Най-кратък път в Граф

Лекция 11 по СДА, Софтуерно Инженерство
Зимен семестър 2018-2019г
д-р Милен Чечев

Това не го ли знаем вече?

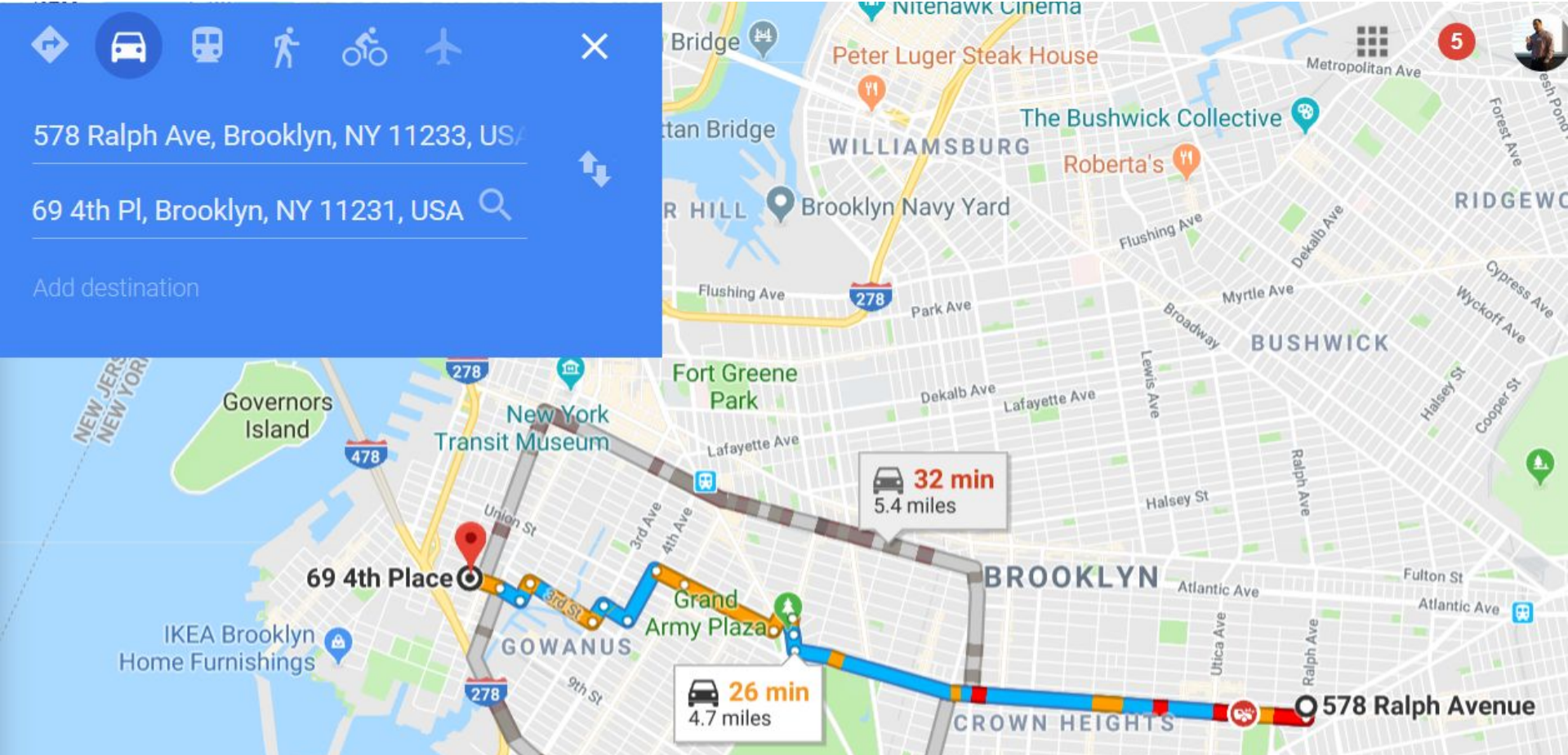
Най-кратък път в граф без тегла по ребрата се намира с BFS!

https://visualgo.net/en/sssp

Днес ще разглеждаме по сложен тип граф - граф на който по ребрата му има тегла.

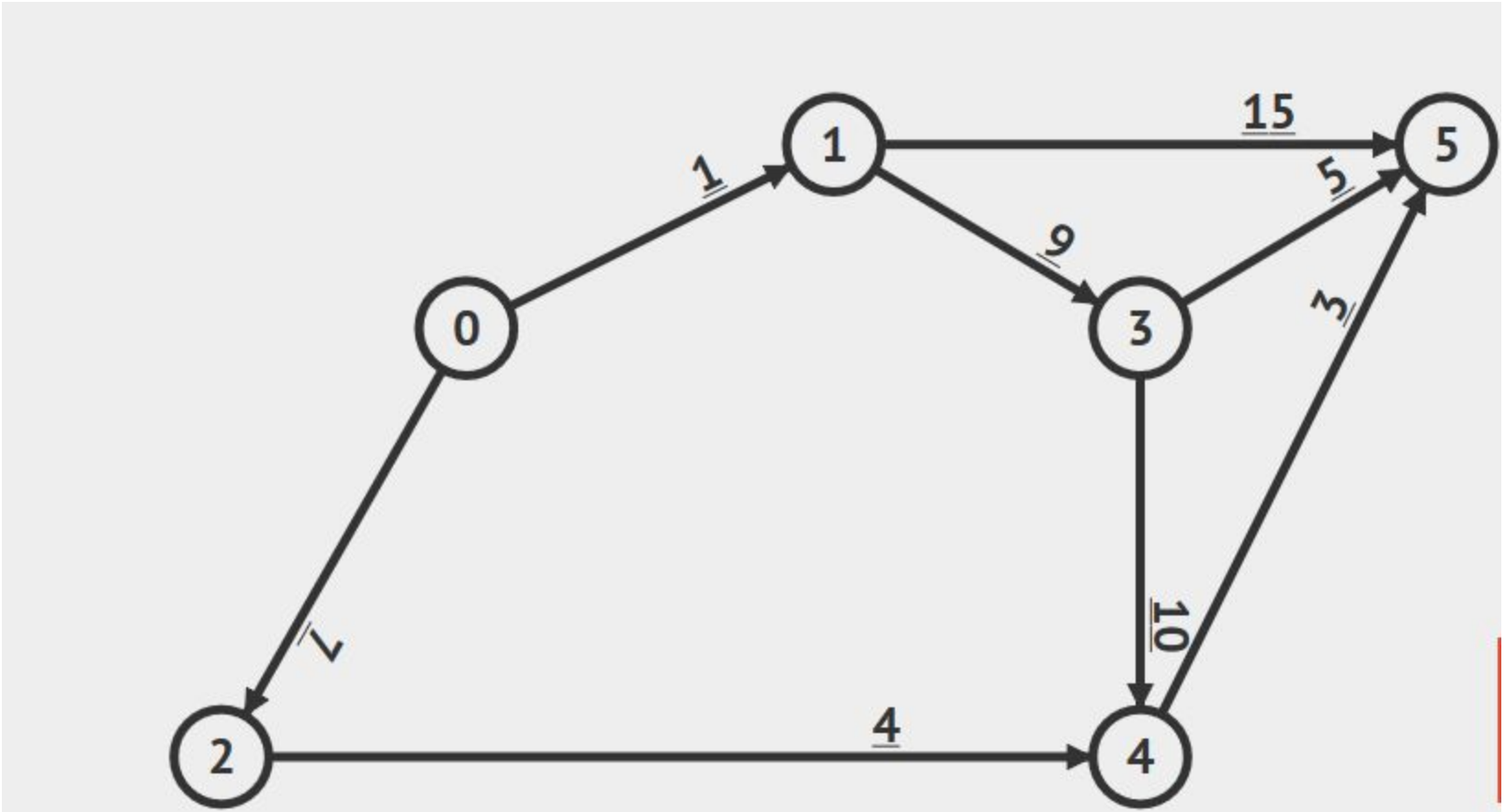
Най-къс път от възел X до възел Y е път, който започва от X и свършва в Y и сумата от теглата на ребрата по които се минава е минимална.

Пример


578 Ralph Ave, Brooklyn, NY 11233, US
69 4th Pl, Brooklyn, NY 11231, USA
Add destination
Nitehawk Cinema
Peter Luger Steak House
The Bushwick Collective
Roberta's
WILLIAMSBURG
Brooklyn Navy Yard
Metropolitan Ave
Flushing Ave
Park Ave
Myrtle Ave
Broadway
BUSHWICK
Dekalb Ave
Lafayette Ave
Halsey St
Governors Island
Fort Greene Park
New York Transit Museum
32 min
5.4 miles
69 4th Place
Union St
3rd Ave
4th Ave
3rd St
Grand Army Plaza
IKEA Brooklyn Home Furnishings
GOWANUS
9th St
26 min
4.7 miles
BROOKLYN
Atlantic Ave
Fulton St
Utica Ave
Ralph Ave
CROWN HEIGHTS
578 Ralph Avenue

Задачи

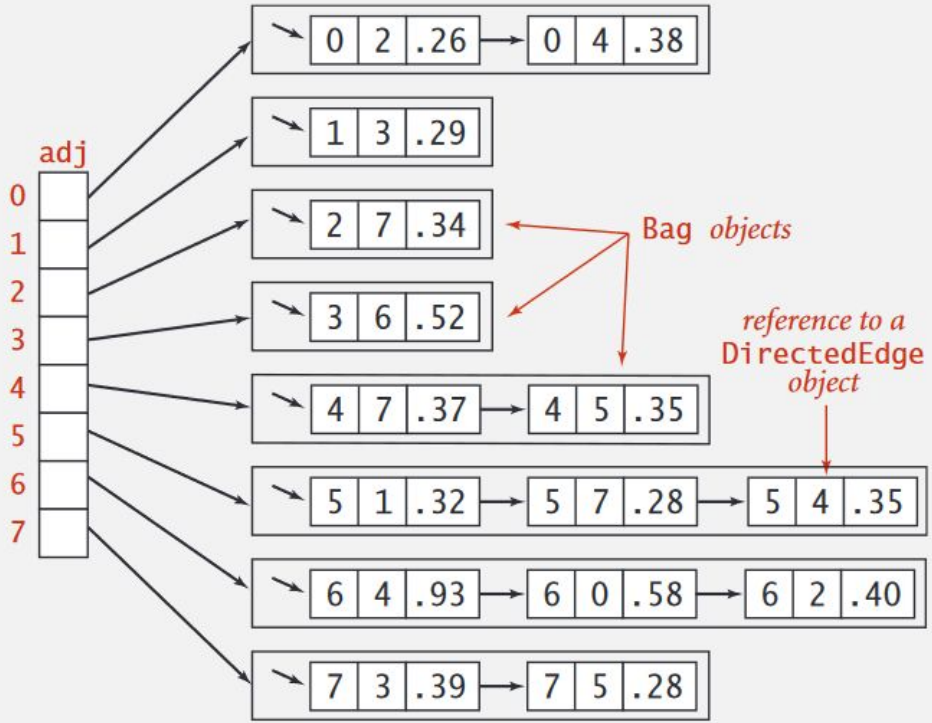
1. Изчисляване на най-къс път от връх до друг връх
2. Изчисляване на най-късите пътища от връх до всички други върхове
3. Изчисляване на най-късите пътища от всеки до всеки връх


0
1
2
3
4
5
1
15
5
9
3
7
10
4

Edge-weighted digraph: adjacency-lists representation

tinyEWD.txt

```
V → 8
    15 ← E
4 5  0.35
5 4  0.35
4 7  0.37
5 7  0.28
7 5  0.28
5 1  0.32
0 4  0.38
0 2  0.26
7 3  0.39
1 3  0.29
2 7  0.34
6 2  0.40
3 6  0.52
6 0  0.58
6 4  0.93
```

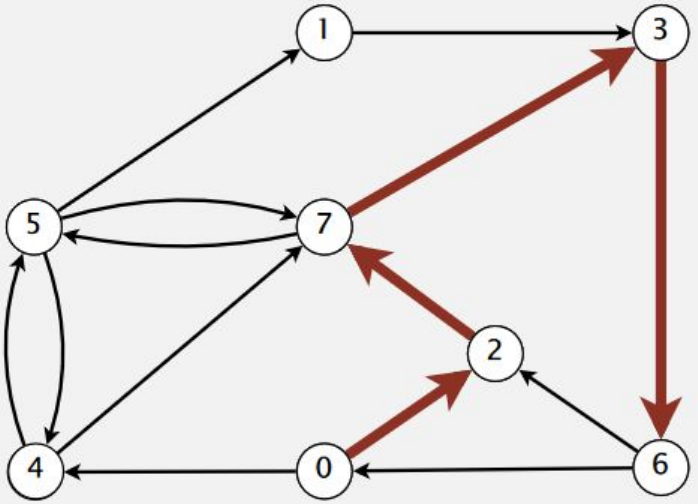


Shortest paths in an edge-weighted digraph

Given an edge-weighted digraph, find the shortest path from s to t.

edge-weighted digraph

```
4->5  0.35
5->4  0.35
4->7  0.37
5->7  0.28
7->5  0.28
5->1  0.32
0->4  0.38
0->2  0.26
7->3  0.39
1->3  0.29
2->7  0.34
6->2  0.40
3->6  0.52
6->0  0.58
6->4  0.93
```



shortest path from 0 to 6

$0 \rightarrow 2 \rightarrow 7 \rightarrow 3 \rightarrow 6$

length of path = 1.51

$(0.26 + 0.34 + 0.39 + 0.52)$

Алгоритъм на Дийкстра

Основна идея: Подобен на търсене в ширина, но вместо с опашка с приоритетна опашка.

```
function Dijkstra(Graph, source):

    create vertex set Q

    for each vertex v in Graph:                 // Initialization
        dist[v] ← INFINITY                      // Unknown distance from source to v
        prev[v] ← UNDEFINED                     // Previous node in optimal path from source
        add v to Q                              // All nodes initially in Q (unvisited nodes)

    dist[source] ← 0                            // Distance from source to source

    while Q is not empty:
        u ← vertex in Q with min dist[u]        // Node with the least distance
                                                // will be selected first
        remove u from Q

        for each neighbor v of u:               // where v is still in Q.
            alt ← dist[u] + length(u, v)
            if alt < dist[v]:                   // A shorter path to v has been found
                dist[v] ← alt
                prev[v] ← u

    return dist[], prev[]
```

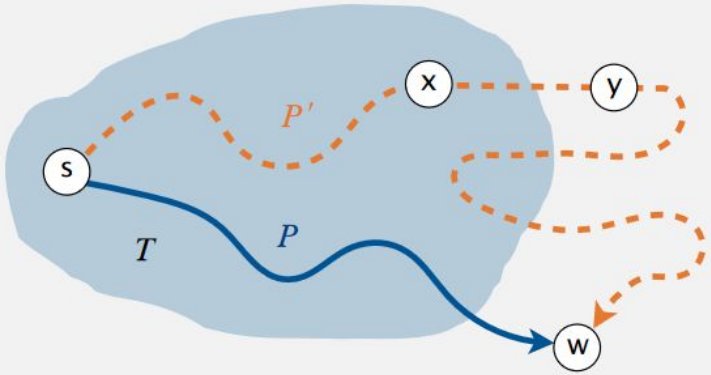
Dijkstra's algorithm: correctness proof

Invariant. For each vertex v in T, `distTo[v]` $= d^*(v)$.

length of shortest $s \rightsquigarrow v$ path

Pf. [by induction on $|T|$]

- Let w be next vertex added to T.
- Let P be the $s \rightsquigarrow w$ path of length `distTo[w]`.
- Consider any other $s \rightsquigarrow w$ path P'.
- Let $x \rightarrow y$ be first edge in P' that leaves T.
- P' is no shorter than P :



by construction

$$\text{length}(P) = \texttt{distTo[w]}$$

Dijkstra chose w instead of y $\longrightarrow$ $\leq \texttt{distTo[y]}$

relax vertex x $\longrightarrow$ $\leq \texttt{distTo[x]} + \mathit{weight}(x, y)$

induction $\longrightarrow$ $= d^*(x) + \mathit{weight}(x, y)$

weight are non-negative $\longrightarrow$ $\leq \text{length}(P')$ ∎

Shortest paths: quiz

What is the order of growth of the running time of Dijkstra's algorithm in the worst case when using a binary heap for the priority queue?

A. $V + E$

B. $V \log V$

C. $E \log V$

D. $E \log E$

Dijkstra's algorithm: which priority queue?

Depends on PQ implementation: V INSERT, V DELETE-MIN, $\leq E$ DECREASE-KEY.

PQ implementation	INSERT	DELETE-MIN	DECREASE-KEY	total
unordered array	1	V	1	V^2
binary heap	$\log V$	$\log V$	$\log V$	$E \log V$
d-way heap	$\log_d V$	$d \log_d V$	$\log_d V$	$E \log_{E/V} V$
Fibonacci heap	1 †	$\log V$ †	1 †	$E + V \log V$

† amortized

Bottom line.

- Array implementation optimal for complete graphs.
- Binary heap much faster for sparse graphs.
- 4-way heap worth the trouble in performance-critical situations.
- Fibonacci heap best in theory, but not worth implementing.

Algorithm for shortest paths

Variations on a theme: vertex relaxations.

- Bellman–Ford: relax all vertices; repeat $V - 1$ times.
- Dijkstra: relax vertices in order of distance from s.

algorithm	worst-case running time	negative weights †	directed cycles
Bellman-Ford	$E\,V$	✔	✔
Dijkstra	$E \log V$		✔

Следва контролно 5

Инструкции:

- Време за работа час и половина 16:30 - 18:00ч
- Две задачи
- Без ограничение за езици за програмиране и използване на стандартни библиотеки
- По време на контролното се позволява използването само на Хакерранк!
  - Не е позволено да се използва IDE
  - Не позволено да се използва помощ от интернет или от налични решения на компютъра