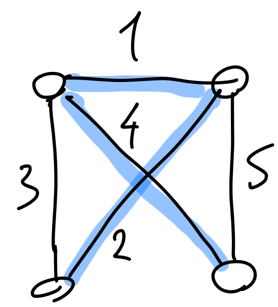
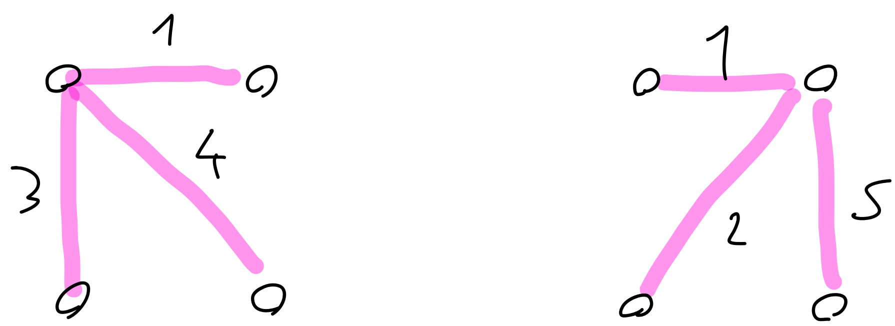
Solutions to exercises

1) NO: think of G = a tree

2) Second-best MST:



w(MST) = 7



w(second-best) = 8

3) Complexity of Find(x) is O(log n)

depth(x) $\forall x$

initially, depth(x)

Depth(x) can increase only because of a Union by 1 in which the root of the tree of x points to another root. This happens only when the tree of x is merged with a tree at least as big $\Longrightarrow$ when the depth of x increases, the size of the tree of x at least doubles. How many times can this happen? $\leq \log_2 n$ times, i.e. the depth of x cannot increase more than $\log_2 n$ times.

$\exists$ alg. with complexity $O(m \log n)$

$O(m)$? Open problem!

Shortest Paths

Definitions & terminology

Given a weighted graph, the length of a path $P = v_1, v_2, \ldots, v_k$ is defined as $len(P) =$

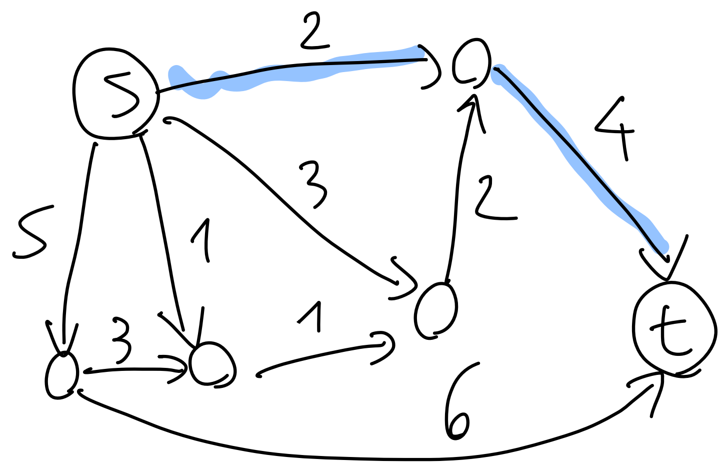
$$= \sum_{i=1}^{k-1} w(v_i, v_{i+1})$$

A shortest path from a vertex u to vertex v is a path with minimum length among all possible u–v paths

The distance between two vertices s and t, denoted $dist(s,t)$, is the length of a shortest path from s to t; if there is no path at all from s to t then $dist(s,t) = +\infty$

Problem: given a directed, weighted graph and a source vertex $s \in V$ and a destination $t \in V$, compute a shortest path from s to v

Example:



$dist(s,t) = 6$

Obs.: in directed graphs, in general $dist(u,v) \neq dist(v,u)$

Applications: - road networks (Google Maps)
- routing in computer networks (Internet)
- ...

we'll solve this problem:

Single-Source Shortest Paths (SSSP)

input: a directed, weighted graph G with edge weights $w: E \to \mathbb{R}$ and a source vertex $s \in V$

output: $\text{dist}(s, v)$ $\forall$ vertex $v \in V$

Comments: - no algorithms are known for the previous problem that run asymptotically faster than the best SSSP algorithm in worst case

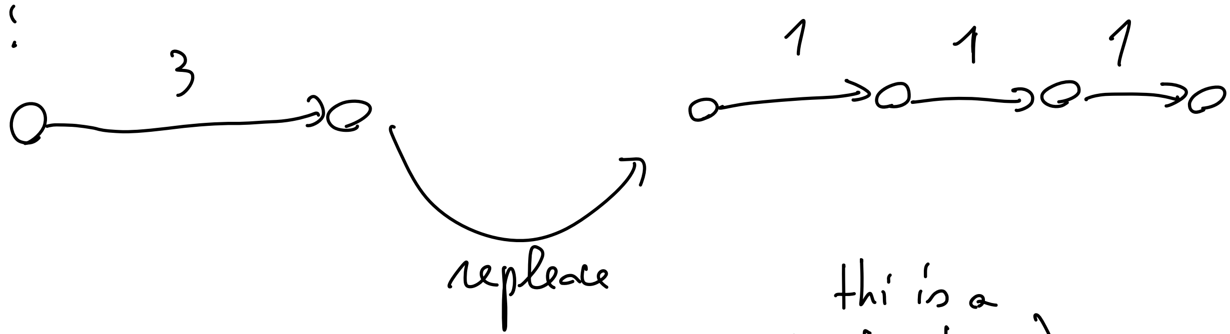
- we'll work with directed graphs, but all the algorithms that we'll see can be adapted easily for undirected graphs

We'll first solve a special case: <u>nonnegative edge weights</u> $w: E \to \mathbb{R}_{\geq 0}$

We've already solved a special case of this?

when all weights are $w=1$ → solved in linear time using BFS

idea:

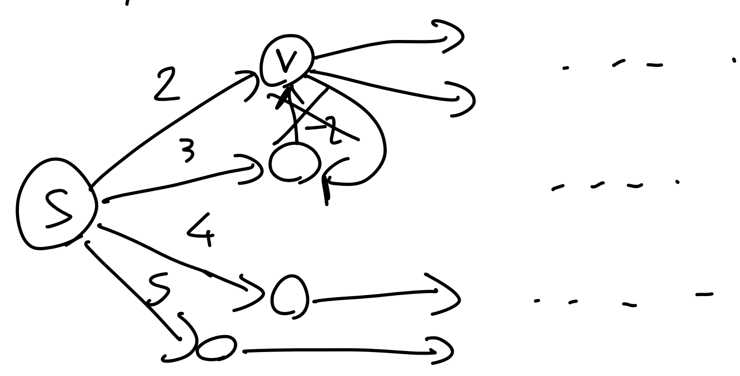


and then run BFS (this is a "reduction")

first issue: integer weights, so it's still a special case

bigger issue: the size of the graph can be much bigger than size of the original graph
⟹ BFS takes linear time in the "bigger" graph, and this is not necessarily linear time in the size of the original graph!

Intuition for a new algorithm:



the arc (s,v) must be the shortest path from s to v since the first segment of any other path is already larger, and weights are nonnegative; a similar reasoning works in the next steps

Dijkstra's algorithm (1956)

greedy algorithm, very similar to Prim

Dijkstra (G, s)

input: directed G, $s \in V$, $w : E \to \mathbb{R}_{\geq 0}$

output: $dist(s,v) = len(v) \quad \forall v \in V$

shorthand for

$X = \{s\}$

$len(s) = 0$

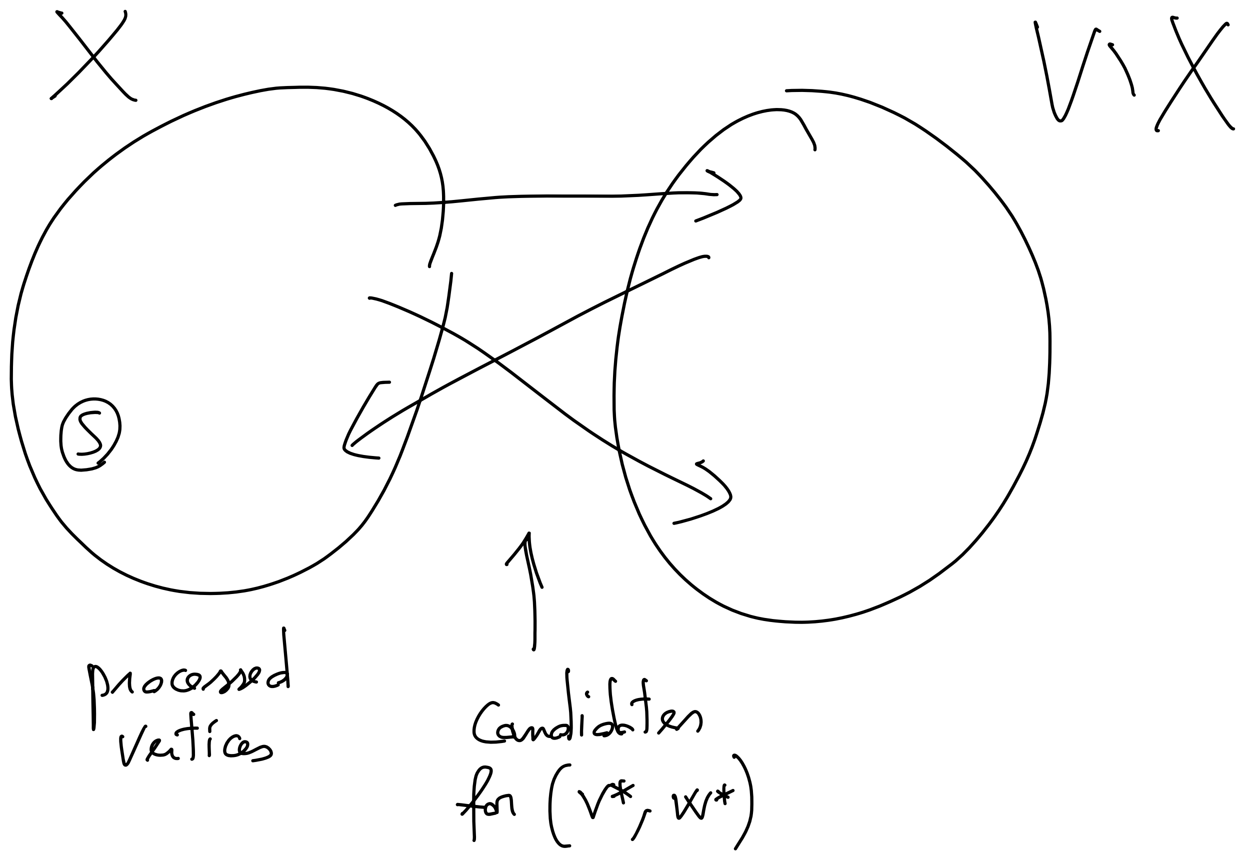
$len(v) = +\infty$ // initial estimated distance

while there is an edge (v,w) with $v \in X$ and $w \notin X$ do

(v^*, w^*) = such an edge minimizing $len(v) + w(v,w)$

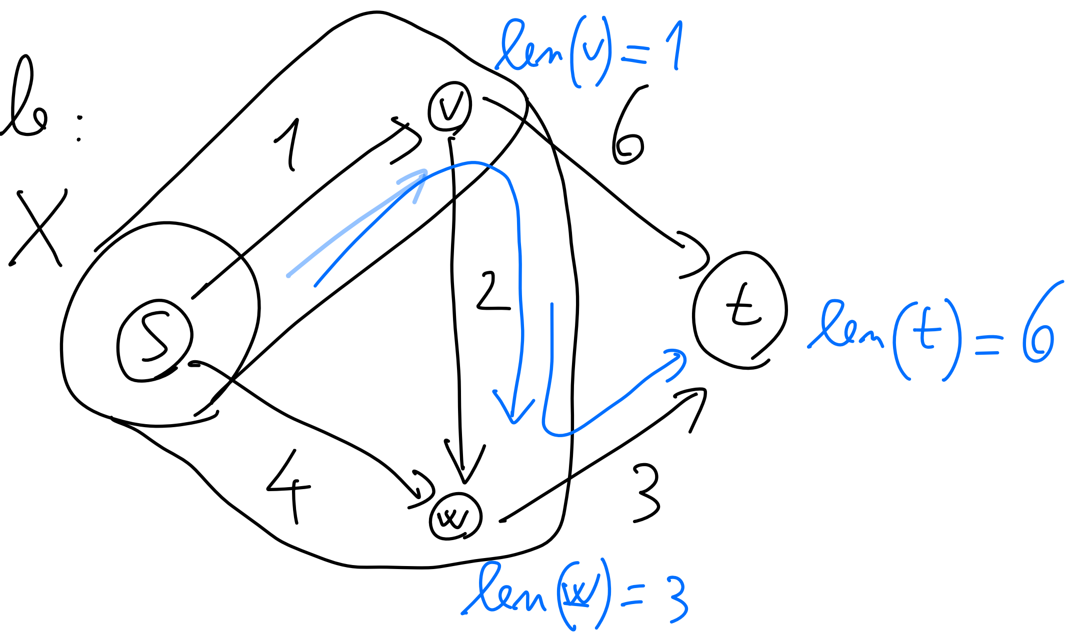
add w^* to X

$len(w^*) = len(v^*) + w(v^*, w^*)$

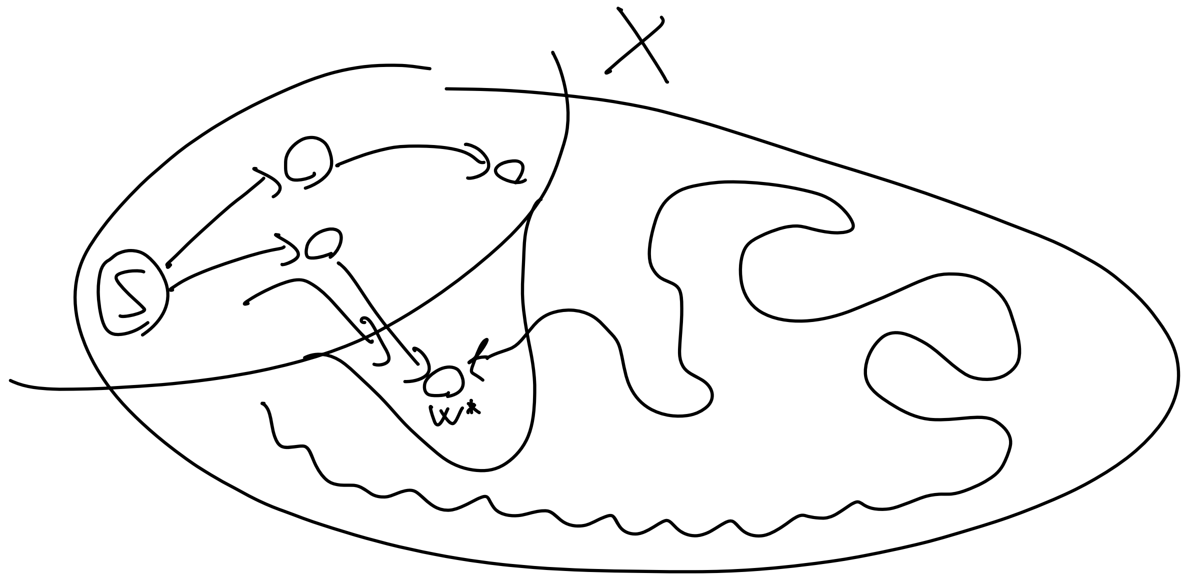


at each iteration a new node gets processed: w*

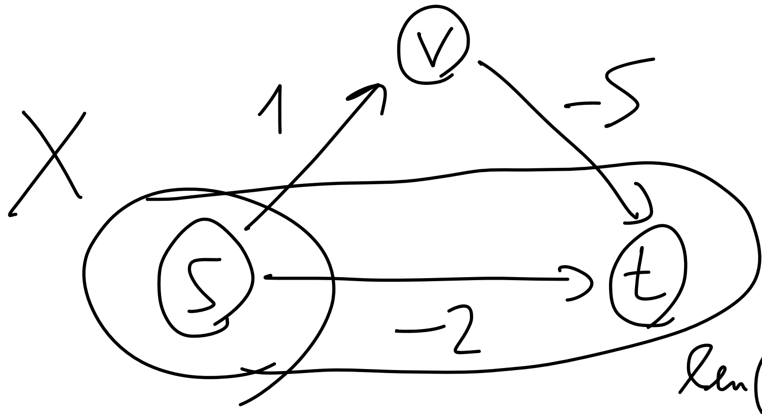
Example:



Praise of Dijkstra: in each iteration it irrevocably and myopically estimates the shortest path distance to one additional vertex, despite having so far looked at only a fraction of the graph!



Obs.: Dijkstra's algorithm does not work on graphs with negative weights:



$len(t) = -2$

but $len(t) = -4$!

Complexity: $O(m \cdot n)$

Exercise: write an implementation of Dijkstra's algorithm with <u>heaps</u>

Correctness of Dijkstra's algorithm:

invariant: $\forall x \in X$, $len(x)$ is $dist(s,x)$

by induction on $|X|$

base case: $|X| = 1$ trivial

inductive hypothesis: invariant is true $\forall |X| = k \geq 1$

- let v the next vertex added to X, and (u,v) the arc

w^*

(v^*, w^*)

- the shortest path from s to u + (u,v) is a path from s to v of length

$$\pi(v) = \min_{\substack{(u,v):\, u \in X \\ v \notin X}} len(u) + w(u,v)$$

- Consider any path P from s to v
  We need to show that P is not shorter
  than $\pi(v)$: next class