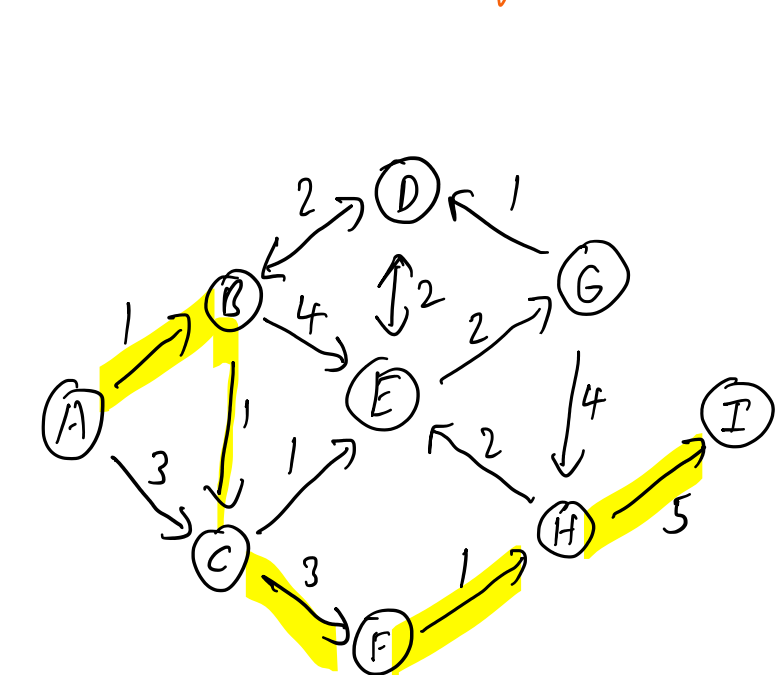
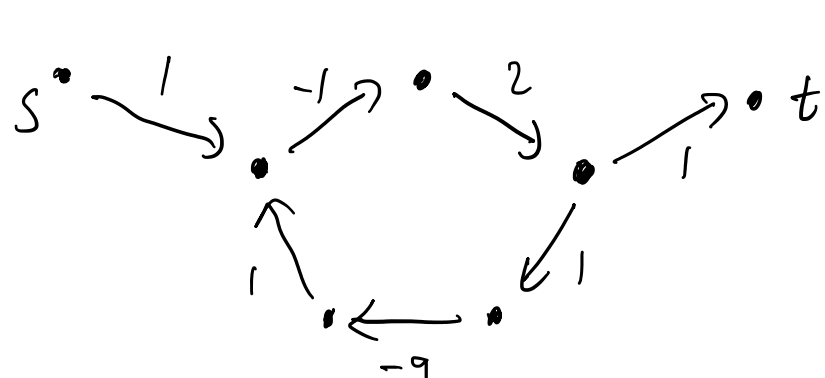
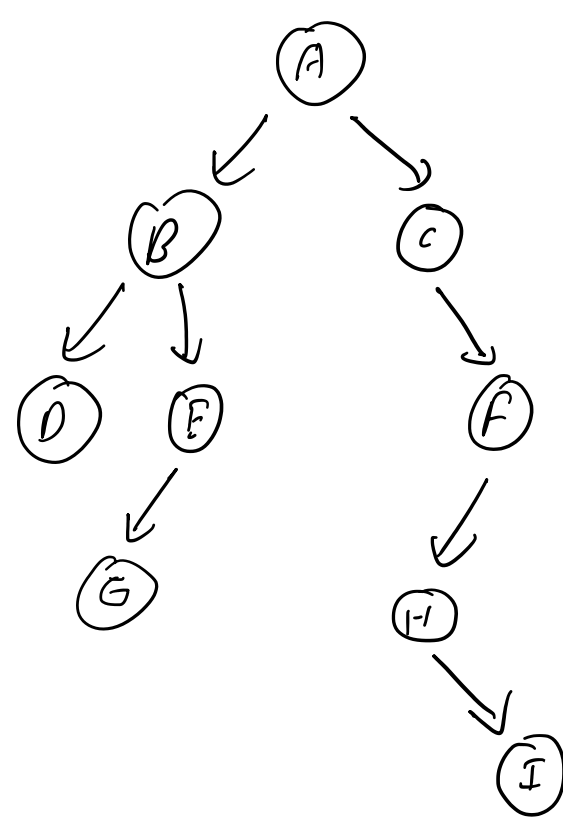


Shortest path in weighted, directed graph from starting node

(also analogous problem in undirected graphs, of course)



In unweighted graph, BFS is sufficient



Shortest path weight?

3 5 -3 ?

Harder to even define shortest path with negative weights, so we will only focus on positive weights for now.

Applications:

- Online maps & GPS navigation
- Routing systems (packets on networks)
- Drug-Pathway interactions
- Epidemiology (path of disease through social network)

Dijkstra: (1959)

Use tree-growing paradigm (start w/ node; keep adding frontier edge)

Recall: Prim's alg: nextEdge = frontier edge of min weight

Shortest path

- Let s be the starting node ($d[s] = 0$)
- Maintain an array $d[u]$ = length of shortest currently known $s \rightarrow u$ path.
- nextEdge = frontier edge (u, v) that minimizes $d[u] + \text{length}(u, v)$
modification from Prim's

Pseudocode:

```

for  $u \in V$ ,  $d[u] = \infty$ ,  $p[u] = \emptyset$ ,  $F = V$ 
 $d[s] = 0$ 
while  $F \neq \emptyset$ :
     $u \leftarrow$  vertex in  $F$  with min  $d[u]$ 
    remove  $u$  from  $F$ 
    for each neighbor  $v$  of  $u$  in  $F$ :
        if  $d[u] + \text{length}(u, v) < d[v]$ :
             $p[v] = u$ 
             $d[v] = d[u] + \text{length}(u, v)$ 
return  $d[u]$ ,  $p[u]$ 
  
```

tentative distance $d[u]$, tentative parent $p[u]$, frontier F

$F = \text{makeheap}(V, d)$

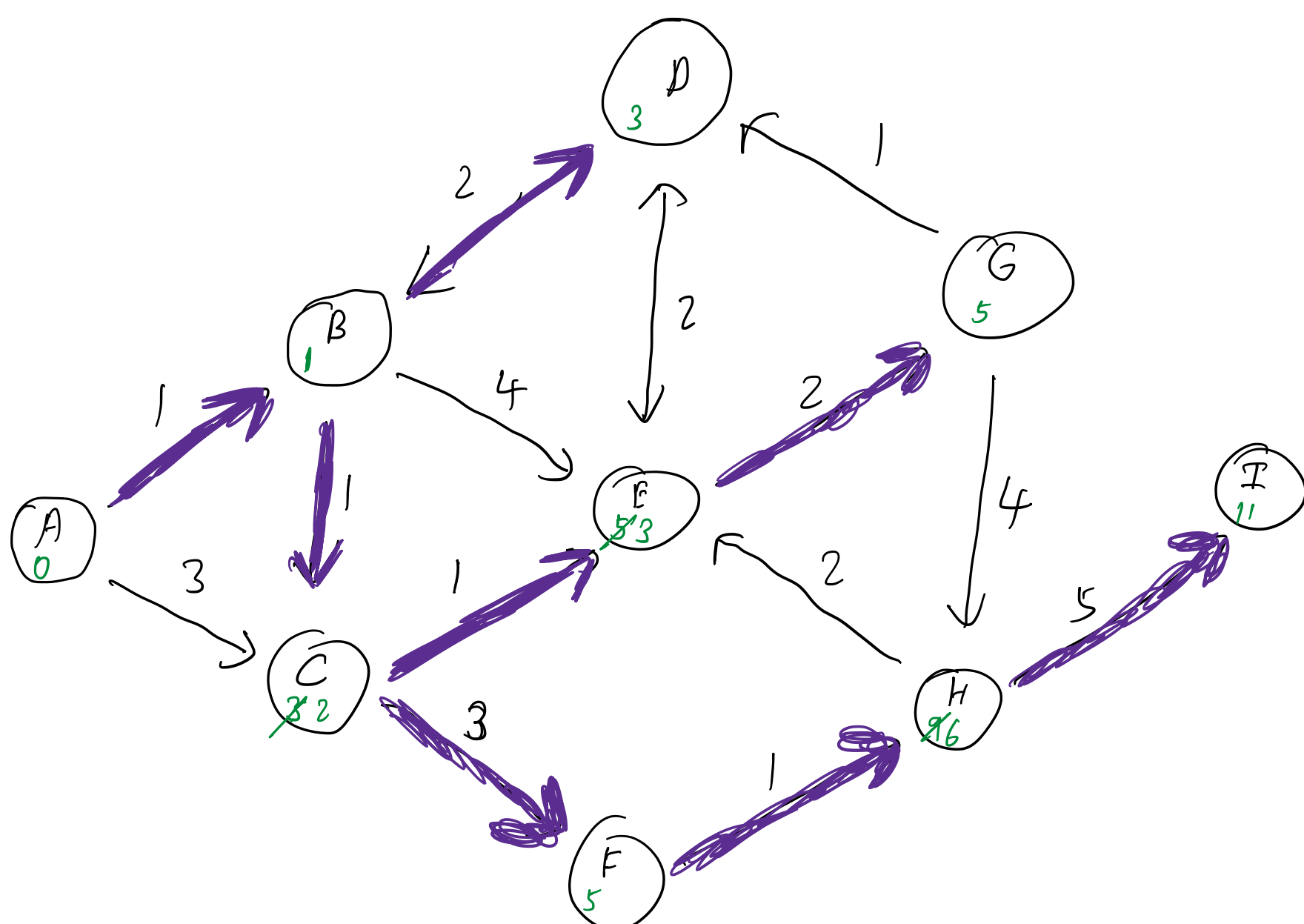
$\leftarrow \text{deletemin}()$ in heap
always explore next nearest node

reduce tentative distances + choose new parent
reduce key of heap

Example run:

(distances)

Dijkstra tree



Correctness:

Let T be the set of nodes explored at some point during the alg.

$\forall u \in T$, the path found by Dijkstra is the shortest.

proof. By induction on the size of T .

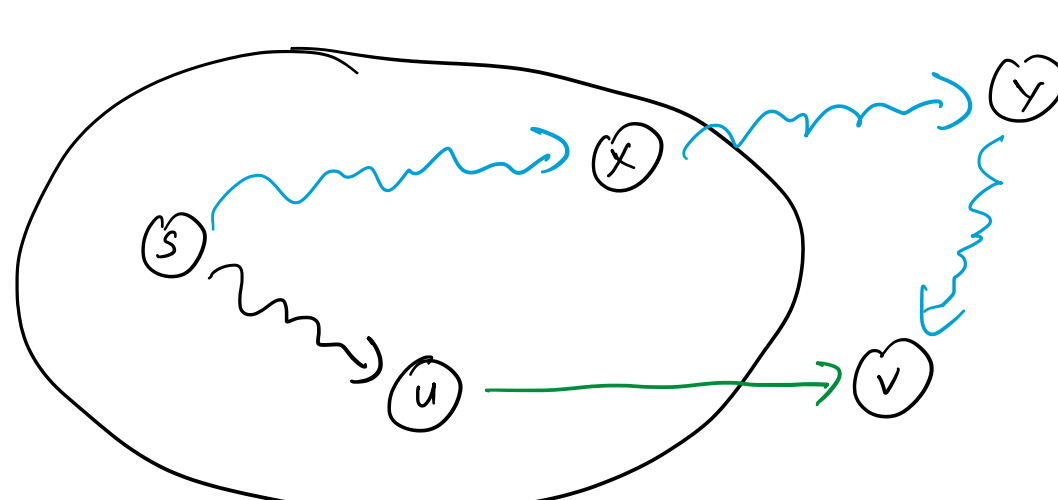
Base case: $|T| = 1$, so $|T| = \{s\}$, $d[s] = 0$, so correct.

Induction hypo: Assume theorem is true when $|T| \leq k$.

Let v be the $(k+1)$ st node added using edge (u, v) .

Let P_v be the path chosen by Dijkstra.

Let P' be any other $s \rightarrow v$ path (blue) (must go through some y not already explored)



$\text{length}(s \rightarrow u \rightarrow v) \leq \text{length}(\text{blue})$ by design of alg, since otherwise would have added y before adding v .

So, induction works. $\square$

Thm. $\exists$ optimal set of shortest paths from s whose union is a tree.

proof. Dijkstra works.

Runtime:

$O(|E| \log |V|)$

$O(|E| + |V| \log |V|)$

$O(|V| \log |E|)$

$O(|E| + |V|)$

Same as Prim's MST

every edge is processed once

Either:

(1) do nothing: $O(1)$

(2) reduce key in heap i : $O(\log |V|)$

total time: $O(|E| \log |V|)$

(using binary heap; can do better with other data structures)