

Алгоритмы и структуры данных

Лекция 5: Пути в графах

А. Куликов

Академия современного программирования

План лекции

- 1 Поиск в ширину
- 2 Алгоритм Дейкстры
- 3 Кратчайшие пути при наличии рёбер отрицательного веса
- 4 Упражнения

План лекции

- 1 Поиск в ширину
- 2 Алгоритм Дейкстры
- 3 Кратчайшие пути при наличии рёбер отрицательного веса
- 4 Упражнения

Поиск в ширину

Поиск в ширину

- **Расстоянием** между двумя вершинами графа без весов (ориентированного или нет) называется длина кратчайшего пути из одной вершины в другую.
- Поиск в ширину обходит вершины графа в порядке увеличения расстояний от начальной вершины.
- Поиск в ширину строит дерево кратчайших расстояний от начальных вершин.
- Время работы алгоритма линейно.

Поиск в ширину

BFS(G, s)

```
1  ▷ Вход: граф  $G(V, E)$  (ориентированный или нет), вершина  $s \in V$ 
2  ▷ Выход: для всех вершин  $u$ , достижимых из  $s$ ,
    $dist[u]$  будет равно расстоянию от  $s$  до  $u$ 
3  for всех вершин  $v \in V$ 
4      do  $dist[v] \leftarrow \infty$ 
5   $dist[s] \leftarrow 0$ 
6   $Q \leftarrow \{s\}$ 
7  while  $Q \neq \emptyset$ 
8      do  $u \leftarrow EJECT(Q)$ 
9         for всех рёбер  $(u, v) \in E$ 
10            do if  $dist[v] = \infty$ 
11                then INJECT( $Q, v$ )
12                    $dist[v] \leftarrow dist[u] + 1$ 
```

План лекции

- 1 Поиск в ширину
- 2 Алгоритм Дейкстры
- 3 Кратчайшие пути при наличии рёбер отрицательного веса
- 4 Упражнения

Очередь с приоритетами

Алгоритм Дейкстры

Алгоритм Дейкстры находит кратчайшие расстояния от заданной вершины в графе с произвольными положительными весами на рёбрах.

Определение

Очередь с приоритетами — структура данных, хранящая записи типа (ключ, значение) и поддерживающая следующие операции:

INSERT Вставляет новый элемент в очередь.

DECREASE-KEY Уведомляет очередь о том, что ключ данной записи был уменьшен.

DELETE-MIN Извлекает из очереди запись с минимальным ключом и возвращает её.

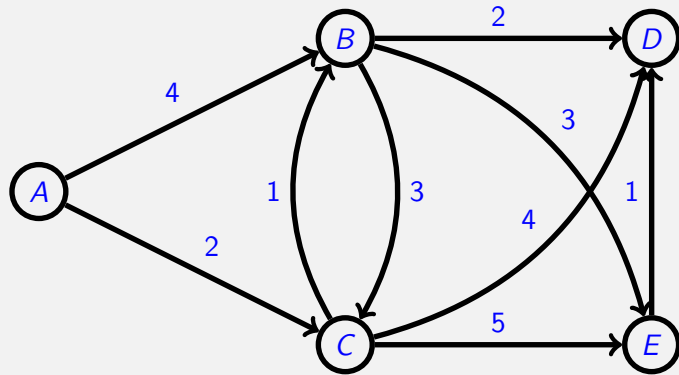
MAKE-QUEUE Создает очередь по заданному множеству записей.

Алгоритм Дейкстры

DIJKSTRA(G, s)

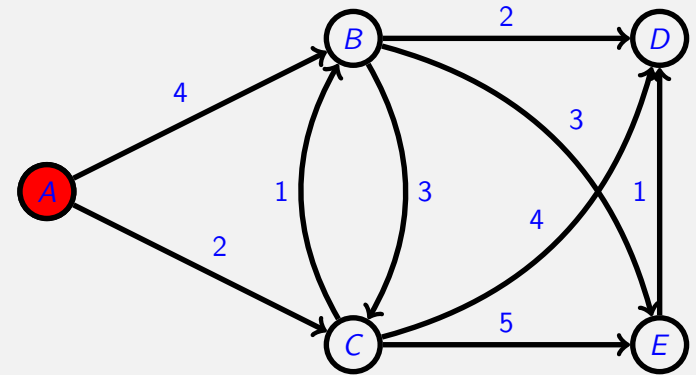
```
1  for всех вершин  $v \in V$ 
2      do  $dist[v] \leftarrow \infty$ 
3          $prev[v] \leftarrow NIL$ 
4   $dist[s] \leftarrow 0$ 
5   $H \leftarrow MAKE-QUEUE(V)$ 
6  while  $H \neq \emptyset$ 
7      do  $u \leftarrow DELETE-MIN(H)$ 
8         for всех рёбер  $(u, v) \in E$ 
9             do if  $dist[v] > dist[u] + l(u, v)$ 
10                 then
11                      $dist[v] \leftarrow dist[u] + l(u, v)$ 
12                      $prev[v] \leftarrow u$ 
13                      $DECREASE-KEY(H, v)$ 
```

Пример работы алгоритма Дейкстры



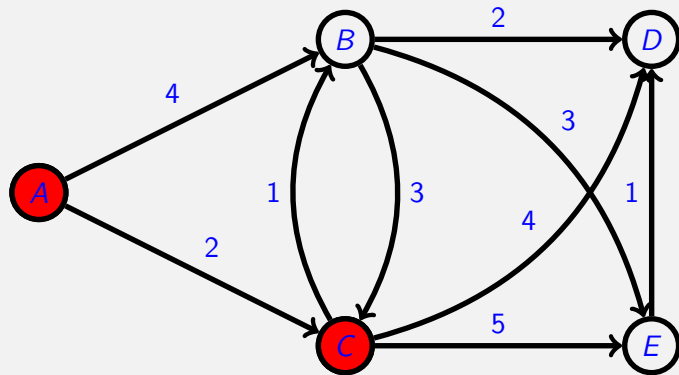
	A	B	C	D	E
<i>dist</i>	∞	∞	∞	∞	∞
<i>prev</i>	NIL	NIL	NIL	NIL	NIL

Пример работы алгоритма Дейкстры



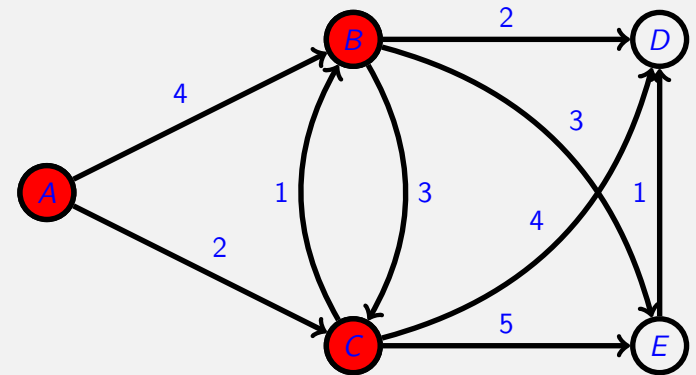
	A	B	C	D	E
<i>dist</i>	0	4	2	∞	∞
<i>prev</i>	NIL	A	A	NIL	NIL

Пример работы алгоритма Дейкстры



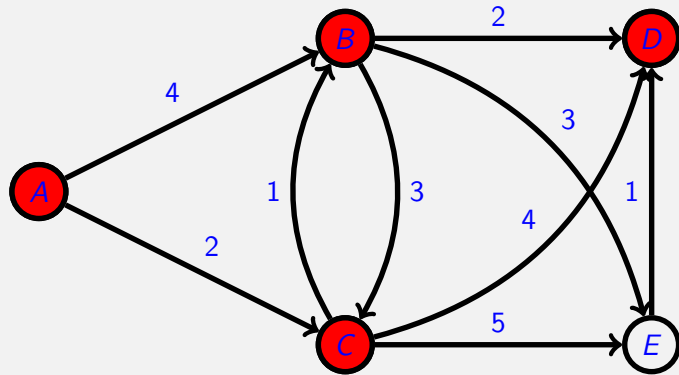
	A	B	C	D	E
<i>dist</i>	0	3	2	6	7
<i>prev</i>	NIL	C	A	C	C

Пример работы алгоритма Дейкстры



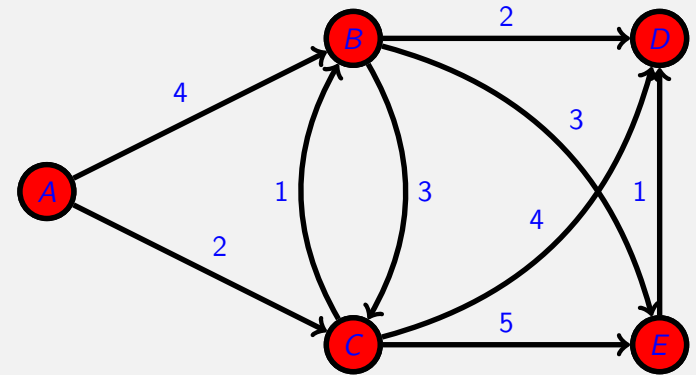
	A	B	C	D	E
<i>dist</i>	0	3	2	5	6
<i>prev</i>	NIL	C	A	B	B

Пример работы алгоритма Дейкстры



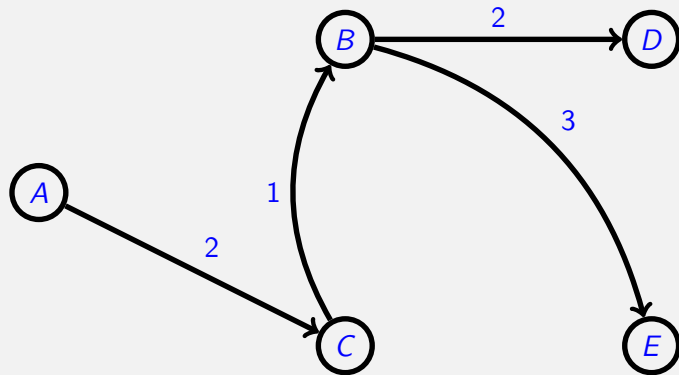
	A	B	C	D	E
<i>dist</i>	0	3	2	5	6
<i>prev</i>	NIL	C	A	B	B

Пример работы алгоритма Дейкстры



	A	B	C	D	E
<i>dist</i>	0	3	2	5	6
<i>prev</i>	NIL	C	A	B	B

Пример работы алгоритма Дейкстры



	A	B	C	D	E
<i>dist</i>	0	3	2	5	6
<i>prev</i>	NIL	C	A	B	B

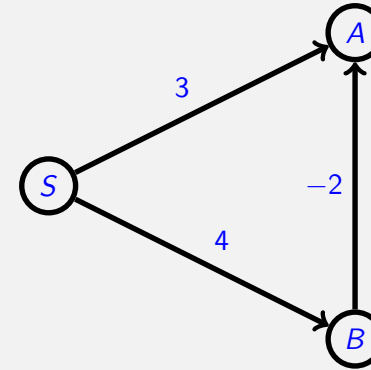
Время работы

реализация	DELETE-MIN	INSERT DECREASE-KEY	$ V \times \text{DELETE-MIN} +$ $(V + E) \times \text{INSERT}$
массив	$ V $	1	$ V ^2$
двоичная куча	$\log V $	$\log V $	$(V + E) \log V $
d -ичная куча	$\frac{d \log V }{\log d}$	$\frac{\log V }{\log d}$	$(V \cdot d + E) \frac{\log V }{\log d}$
Фиб. куча	$O(\log V)$	$O(1)$	$O(V \log V + E)$

План лекции

- 1 Поиск в ширину
- 2 Алгоритм Дейкстры
- 3 Кратчайшие пути при наличии рёбер отрицательного веса
- 4 Упражнения

Простейший пример графа с рёбрами отрицательно веса, на котором сломается алгоритм Дейкстры



Рёбра отрицательно веса

Рёбра отрицательно веса

- Алгоритм Дейкстры использует тот факт, что в графе с неотрицательными весами кратчайший путь из s в v проходит только через вершины, которые ближе к s .
- В ходе работы поддерживается важный инвариант: $dist[v]$ является оценкой сверху на длину кратчайшего пути до v .
- В процессе работы расстояния улучшаются вдоль рёбер:

$UPDATE((u, v) \in E)$

$$1 \quad dist[v] = \min\{dist[v], dist[u] + l(u, v)\}$$

- Два важных свойства процедуры $UPDATE$:

- 1 Она правильно устанавливает расстояние до v в случае, если u является предпоследней вершиной на кратчайшем пути из s в v и $dist[u]$ уже вычислено правильно.
- 2 $dist[v]$ никогда не становится меньше, чем нужно. В частности, лишние вызовы процедуры ничего испортить не могут.

Порядок вызовов UPDATE

Порядок вызовов UPDATE

- Алгоритм Дейкстры можно рассматривать просто как последовательность вызовов процедуры $UPDATE$.
- Как мы уже увидели, такая последовательность не работает для графов с рёбрами отрицательного веса.
- Есть, однако, последовательность вызовов процедуры $UPDATE$, которая для данной конкретной вершины верно установит кратчайшее расстояние.
- Рассмотрим кратчайший путь из s в t : $s \rightarrow u_1 \rightarrow \dots \rightarrow u_k \rightarrow t$.
- Количество рёбер в таком пути не более $|V| - 1$.
- Если $UPDATE$ вызывается для рёбер $(s, u_1), (u_1, u_2), \dots, (u_k, t)$ **именно в таком порядке, но не обязательно подряд**, тогда расстояние до t будет вычислено верно.
- Как же найти правильный порядок?
- Просто вызовем процедуру для каждого ребра $|V| - 1$ раз!

Алгоритм Беллмана-Форда

SHORTEST-PATHS(G, s)

```
1  ▷ Вход: ориентированный граф  $G(V, E)$  (без циклов
   отрицательного веса), вершина  $s \in V$ 
2  for всех вершин  $v \in V$ 
3      do  $dist[v] \leftarrow \infty$ 
4       $prev[v] \leftarrow NIL$ 
5   $dist[s] \leftarrow 0$ 
6  repeat  $|V| - 1$  раз
7      do for всех рёбер  $e \in E$ 
8          do UPDATE( $e$ )
```

Циклы отрицательного веса

Циклы отрицательного веса

- Если в графе есть цикл отрицательного веса, то кратчайшие пути по крайней мере до некоторых вершин не определены.
- Несложно определить, есть ли такой цикл: вместо $|V| - 1$ итерации проведём на одну больше; цикл отрицательного веса есть тогда и только тогда, когда за последнюю итерацию какое-нибудь $dist$ уменьшится.

Ациклические графы

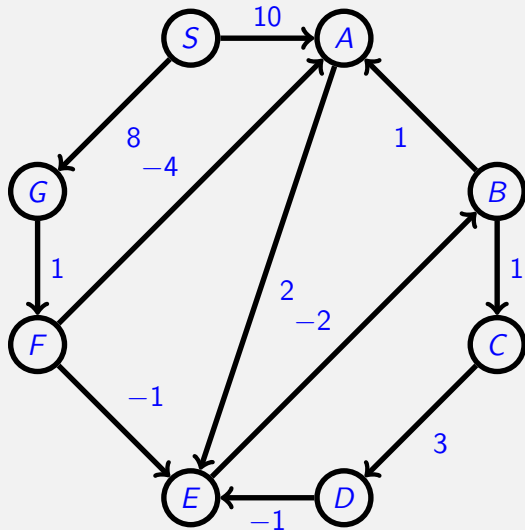
Ациклические графы

- Если два типа графов, которые автоматически исключают наличие циклов отрицательного веса, а именно: графы без рёбер отрицательного веса и графы без циклов.
- Для ациклического графа задача решается легко: найдем топологическую сортировку и будем вызывать UPDATE в соответствии с найденным порядком.
- Полученный алгоритм, таким образом, имеет линейное время работы.

План лекции

- 1 Поиск в ширину
- 2 Алгоритм Дейкстры
- 3 Кратчайшие пути при наличии рёбер отрицательного веса
- 4 Упражнения

Упражнения



Применить алгоритм Беллмана-Форда.

Упражнения

Упражнения

- Постройте алгоритм, проверяющий за время $O(|V|^3)$, есть ли в во входном неориентированном графе без весов простой цикл длины 4.
- Постройте линейный по времени алгоритм, находящий по данному неориентированному графу без весов количество кратчайших путей между двумя заданными вершинами.
- Постройте линейный алгоритм, проверяющий по данному взвешенному ориентированному графу и дереву на его вершинах, является ли это дерево деревом кратчайших путей от заданной вершины.

Упражнения

Упражнения

- Профессор предлагает такой метод нахождения кратчайшего пути между двумя заданными вершинами в графе с рёбрами отрицательного веса: прибавим ко всем рёбрам достаточно большое число, после чего запустим алгоритм Дейкстры. Прав ли он?
- Постройте алгоритм, работающий не более $O(|V|^3)$ и находящий длину самого короткого цикла в ориентированном графе с положительными весами.
- Постройте алгоритм, работающий не более $O(|V|^2)$ и находящий длину самого короткого цикла, содержащего заданное ребро, в ориентированном графе с положительными весами.