ETH zürich

Exercise Session 10

Data Structures and Algorithms, D-MATH, ETH Zurich

Program of today

Feedback of last exercises

Recap Theory
- graphs
- Shortest Paths
- Dijkstra
- Heaps, DecreaseKey and Lazy Deletion
- Running Time of the Algorithms
- Dijkstra and Negative Edge Weights?
- Bellman-Ford

1. Feedback of last exercises

2. Recap Theory

2.1 graphs

Quiz: Runtimes of simple Operations

Operation	Matrix	List
Find neighbours/successors of $v \in V$		
find $v \in V$ without neighbour/successor		
$(v, u) \in E$?		
Insert edge		
Delete edge		

Quiz: Runtimes of simple Operations

Operation	Matrix	List
Find neighbours/successors of $v \in V$	$\Theta(n)$	
find $v \in V$ without neighbour/successor		
$(v, u) \in E$?		
Insert edge		
Delete edge		

Quiz: Runtimes of simple Operations

Operation	Matrix	List
Find neighbours/successors of $v \in V$	$\Theta(n)$	$\Theta(\deg^+ v)$
find $v \in V$ without neighbour/successor		
$(v, u) \in E$?		
Insert edge		
Delete edge		

Quiz: Runtimes of simple Operations

Operation	Matrix	List
Find neighbours/successors of $v \in V$	$\Theta(n)$	$\Theta(\deg^+ v)$
find $v \in V$ without neighbour/successor	$\Theta(n^2)$	
$(v, u) \in E$?		
Insert edge		
Delete edge		

Quiz: Runtimes of simple Operations

Operation	Matrix	List
Find neighbours/successors of $v \in V$	$\Theta(n)$	$\Theta(\deg^+ v)$
find $v \in V$ without neighbour/successor	$\Theta(n^2)$	$\Theta(n)$
$(v, u) \in E$?		
Insert edge		
Delete edge		

Quiz: Runtimes of simple Operations

Operation	Matrix	List
Find neighbours/successors of $v \in V$	$\Theta(n)$	$\Theta(\deg^+ v)$
find $v \in V$ without neighbour/successor	$\Theta(n^2)$	$\Theta(n)$
$(v, u) \in E$?	$\Theta(1)$	
Insert edge		
Delete edge		

Quiz: Runtimes of simple Operations

Operation	Matrix	List
Find neighbours/successors of $v \in V$	$\Theta(n)$	$\Theta(\deg^+ v)$
find $v \in V$ without neighbour/successor	$\Theta(n^2)$	$\Theta(n)$
$(v, u) \in E$?	$\Theta(1)$	$\Theta(\deg^+ v)$
Insert edge		
Delete edge		

Quiz: Runtimes of simple Operations

Operation	Matrix	List
Find neighbours/successors of $v \in V$	$\Theta(n)$	$\Theta(\deg^+ v)$
find $v \in V$ without neighbour/successor	$\Theta(n^2)$	$\Theta(n)$
$(v, u) \in E$?	$\Theta(1)$	$\Theta(\deg^+ v)$
Insert edge	$\Theta(1)$	
Delete edge		

Quiz: Runtimes of simple Operations

Operation	Matrix	List
Find neighbours/successors of $v \in V$	$\Theta(n)$	$\Theta(\deg^+ v)$
find $v \in V$ without neighbour/successor	$\Theta(n^2)$	$\Theta(n)$
$(v, u) \in E$?	$\Theta(1)$	$\Theta(\deg^+ v)$
Insert edge	$\Theta(1)$	$\Theta(1)$
Delete edge		

Quiz: Runtimes of simple Operations

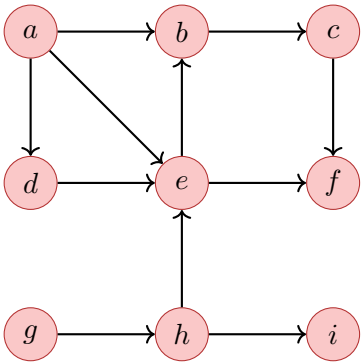
Operation	Matrix	List
Find neighbours/successors of $v \in V$	$\Theta(n)$	$\Theta(\deg^+ v)$
find $v \in V$ without neighbour/successor	$\Theta(n^2)$	$\Theta(n)$
$(v, u) \in E$?	$\Theta(1)$	$\Theta(\deg^+ v)$
Insert edge	$\Theta(1)$	$\Theta(1)$
Delete edge	$\Theta(1)$	

Quiz: Runtimes of simple Operations

Operation	Matrix	List
Find neighbours/successors of $v \in V$	$\Theta(n)$	$\Theta(\deg^+ v)$
find $v \in V$ without neighbour/successor	$\Theta(n^2)$	$\Theta(n)$
$(v, u) \in E$?	$\Theta(1)$	$\Theta(\deg^+ v)$
Insert edge	$\Theta(1)$	$\Theta(1)$
Delete edge	$\Theta(1)$	$\Theta(\deg^+ v)$

Breadth-First-Search BFS

BFS starting from a:

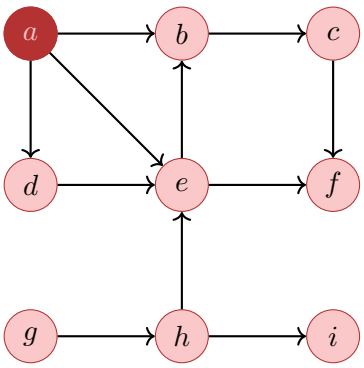


BFS-Tree: Distances and Parents

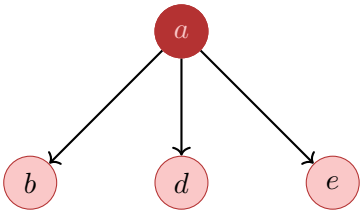
Distance 0

Breadth-First-Search BFS

BFS starting from a:



BFS-Tree: Distances and Parents

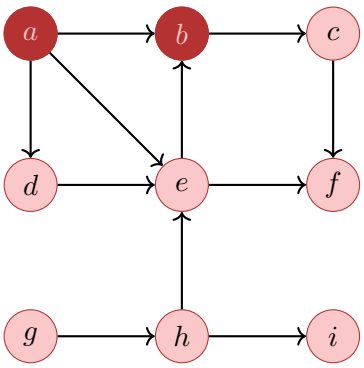


Distance 0

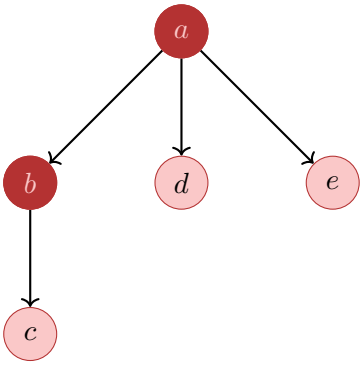
Distance 1

Breadth-First-Search BFS

BFS starting from a:



BFS-Tree: Distances and Parents



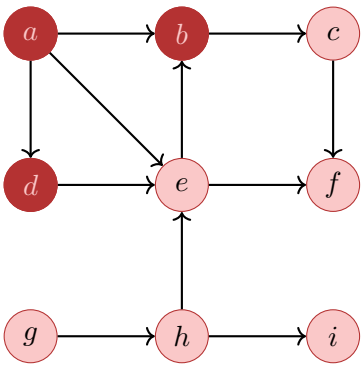
Distance 0

Distance 1

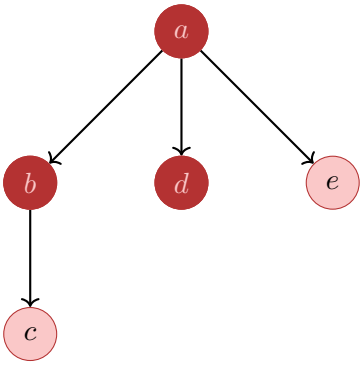
Distance 2

Breadth-First-Search BFS

BFS starting from a:



BFS-Tree: Distances and Parents



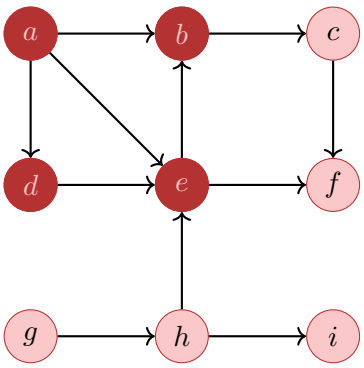
Distance 0

Distance 1

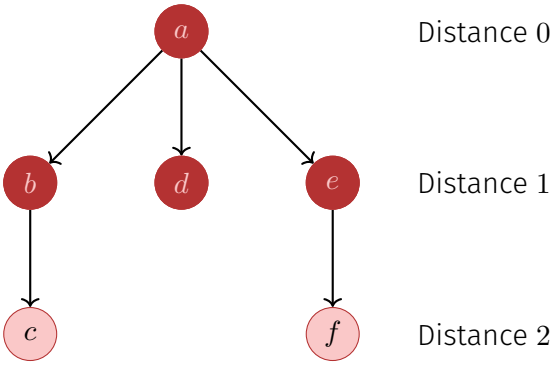
Distance 2

Breadth-First-Search BFS

BFS starting from a:

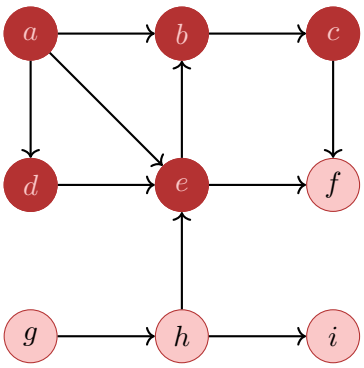


BFS-Tree: Distances and Parents

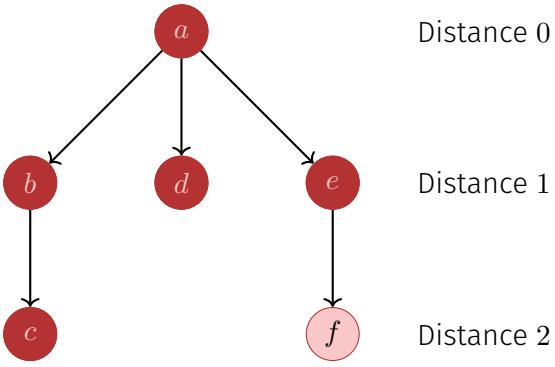


Breadth-First-Search BFS

BFS starting from a:

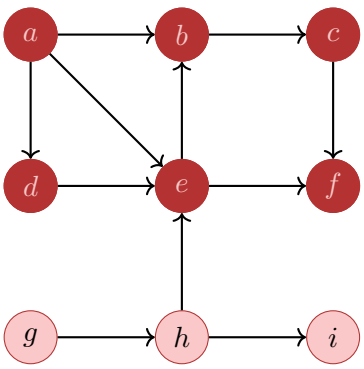


BFS-Tree: Distances and Parents

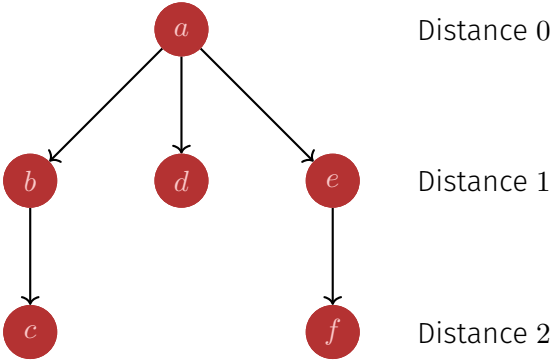


Breadth-First-Search BFS

BFS starting from a:

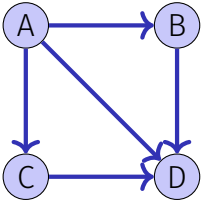


BFS-Tree: Distances and Parents



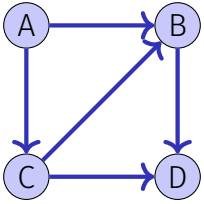
Quiz: Topological Sorting

In how many ways can the following directed graphs be topologically sorted each?



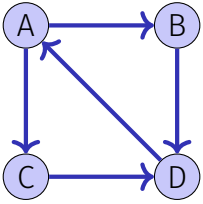
number sortings

?



number sortings

?

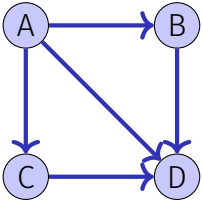


number sortings

?

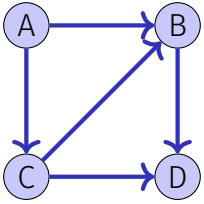
Quiz: Topological Sorting

In how many ways can the following directed graphs be topologically sorted each?



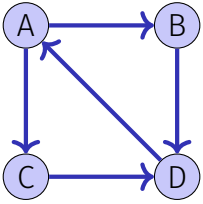
number sortings

2



number sortings

1



number sortings

0

2.2 Shortest Paths

General Algorithm

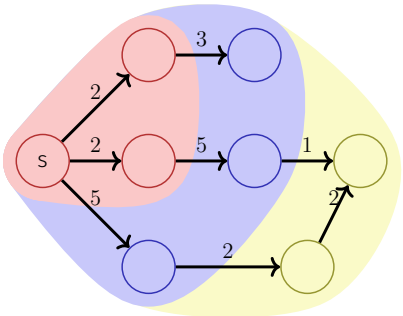
1. Initialise d_s and π_s: $d_s[v] = \infty$, $\pi_s[v] =$ null for each $v \in V$
2. Set $d_s[s] \leftarrow 0$
3. Choose an edge $(u, v) \in E$

   Relax (u, v):
   if $d_s[v] > d[u] + c(u, v)$ then
   $d_s[v] \leftarrow d_s[u] + c(u, v)$
   $\pi_s[v] \leftarrow u$
4. Repeat 3 until nothing can be relaxed any more.
   (until $d_s[v] \leq d_s[u] + c(u, v) \quad \forall (u, v) \in E$)

Dijkstra (positive egde weights)

Set V of nodes is partitioned into

- the set M of nodes for which a shortest path from s is already known,
- the set $R = \bigcup_{v \in M} N^+(v) \setminus M$ of nodes where a shortest path is not yet known but that are accessible directly from M,
- the set $U = V \setminus (M \cup R)$ of nodes that have not yet been considered.



Algorithm Dijkstra(G, s)

Input: Positively weighted Graph $G = (V, E, c)$, starting point $s \in V$,
Output: Minimal weights d of the shortest paths and corresponding predecessor node for each node.

foreach $u \in V$ **do**
    $d_s[u] \leftarrow \infty$; $\pi_s[u] \leftarrow null$
$d_s[s] \leftarrow 0$; $R \leftarrow \{s\}$
while $R \neq \emptyset$ **do**
    $u \leftarrow$ ExtractMin(R)
    foreach $v \in N^+(u)$ **do**
        if $d_s[u] + c(u, v) < d_s[v]$ **then**
            $d_s[v] \leftarrow d_s[u] + c(u, v)$
            $\pi_s[v] \leftarrow u$
            $R \leftarrow R \cup \{v\}$

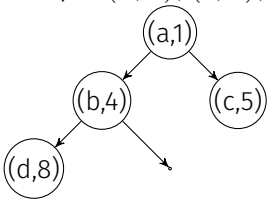
Implementation: Data Structure for R?

Relax for Dijkstra:

if $d_s[u] + c(u, v) < d_s[v]$ **then**
- $d_s[v] \leftarrow d_s[u] + c(u, v)$
- $\pi_s[v] \leftarrow u$
- **if** $v \notin R$ **then**
  - Add(R, v) // Insertion of a new $(v, d(v))$ in the heap of R
- **else**
  - DecreaseKey(R, v) // Update of a $(v, d(v))$ in the heap of R

DecreaseKey ?

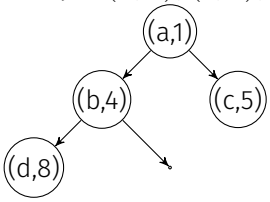
Heap ($(a,1),(b,4),(c,5),(d,8)$) =



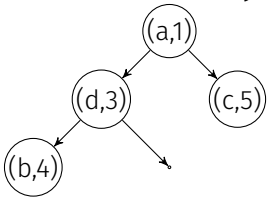
after DecreaseKey($d,3$):

DecreaseKey ?

Heap ($(a,1),(b,4),(c,5),(d,8)$) =



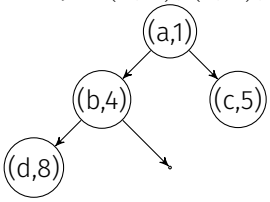
after DecreaseKey($d,3$):



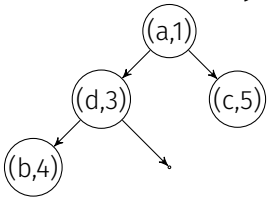
2 problems:

DecreaseKey ?

Heap ($(a,1),(b,4),(c,5),(d,8)$) =



after DecreaseKey($d,3$):

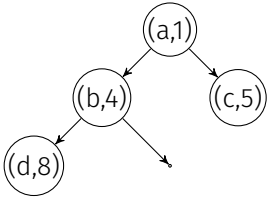


2 problems:

- Position of d unknown at first. Search: $\Theta(n)$
- Positions of the nodes can change during DecreaseKey

Lazy Deletion !

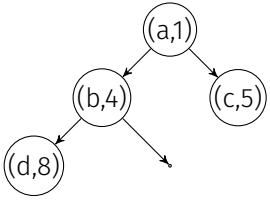
Heap ($(a,1),(b,4),(c,5),(d,8)$) =



Insert$(d,3)$:

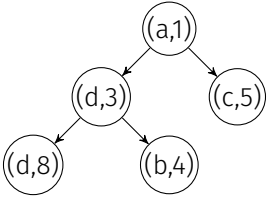
Lazy Deletion !

Heap ($(a,1),(b,4),(c,5),(d,8)$) =



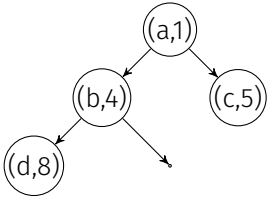
ExtractMin()

Insert($d,3$):

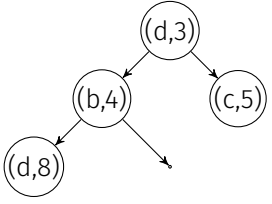


Lazy Deletion !

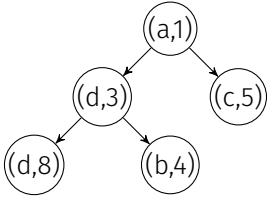
Heap ($(a,1),(b,4),(c,5),(d,8)$) =



ExtractMin() $\rightarrow (a,1)$



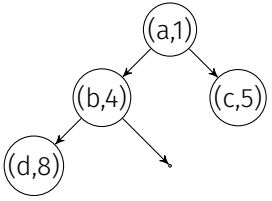
Insert$(d,3)$:



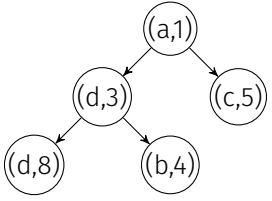
ExtractMin()

Lazy Deletion !

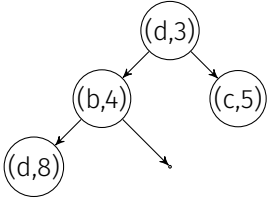
Heap ($(a,1),(b,4),(c,5),(d,8)$) =



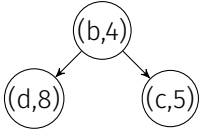
Insert$(d,3)$:



ExtractMin() $\rightarrow (a,1)$



ExtractMin() $\rightarrow (d,3)$



Later ExtractMin() $\rightarrow (d,8)$ must be ignored

Runtime Dijkstra

$n := |V|$, $m := |E|$

- $n\times$ ExtractMin: $\mathcal{O}(n \log n)$
- $m\times$ Insert or DecreaseKey: $\mathcal{O}(m \log n)$
- $1\times$ Init: $\mathcal{O}(n)$
- Overall: $\mathcal{O}((n + m) \log n)$. For connected graphs: $\mathcal{O}(m \log n)$.

2.5 Running Time of the Algorithms

Quiz: Single Source Shortest Paths

$n := |V|, m := |E|$

problem	method	runtime	dense $m \in \mathcal{O}(n^2)$	sparse $m \in \mathcal{O}(n)$
$c \equiv 1$	BFS			

Quiz: Single Source Shortest Paths

$n := |V|, m := |E|$

problem	method	runtime	dense $m \in \mathcal{O}(n^2)$	sparse $m \in \mathcal{O}(n)$
$c \equiv 1$	BFS	$\mathcal{O}(m+n)$		

Quiz: Single Source Shortest Paths

$n := |V|, m := |E|$

problem	method	runtime	dense $m \in \mathcal{O}(n^2)$	sparse $m \in \mathcal{O}(n)$
$c \equiv 1$	BFS	$\mathcal{O}(m+n)$	$\mathcal{O}(n^2)$	

Quiz: Single Source Shortest Paths

$n := |V|, m := |E|$

problem	method	runtime	dense $m \in \mathcal{O}(n^2)$	sparse $m \in \mathcal{O}(n)$
$c \equiv 1$	BFS	$\mathcal{O}(m+n)$	$\mathcal{O}(n^2)$	$\mathcal{O}(n)$
DAG	Top-Sort			

Quiz: Single Source Shortest Paths

$n := |V|, m := |E|$

problem	method	runtime	dense $m \in \mathcal{O}(n^2)$	sparse $m \in \mathcal{O}(n)$
$c \equiv 1$	BFS	$\mathcal{O}(m+n)$	$\mathcal{O}(n^2)$	$\mathcal{O}(n)$
DAG	Top-Sort	$\mathcal{O}(m+n)$		

Quiz: Single Source Shortest Paths

$n := |V|, m := |E|$

problem	method	runtime	dense $m \in \mathcal{O}(n^2)$	sparse $m \in \mathcal{O}(n)$
$c \equiv 1$	BFS	$\mathcal{O}(m+n)$	$\mathcal{O}(n^2)$	$\mathcal{O}(n)$
DAG	Top-Sort	$\mathcal{O}(m+n)$	$\mathcal{O}(n^2)$	

Quiz: Single Source Shortest Paths

$n := |V|, m := |E|$

problem	method	runtime	dense $m \in \mathcal{O}(n^2)$	sparse $m \in \mathcal{O}(n)$
$c \equiv 1$	BFS	$\mathcal{O}(m+n)$	$\mathcal{O}(n^2)$	$\mathcal{O}(n)$
DAG	Top-Sort	$\mathcal{O}(m+n)$	$\mathcal{O}(n^2)$	$\mathcal{O}(n)$
$c \geq 0$	Dijkstra			

Quiz: Single Source Shortest Paths

$n := |V|, m := |E|$

problem	method	runtime	dense $m \in \mathcal{O}(n^2)$	sparse $m \in \mathcal{O}(n)$
$c \equiv 1$	BFS	$\mathcal{O}(m+n)$	$\mathcal{O}(n^2)$	$\mathcal{O}(n)$
DAG	Top-Sort	$\mathcal{O}(m+n)$	$\mathcal{O}(n^2)$	$\mathcal{O}(n)$
$c \geq 0$	Dijkstra	$\mathcal{O}((m+n)\log n)$		

Quiz: Single Source Shortest Paths

$n := |V|, m := |E|$

problem	method	runtime	dense $m \in \mathcal{O}(n^2)$	sparse $m \in \mathcal{O}(n)$
$c \equiv 1$	BFS	$\mathcal{O}(m+n)$	$\mathcal{O}(n^2)$	$\mathcal{O}(n)$
DAG	Top-Sort	$\mathcal{O}(m+n)$	$\mathcal{O}(n^2)$	$\mathcal{O}(n)$
$c \geq 0$	Dijkstra	$\mathcal{O}((m+n)\log n)$	$\mathcal{O}(n^2 \log n)$	

Quiz: Single Source Shortest Paths

$n := |V|, m := |E|$

problem	method	runtime	dense $m \in \mathcal{O}(n^2)$	sparse $m \in \mathcal{O}(n)$
$c \equiv 1$	BFS	$\mathcal{O}(m+n)$	$\mathcal{O}(n^2)$	$\mathcal{O}(n)$
DAG	Top-Sort	$\mathcal{O}(m+n)$	$\mathcal{O}(n^2)$	$\mathcal{O}(n)$
$c \geq 0$	Dijkstra	$\mathcal{O}((m+n)\log n)$	$\mathcal{O}(n^2 \log n)$	$\mathcal{O}(n \log n)$
general	Bellman-Ford	$\mathcal{O}(m \cdot n)$		

Quiz: Single Source Shortest Paths

$n := |V|, m := |E|$

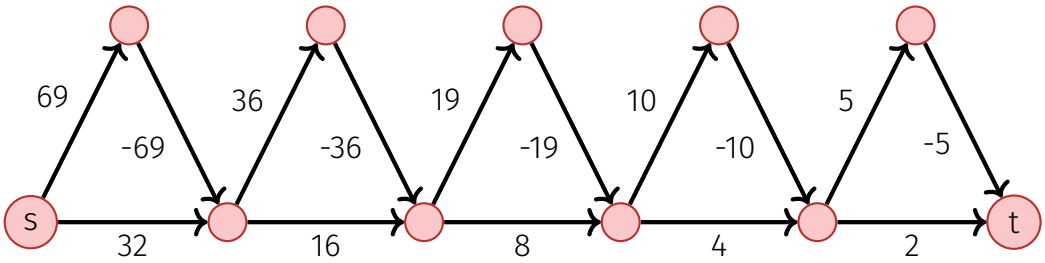
problem	method	runtime	dense $m \in \mathcal{O}(n^2)$	sparse $m \in \mathcal{O}(n)$
$c \equiv 1$	BFS	$\mathcal{O}(m+n)$	$\mathcal{O}(n^2)$	$\mathcal{O}(n)$
DAG	Top-Sort	$\mathcal{O}(m+n)$	$\mathcal{O}(n^2)$	$\mathcal{O}(n)$
$c \geq 0$	Dijkstra	$\mathcal{O}((m+n)\log n)$	$\mathcal{O}(n^2 \log n)$	$\mathcal{O}(n \log n)$
general	Bellman-Ford	$\mathcal{O}(m \cdot n)$	$\mathcal{O}(n^3)$	

Quiz: Single Source Shortest Paths

$n := |V|, m := |E|$

problem	method	runtime	dense $m \in \mathcal{O}(n^2)$	sparse $m \in \mathcal{O}(n)$
$c \equiv 1$	BFS	$\mathcal{O}(m+n)$	$\mathcal{O}(n^2)$	$\mathcal{O}(n)$
DAG	Top-Sort	$\mathcal{O}(m+n)$	$\mathcal{O}(n^2)$	$\mathcal{O}(n)$
$c \geq 0$	Dijkstra	$\mathcal{O}((m+n)\log n)$	$\mathcal{O}(n^2 \log n)$	$\mathcal{O}(n \log n)$
general	Bellman-Ford	$\mathcal{O}(m \cdot n)$	$\mathcal{O}(n^3)$	$\mathcal{O}(n^2)$

Quiz: An Interesting Graph



Does Dijkstra work?

Answer

Answer

Dijkstra (as we have presented it) works also for graphs with negative edge weights, if no negative weight cycles are present. But Dijkstra may then exhibit exponential running time!

General Weighted Graphs

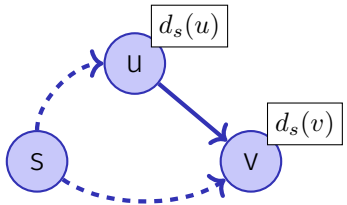
Relax(u, v) $(u, v \in V, (u, v) \in E)$

if $d_s(v) > d_s(u) + c(u, v)$ **then**

$d_s(v) \leftarrow d_s(u) + c(u, v)$

return true

return false



Problem: cycles with negative weights can shorten the path, a shortest path is not guaranteed to exist.

Dynamic Programming Approach (Bellman)

Induction over number of edges $d_s[i, v]$: Shortest path from s to v via maximally i edges.

$$d_s[i, v] = \min\{d_s[i-1, v], \min_{(u,v)\in E}(d_s[i-1, u] + c(u, v))$$

$$d_s[0, s] = 0, d_s[0, v] = \infty \; \forall v \neq s.$$

Algorithm Bellman-Ford(G, s)

Input: Graph $G = (V, E, c)$, starting point $s \in V$
Output: If return value true, minimal weights d for all shortest paths from s, otherwise no shortest path.

foreach $u \in V$ **do**
    $d_s[u] \leftarrow \infty$; $\pi_s[u] \leftarrow$ null
$d_s[s] \leftarrow 0$;
for $i \leftarrow 1$ **to** $|V|$ **do**
    $f \leftarrow$ false
    foreach $(u, v) \in E$ **do**
        $f \leftarrow f \vee \mathrm{Relax}(u, v)$
    if $f =$ false **then return** true
return false;