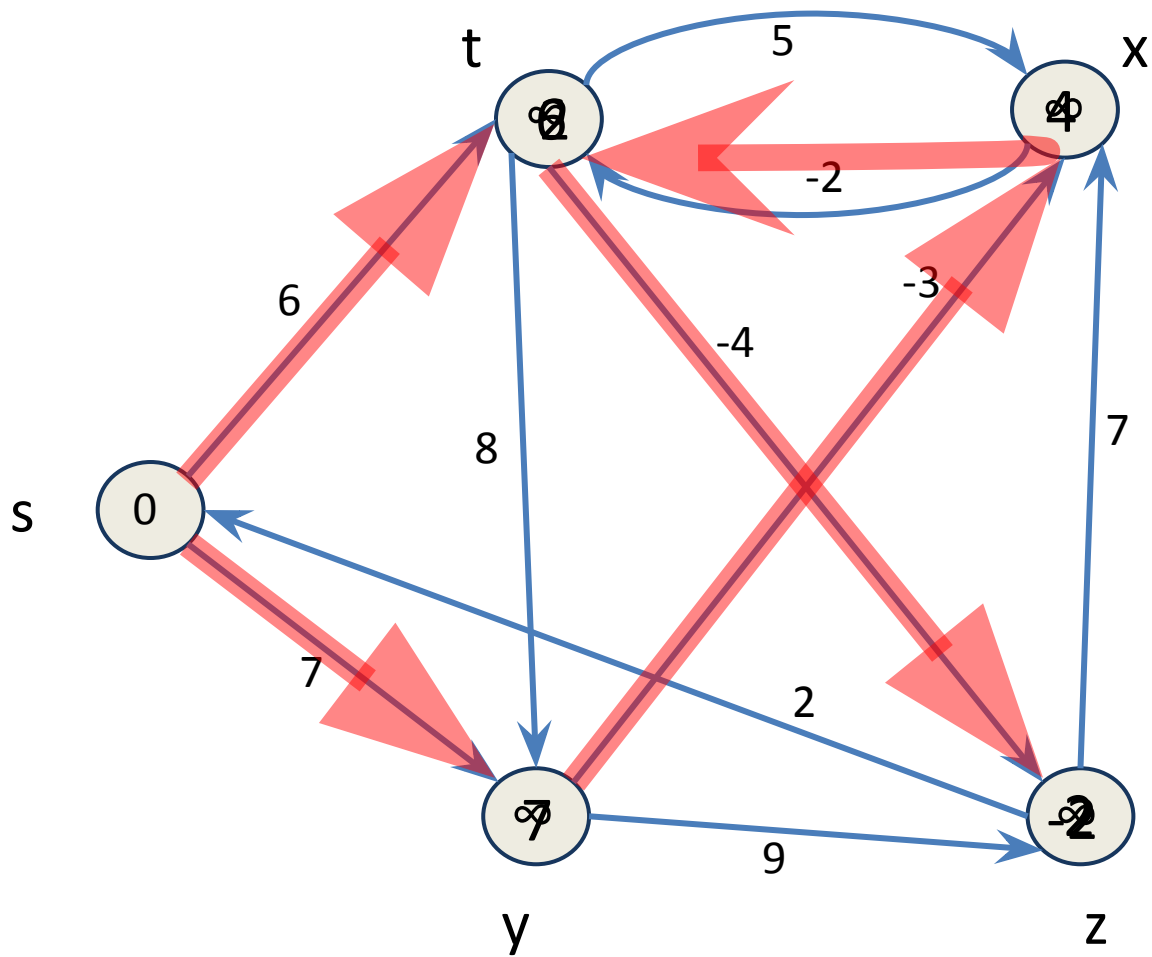
for  $(u, v) \in E$

    if  $d[v] > d[u] + c(u, v)$

        return *false*             // цикл отрицательного веса

return *true*

# Алгоритм Беллмана-Форда. Пример



## Алгоритм Беллмана-Форда

### Оценка сложности алгоритма:

- Инициализация занимает  $\Theta(V)$  времени,
- каждый из  $|V| - 1$  проходов требует  $\Theta(E)$  времени,
- обход по всем ребрам для проверки наличия отрицательного цикла занимает  $O(E)$  времени.

Значит, алгоритм Беллмана-Форда работает за  $O(VE)$  времени.

**Нахождение кратчайших путей  
между всеми парами вершин**

# Алгоритм Флойда-Уоршола

Пусть  $G = (V, E)$  – ориентированный граф,

$c : E \rightarrow R^+$  – функция стоимости ребер графа  $G$ .

Строим матрицу стоимостей:

$$M[i, j] = \begin{cases} c(i, j), & \text{если ребро } (i, j) \in E \\ +\infty, & \text{если ребро } (i, j) \notin E \\ 0, & \text{если } i = j \end{cases}$$

Обозначим через  $d[i, j]$  матрицу кратчайших путей между всеми вершинами.

Вершины занумеруем числами от 1 до  $n$ .

## Алгоритм Флойда-Уоршоппа

Обозначим через  $d_{ij}^{(k)}$  стоимость кратчайшего пути из вершины с номером  $i$  в вершину с номером  $j$  с промежуточными вершинами из множества  $\{1, 2, \dots, k\}$ .

$$d_{ij}^{(k)} = \begin{cases} M[i, j], & \text{если } k = 0, \\ \min(d_{ij}^{(k-1)}, d_{ik}^{(k-1)} + d_{kj}^{(k-1)}), & \text{если } k \geq 1 \end{cases}$$

$D^{(n)}$  содержит искомое решение



## *Алгоритм Флойда-Уоршола*

Floyd-Warshall(M, n)

{

$D^{(0)} \leftarrow M;$

for k  $\leftarrow$  1 to n do

  for i  $\leftarrow$  1 to n do

    for j  $\leftarrow$  1 to n do

$d_{ij}^{(k)} \leftarrow \min(d_{ij}^{(k-1)}, d_{ik}^{(k-1)} + d_{kj}^{(k-1)});$

return  $D^{(n)};$

}

# Транзитивное замыкание графа

Пусть  $G = (V, E)$  ориентированный граф.

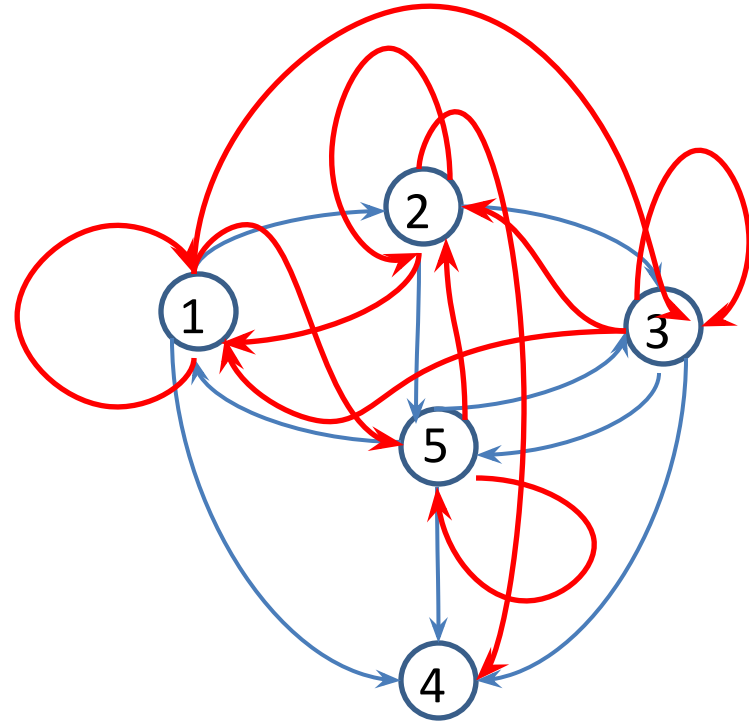
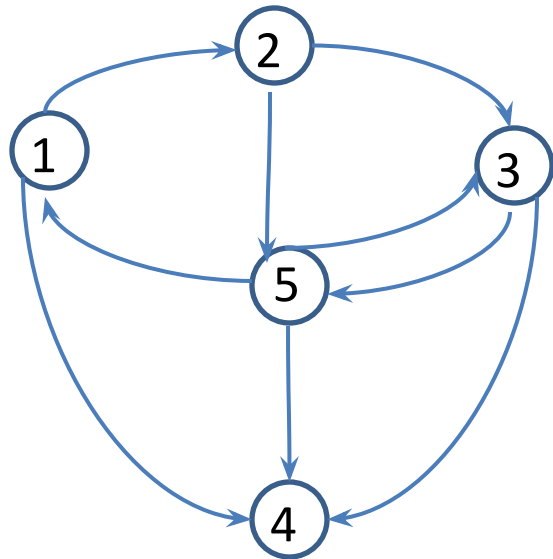
**Транзитивным замыканием** графа  $G$  называется граф  $G' = (V, E')$ , в котором из вершины  $v$  в вершину  $w$  идет дуга  $\Leftrightarrow$  существует путь из вершины  $v$  в вершину  $w$  в графе  $G$ .

$E'$ :

$$(a, b) \in E \ \& \ (b, c) \in E \Rightarrow$$

$$(a, b) \in E' \ \& \ (b, c) \in E' \ \& \ (a, c) \in E'$$

# Построение транзитивного замыкания графа. Пример



# Построение транзитивного замыкания графа

Обозначим через  $t_{ij}^{(k)}$  наличие пути из вершины с номером  $i$  в вершину с номером  $j$  с промежуточными вершинами из множества  $\{1, 2, \dots, k\}$ .  $M$  – матрица смежности графа  $G$ .

$$t_{ij}^{(k)} = \begin{cases} M[i, j], & \text{если } k = 0, \\ t_{ij}^{(k-1)} \vee (t_{ik}^{(k-1)} \wedge t_{kj}^{(k-1)}), & \text{если } k \geq 1 \end{cases}$$

$T^{(n)}$  содержит искомое решение.

## Алгоритм построения транзитивного замыкания графа

**процедура** Транзитивное\_замыкание ( $M, n$ )

$T \leftarrow M$

для всех  $k$  от 1 до  $n$  выполнить

    для всех  $i$  от 1 до  $n$  выполнить

        для всех  $j$  от 1 до  $n$  выполнить

            если  $t_{ik} = 1$  и  $t_{kj} = 1$  то

$t_{ij} \leftarrow 1$

            конец цикла

        конец цикла

    конец цикла

выдать  $T$

конец процедуры