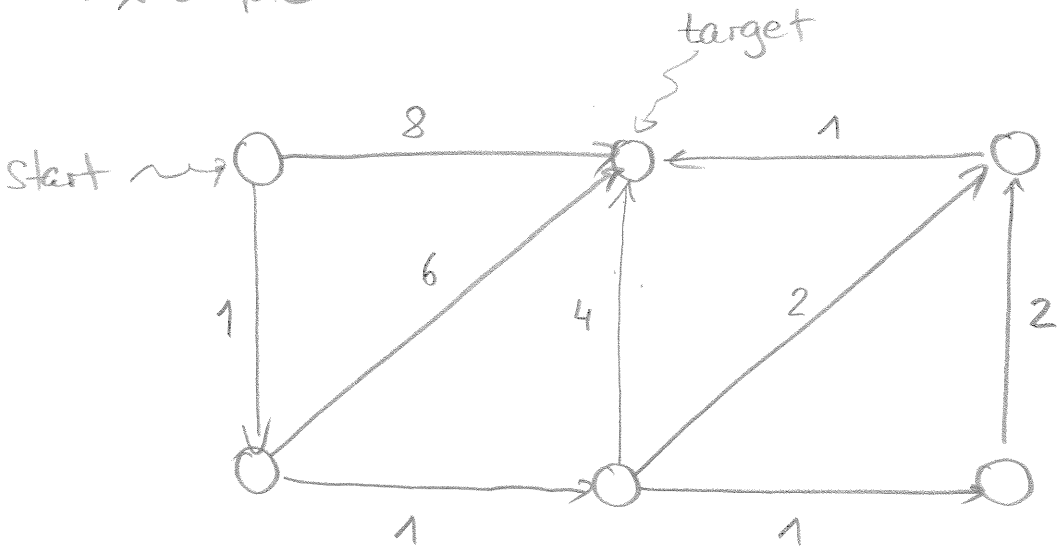
Shortest Paths

Example:



Formally:

- digraph $(V, E) = G$
- weight function $w: V \longrightarrow \mathbb{R}^+$
- path $p = v_0 \longrightarrow v_1 \cdots v_{k-1} \longrightarrow v_k$ has weight

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

- a shortest path from u to v has minimum weight among all such paths
- shortest path weight (or distance) from u to v is

$$\delta(u,v) := \min \{ w(p) \mid p \text{ path } u \longrightarrow v \}$$

Applications:

- Route finding (Google maps)
  - based on SP, but heavily optimized

- IP packet routing
  - nodes $\hat{=}$ routers
  - weights $\hat{=}$ link capacity (bandwidth^{-1})
  - router computes routing table:
    which is the best link on path to target t?
    - OSPF (= open shortest path first) protocol

Problems: Given $G = (V, E)$, $s, t \in V$

1) Which is the/a shortest path from s to t?

2) What is the length of the shortest path from s to t?

3) What is the length of the shortest path from s to v, for all $v \in V$?

I.e., compute

$$dist : V \longrightarrow \mathbb{R}^+, \quad dist(v) = \delta(s, v)$$

Approach:

- Solve 3)
- Keep additional info to construct path(s)

Options:

- D&C ?
  - splitting a graph does not yield independent subproblems

- DP ? Greedy?

Optimal Substructure

Suppose $p = u \rightarrow\rightarrow v$ is a shortest path

Then any subpath $x \rightarrow\rightarrow y$ is a shortest path from x to y

Proof: Cut & Paste!

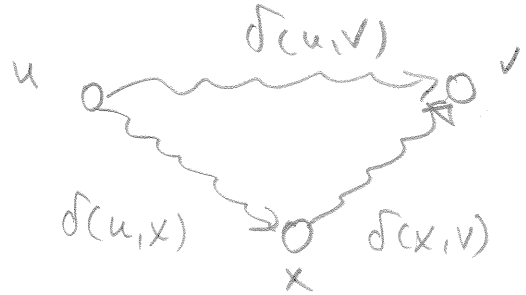
If there is a shorter path from x to y, then copy it into $u \rightarrow\rightarrow v$. ↯

Triangle Inequality

For all $u, v, x \in V$

$$\delta(u,v) \leq \delta(u,x) + \delta(x,v)$$

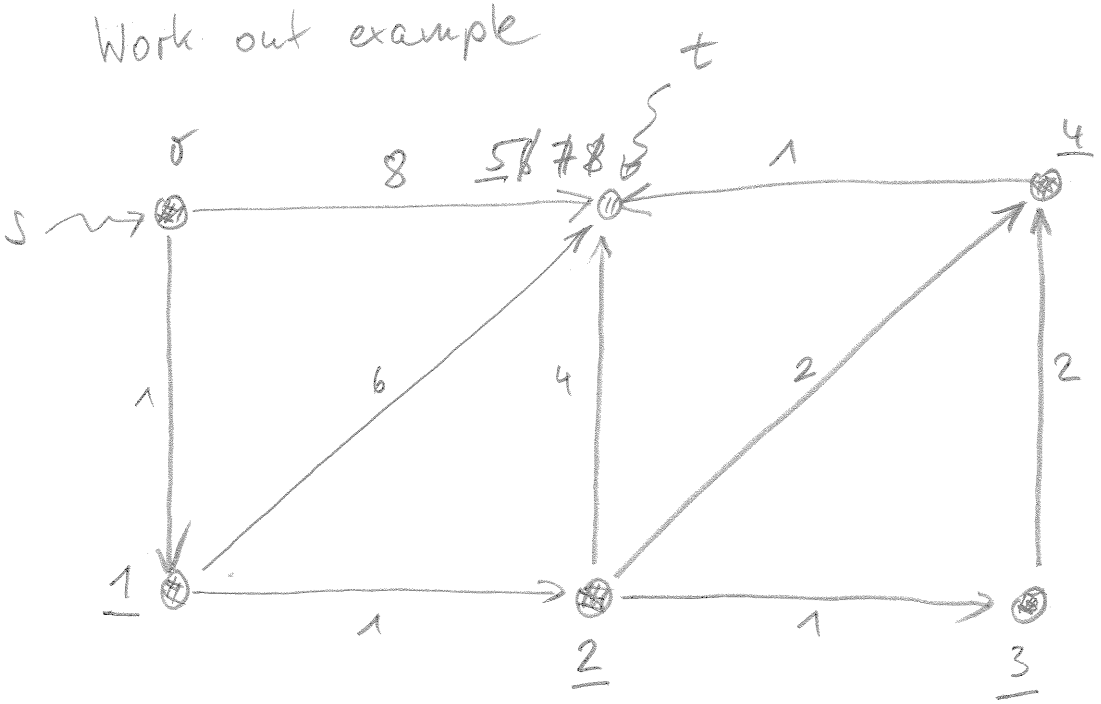
Proof:



Idea: Traverse G starting from s

- 3 node sets:
  - Black : fully explored (B)
  - White : unexplored (W)
  - Gray : under exploration (G)
- $v \in B$: $\delta(s,v)$ found
  $v \in G$: upper bound for $\delta(s,v)$ estimated
  $v \in W$: upper bound $= \infty$
- with each step, finish some $v \in G$
  - record distance $\delta(s,v)$
  - improve estimates for adjacent nodes
  - v becomes black

Work out example



Shortest Path Algorithm (Dijkstra)

Idea:

- Maintain $V \setminus B$ as priority queue ($= G \cup W$)
- key each $v \in Q$ with best estimate

  ($= \min \{ \delta(s,u) + w(u,v) \mid u \in B, (u,v) \in E \}$

  if there is an edge (u,v), ∞ otherwise

```
Q.init(V)
for v in V do  v.key := ∞
s.key := 0

while Q.nonEmpty() do
    u := Q.extractMin()
    for v in u.adj() do
        if v ∈ Q and u.dist + w(u,v) < v.key
"relaxation" ~~~> then Q.decreaseKey(v, u.dist + w(u.v))
                   v.pred := u

return t.dist;
```

$\{(v_0, v_1), \dots, (v_{k-1}, v_k)\}$

where $v_k = t$, $v_{i-1} = v_i.pred$,

$v_0 = s$

Correctness 1: Estimates are Pessimistic

After initialization and

after each relaxation

$$v.dist \geq \delta(s,v) \qquad (*)$$

for all $v \in V$

Proof: (Induction)

Base case: Initialization

- $s.dist = 0 = \delta(s,s)$
- $v.dist = \infty \geq \delta(s,v)$, for all $v \neq s$

Inductive step:

Consider relaxation

$$v.dist := u.dist + w(u,v)$$

when

$$u.dist \geq \delta(s,u) \qquad \text{(Ind. Hyp.)}$$

Then

$$v.dist \geq \delta(s,u) + \delta(u,v)$$

$$\geq \delta(s,v) \qquad \text{(Triangle Ineq.)}$$

□

Correctness 2: Correct Estimates Propagate

Consider an iteration step where u is extracted.

If $u.dist = \delta(s,u)$

and $s \to \dots \to u \to v$ is a shortest path

then $v.dist = \delta(s,v)$ after relaxation of v.

Proof: We have

$$\delta(s,v) = \underbrace{w(s \to \dots \to u)}_{\delta(s,u)} + w(u,v)$$

Correctness 1 $\Rightarrow$ $v.dist \geq \delta(s,v)$

Case 1: $v.dist = \delta(s,v)$ before relaxation

$\Rightarrow$ ✓

Case 2: $v.dist > \delta(s,v)$

$\Rightarrow$ we relax

$$\begin{aligned} v.dist &:= u.dist + w(u,v) \\ &= \delta(s,u) + w(u,v) \quad \text{(Assumption)} \\ &= \delta(s,v) \quad \text{(shortest path)} \end{aligned}$$

□

Question: Is $u.dist = \delta(s,u)$ when u is extracted?

Correctness 3: Extracted Estimates are Correct

Consider a step when u is extracted. Then

$$u.dist = \delta(s,u)$$

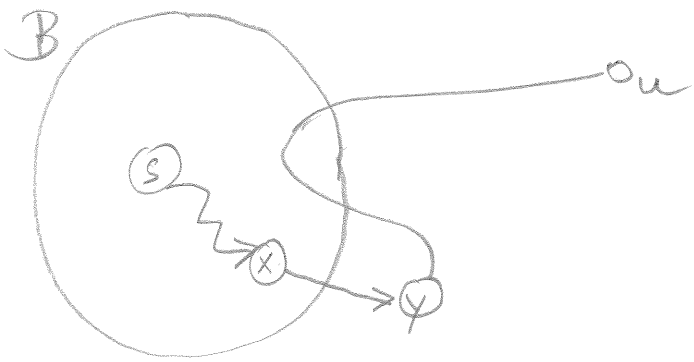
Proof: (Contradiction)

Suppose u is the first extracted vertex such that

$$u.dist \neq \delta(s,u)$$

C1 $\Rightarrow$ $u.dist > \delta(s,u)$

Let $p = s \rightsquigarrow u$ be a shortest path, i.e., $w(p) = \delta(s,u)$



Let $x \to y$ be the first black-grey edge on p.

Assumption: u is first violation

$\Rightarrow$ $x.dist = \delta(s,x)$

C2 $\Rightarrow$ $y.dist = \delta(s,y)$

Assumption $\Rightarrow$ $y \neq u \Rightarrow u.dist > \delta(s,u) > \delta(s,y) = y.dist$

$\Rightarrow$ y should have been extracted ↯ □

Dijkstra's Algorithm : Analysis

Running Time = $T_{Init} + |V| \cdot T_{extractMin} + |E| \cdot T_{decreaseKey}$

RT depends on implementation of priority queue

- Array, List: $\Theta(V + V^2 + E) = \Theta(V^2)$
- RB-Tree: $\Theta(V \cdot \log V + V \cdot \log V + E \cdot \log V)$
  $= \Theta((V+E) \log V)$
- Maxheap: $\Theta(V + V \cdot \log V + E \cdot \log V)$
  $= \Theta((V+E) \log V)$

Which is best?

- sparse graphs ($E = \Theta(V)$)
- dense graphs ($E = \Theta(V^2)$)