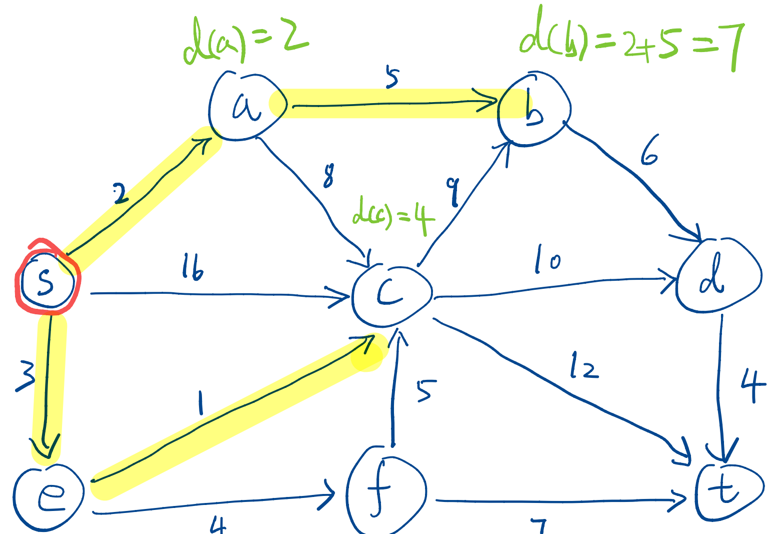
(IV) Shortest Path

Input: (directed or undirected) $G=(V,E)$, $s \in V$,

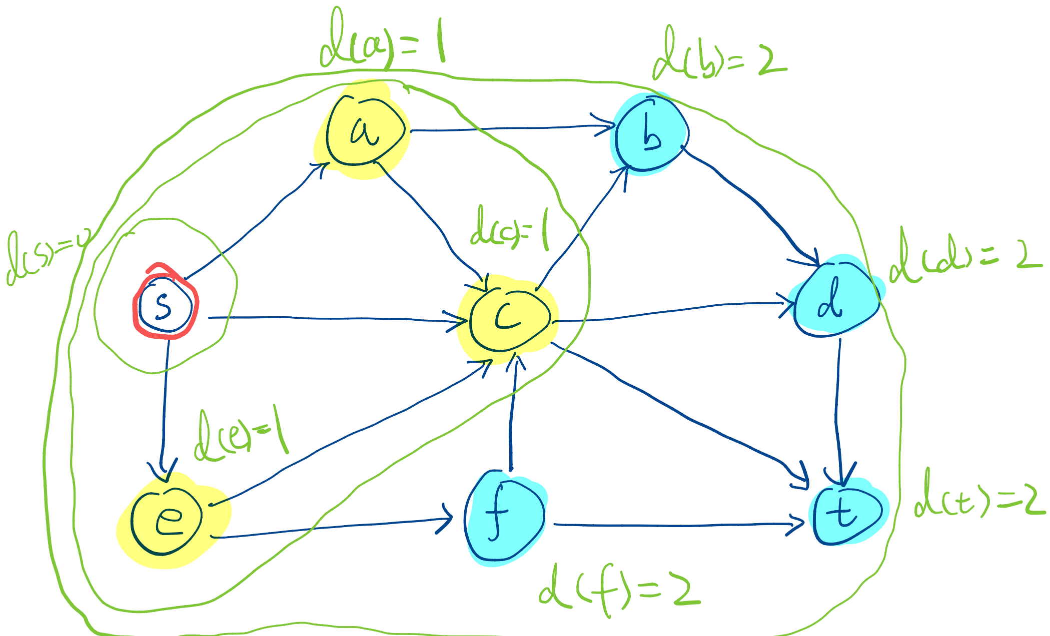
$w: E \mapsto \mathbb{R}_{\geq 0}$

Output: Shortest path from s to all other vertices.



▷ Warm-up: when $w(u,v) \equiv 1$.

- BFS solves this problem

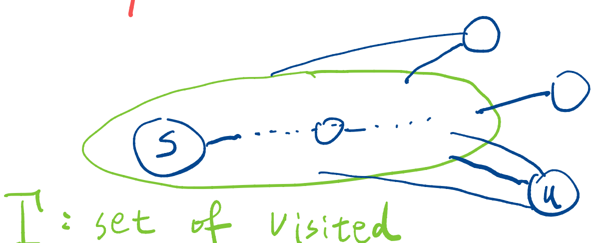


▷ Dijkstra's Alg: for arbitrary non-negative $w: E \mapsto \mathbb{R}_{\geq 0}$

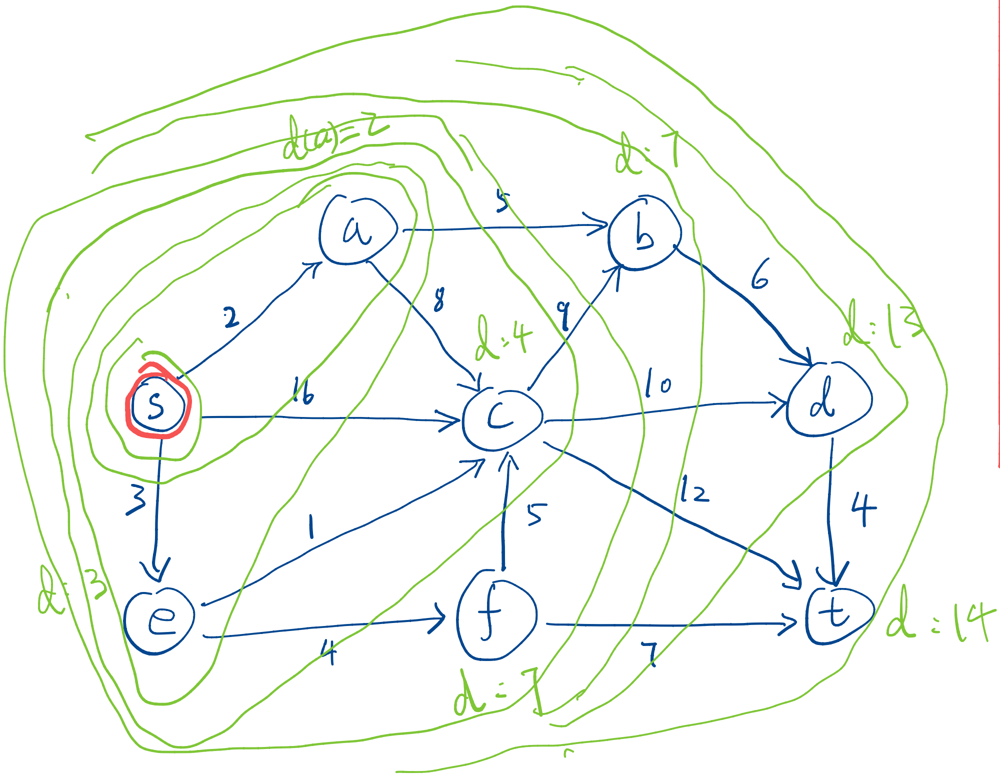
- Generalizing the BFS alg for the $w \equiv 1$ case
- Starting from s, explore all vertices from closest to farthest.
- Greedy choice of the next vertex u to visit:

  (1) u should neighbor the set of visited vertices

  (2) The length of the shortest (s,u)-path that goes through only visited vertices is minimized



$$u = \underset{v \text{ neighboring } I}{\arg\min} \left\{ \min_{x \in I} d(x) + w(x,v) \right\}$$



Alg-Dijkstra:

$\Gamma \leftarrow \phi$

$d(s) \leftarrow 0$

$d(v) \leftarrow \infty, \ \forall v \neq s$

while $\Gamma \neq V$:

$\quad u \leftarrow \underset{v \in V \setminus \Gamma}{\operatorname{argmin}} \ d(v)$

$\quad \Gamma \leftarrow \Gamma \cup \{u\}$

$\quad$ for all $v \in V \setminus \Gamma$ s.t. $(u,v) \in E$:

$\quad\quad d(v) \leftarrow \min\{d(u) + w(u,v), d(v)\}$

return d

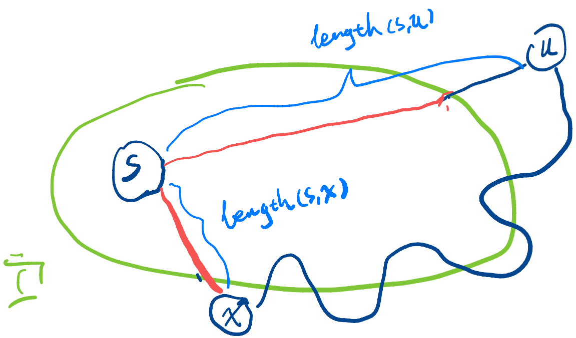
▷ Correctness for Dijkstra's Alg:

I: current set of visited vetices

u: the next vertex that's going to be added to I

$d(v)$: length of the shortest (s,v)-path that goes over only vertices in I.

Claim 1: When we add u to I, $d(u)$ is the shortest path length from s to u.



by def of u:

$\text{length}(s,x) \geq \text{length}(s,u)$

Alg-Dijkstra:

$I \leftarrow \phi$

$d(s) \leftarrow 0$

$d(v) \leftarrow \infty, \ \forall v \neq s$

while $I \neq V$:

  $u \leftarrow \underset{v \in V \setminus I}{\operatorname{argmin}} \ d(v)$

  $I \leftarrow I \cup \{u\}$

  for all $v \in V \setminus I$ s.t. $(u,v) \in E$:

    $d(v) \leftarrow \min\{d(u) + w(u,v), \ d(v)\}$

return $d(\cdot)$

Thm: At the end of Alg-Dijkstra, $\forall v$, $d(v)$ is the length of the shortest (s, v)-path.

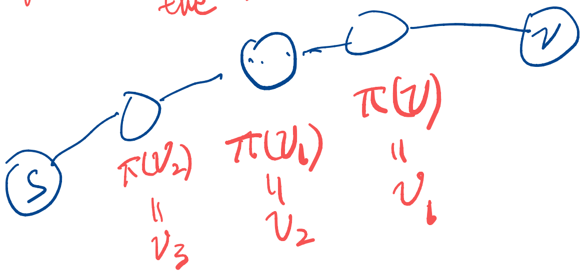
Pf: Since whenever an vertex u is added to I, by Claim I, we know $d(u)$ = length of shortest (s,u)-path.

When Alg-Dijkstra finishes, we have $I = V$. $\Rightarrow \forall v \in V$, $d(v)$ is the length of shortest (s,v)-path.

$\square$

▷ Implementation & Running Time

$\pi(v)$ = the predecessor of v on the shortest (s, v)-path.



Ex: show that $\pi(\pi(v))$ is also a vertex on the shortest (s, v)-path. In fact, the shortest $(s, \pi(v))$-path plus edge $(\pi(v), v)$ is exactly the shortest (s, v)-path.

Alg-Dijkstra:

$I \leftarrow \phi$

$d(s) \leftarrow 0$

$d(v) \leftarrow \infty, \ \forall v \neq s$

$\pi(s) \leftarrow \text{NULL}$

while $I \neq V$:

  $u \leftarrow \operatorname{argmin}_{v \in V \setminus I} d(v)$

  $I \leftarrow I \cup \{u\}$

  for all $v \in V \setminus I$ s.t. $(u, v) \in E$:

    If $d(u) + w(u, v) < d(v)$:

      $d(v) \leftarrow d(u) + w(u, v)$

      $\pi(v) \leftarrow u$

return $d(\cdot), \pi(\cdot)$

$n := |V|$, $m := |E|$ $(d(v), v)$

① Heap $\longrightarrow$ Priority Queue.

$O(1) \times n + O(\log n) \times n + O(\log n) \times m$

$= O(n \log n + m \log n)$

② Fibonacci Heap

$O(\log n)$ - time retrieving smallest key

$O(1)$ - time modification

$\Longrightarrow O(n \log n + m)$

```
Alg-Dijkstra:
  Γ ← φ
  d(s) ← 0
  d(v) ← ∞ , ∀ v ≠ s
  π(s) ← NULL
  while Γ ≠ V:
      u ← argmin_{v ∈ V\Γ} d(v)           O(1)
      Γ ← Γ ∪ {u}                         O(log n)
      for all v ∈ V\Γ s.t. (u,v) ∈ E:
          If d(u) + w(u,v) < d(v):
              d(v) ← d(u) + w(u,v)
              π(v) ← u                    O(log n)
  return d(·), π(·)
```