Shortest Paths

1. BFS → Bipartiteness testing
   → "Shortest paths"
2. DFS
   → "Checking for cycles"
   → "Topological Sort.

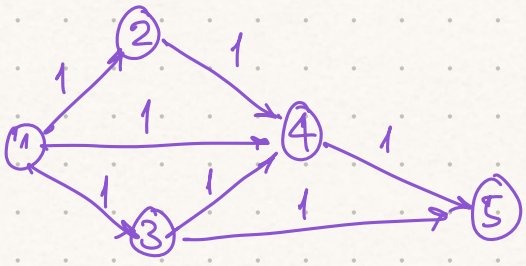
Tutorial on
10.08.2024
at 2pm in H105

Tentative:
Wednesdays
at 2pm (H205?)

$G = (V, E)$

(s, t)

(1, 4)



2 via 1–2–4, 1–3–4
1 via 1–4

If all edge wts are "equal" then BFS gives us shortest paths.

Shortest paths in DAGs

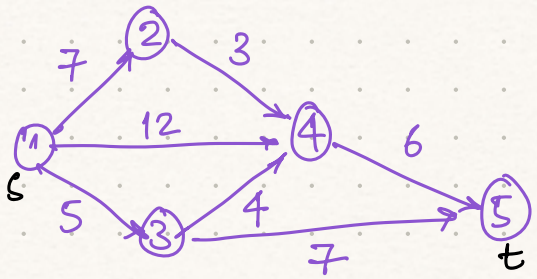
Directed Acyclic Graphs

$G = (V, E)$, $wt: E \to \mathbb{R}$.

$wt(e)$

If P is a path with edges $e_1, \ldots, e_k$ in it then

$wt(P) = wt(e_1) + wt(e_2) + \cdots + wt(e_k)$



Qu: Want the shortest distance from $1 \rightsquigarrow 5$.

- Place the nodes in topological order.

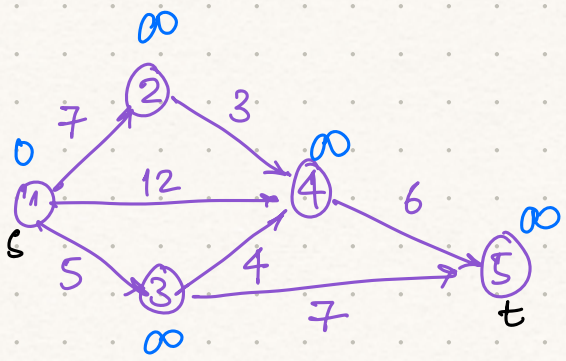
- $\forall\ v \in V \setminus \{s\}$, $dist(s,t) \leftarrow \infty$
- Visited $\leftarrow \{s\}$.
- While Visited $\neq V$:

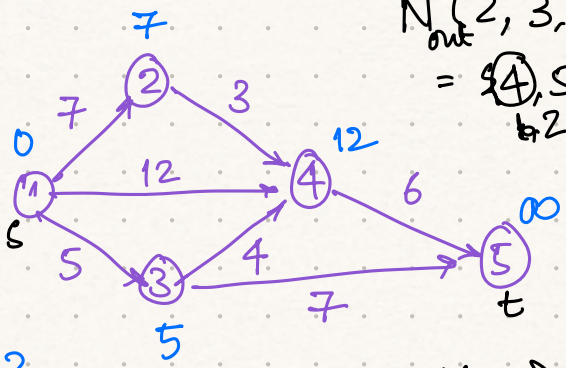
Not precise

For all $v \in N(\text{Visited})$:

$$\text{dist}(s,v) = \min\{\{\text{dist}(s,u) + wt(u,v) \mid (u,v)\in E, u \in \text{Visited}\} \cup \{\text{dist}(s,t)\}.$$

Visited $\leftarrow$ Visited $\cup \{v\}$.



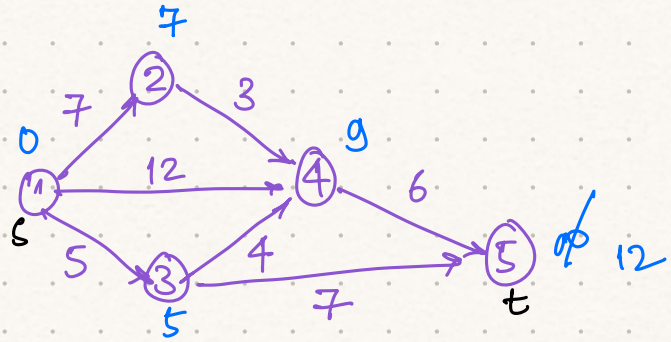


$N_{out}(2,3,4) = \{4,5\}$.

$\{2,3\}$

$$\text{dist}(s,4) = \min\{\text{dist}(1,4), \text{dist}(1,2)+wt(2,4), \text{dist}(1,3)+wt(3,4)\}$$

12, 10, 9



Single source graphs with no-negative edges.

→ Djikstra's algorithm.

Visited $\leftarrow \{s\}$, $d[s]=0$

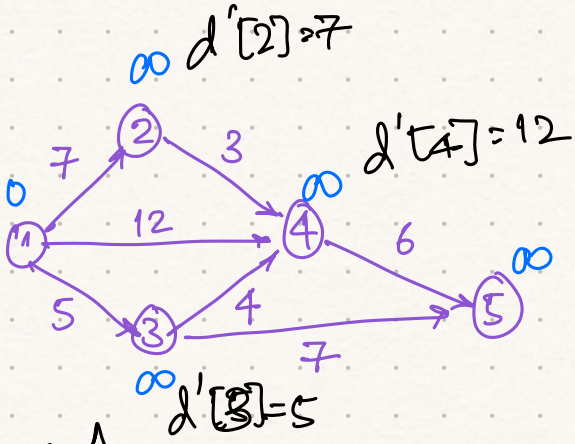
For every $v \in V \setminus \{s\}$:

$d(v) \leftarrow \infty$

While Visited $\neq V$:

For all $v \in$ (A) $(V \setminus \text{visited}) \cap N(\text{Visited})$:

$$d'[v] = \min_{(u,v)\in E} \{d(u) + wt((u,v))\}$$



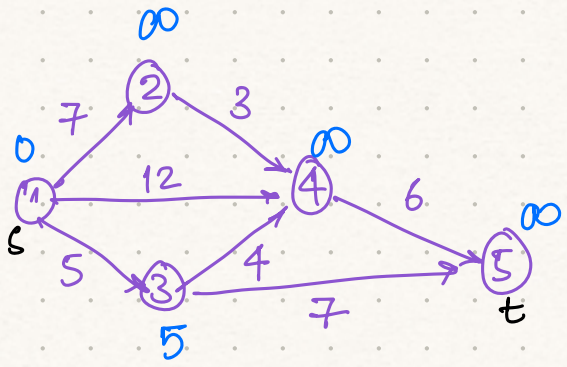
ue Visited

Select $v \in (V \setminus \text{Visited}) \cap (N(\text{Visited}))$ s.t

$d'[v] = \min \{ d'[v] \mid v \in A \}$

set $d[v] \leftarrow d'[v]$.

Add v to Visited.



$N(1,3) = \{2,4,5\}$.

$d'[2] = \min \{ wt(1,2) \} = 7$

$d'[4] = \min \{ d[2] + wt(2,4), d[1] + wt(1,4), d[3] + wt(3,4) \} = \{\infty, 12, 9\} = 9$

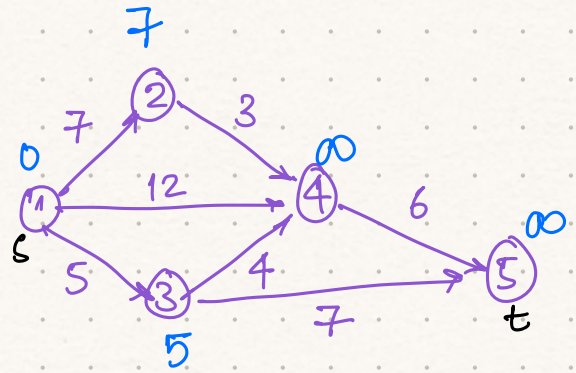
Visited $\leftarrow \{1,3\}$

$d'[5] = \min \{ d[3] + wt(3,5) \} = 12$

$\Rightarrow$ 2 attains min d' value.

$d[2] = d'[2] = 7$.

Visited $= \{1,2,3\}$.



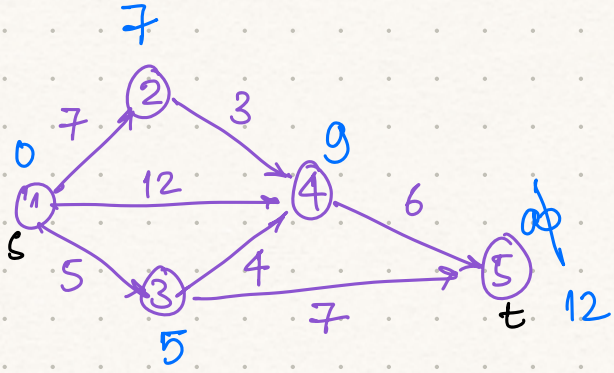
$N(\{1,2,3\}) = \{4,5\}$.

$d'[4] = \min \{ d[2] + wt(2,4), d[3] + wt(3,4), wt(1,4) \} = \min \{ 7+3, 5+4, 12 \} = 9$

$d'[5] = \min \{ d[3] + wt(3,5) \} = 12$

$\Rightarrow$ 4 attains the min d' value.

$d[4] = d'[4]$.



Visited $= \{1,2,3,4\}$

$N(\text{Visited}) = \{5\}$.

$d'[5] = \min \{ d[4] + wt(4,5), d[3] + wt(3,5) \}$

d[u] ← shortest $s \rightsquigarrow u$ path.

Obs:

1. Visited set grows but elements once added are not disturbed in the later stage of the algorithm.
2. Shortest distances once computed are not updated ever again.

Correctness:

Lemma: Consider the set visited at an arbitrary point of time in the algo's execution. For all $u \in$ Visited, d[u] is the shortest distance from $s \rightsquigarrow u$.

S'

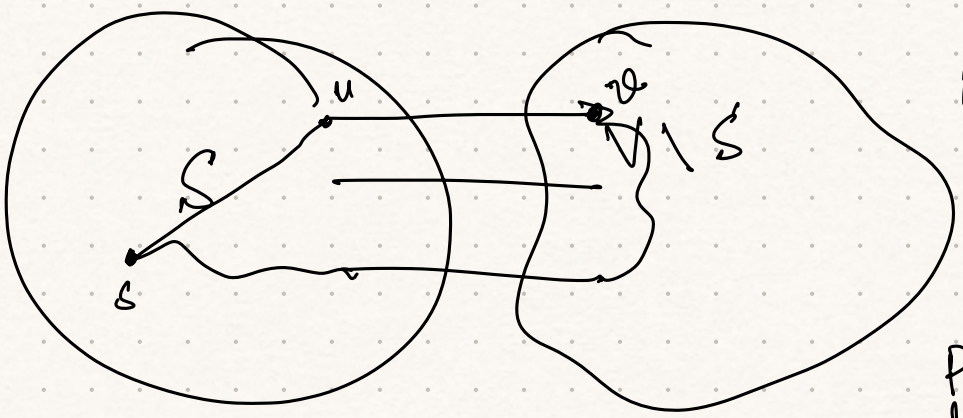
Proof: Proof by induction on the size of the set Visited.

Base case: $|S| = 1$.

$S' : \{s\}$ $\quad d[s] = 0$

Induction hypothesis: The statement of the lemma is true S s.t $|S| = k$ where $k \geq 1$.

$k \leq n-1$.



$N(S) \cap (V \setminus S)$

For each $v' \in$ we compute d' values.

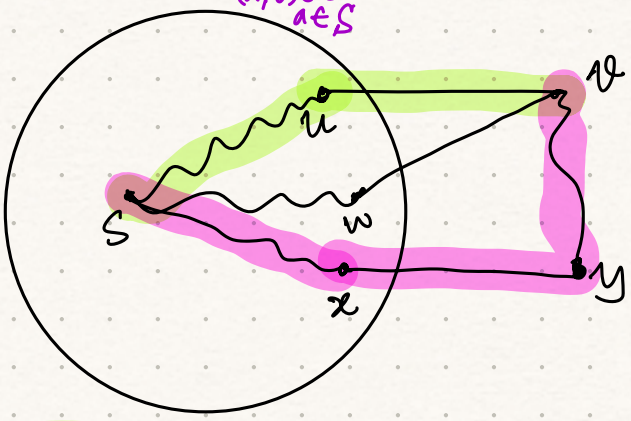
Pick an element that attains min d' value.

Through this process say v attains the min d' value. Here d[v] is set to d'[v].

Qu: Does there exist any other path than $s \rightsquigarrow u \to v$ that gives a shorter $s \rightsquigarrow v$ distance?

$s \rightsquigarrow w \to v \geq d'[v]$

$d[v] = \min_{\substack{(a,v)\in E \\ a \in S}} \{d[a] + wt(a,v)\}$.



Let $x \in S'$ and $(x,y) \in E$

$\to d'[v] \leq d'[y]$ | $\forall e \in E$, $w(e) > 0$, $y \neq v$.

$d'[y] = \min_{\substack{(w,y)\in E \\ w \in S}} \{d[w] + wt(w,y)\}$

$s \rightsquigarrow u \to v$

$d[s,u] + wt(u,v)$

$= d[u] + wt(u,v)$

$= d'[v]$

$s \rightsquigarrow x \to y \rightsquigarrow v$.

$d[x] + wt(x,y) + l(y,v)$

$\geq d'[y] + l(y,v)$.

$\geq d'[v] + l(y,v)$ ($l(y,v) > 0$)

$\geq d'[v]$.

$s \rightsquigarrow w \to v$.

$\geq d'[v]$

Say v attains min d' value through x.

$d'[v] = \min_{\substack{(a,v)\in E \\ a \in S}} \{d[a] + wt(a,v)\}$ → $d[x] + wt(x,v)$

$d[w] + wt(w,v)$ (w, x)

$d'[w] \geq d'[v]$

From these arguments, we get that $d[v]$ thus computed is the shortest $s \rightsquigarrow v$ distance.

Running time analysis:

1. Should d' be computed every time?

↳ in each iteration of the while loop, only the neighbours of the min vertex could have their values updated.

→ Extract min from the data structure which stores

d' values.

→ If v attains min value then $\deg(v)$ many updates are performed.

→ $(n-1)$ iterations of while loop → 1 Extract Min op
$O(n)$ Book keeping. → $\deg(v)$ updates

$O(n) + O(n-1)$ Extract Mins + $2m$ updates → $n-1$ Extract mins
$2m$ updates.