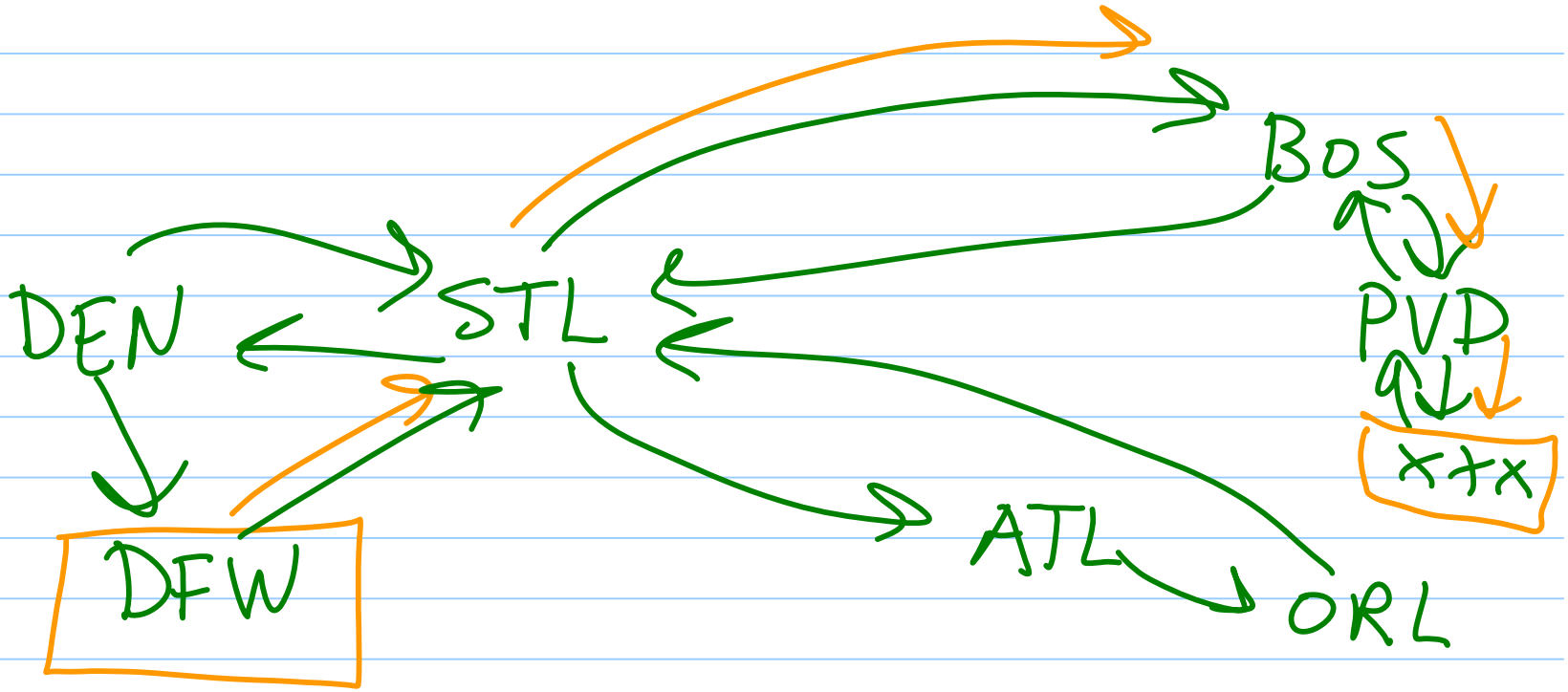
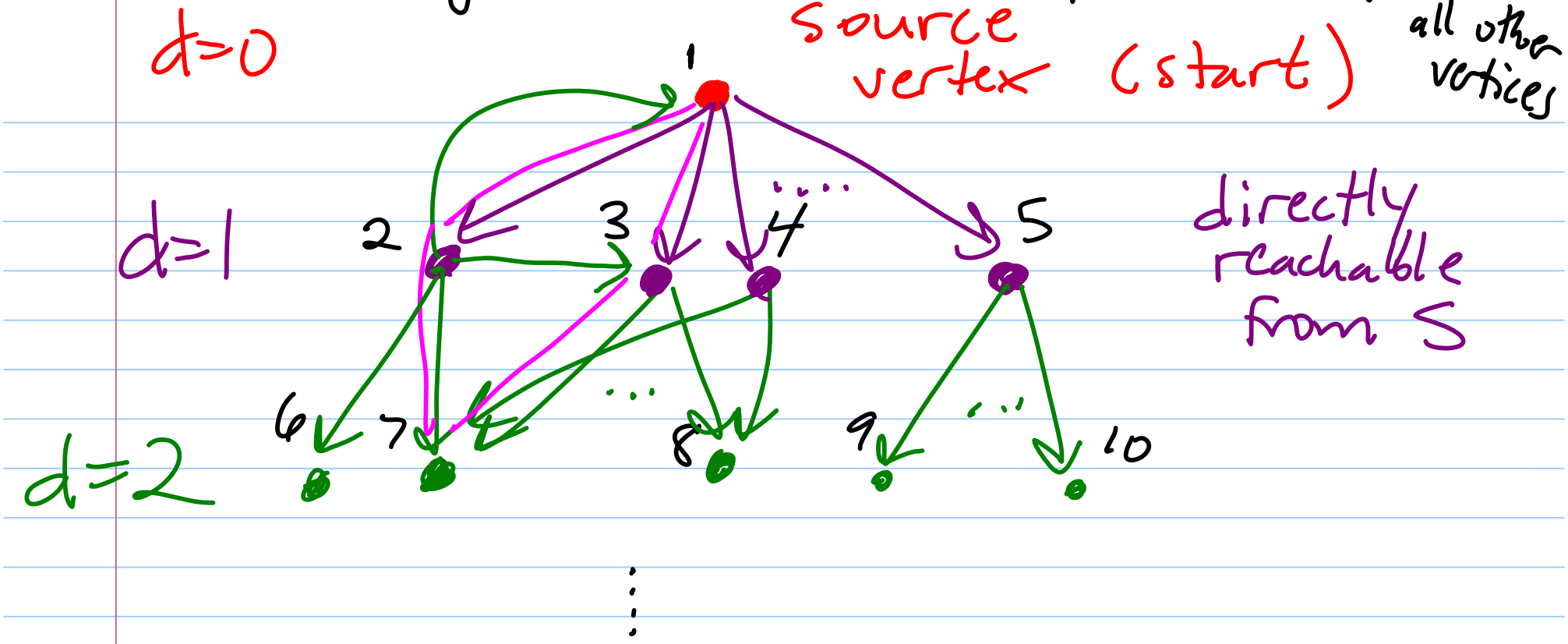
Finding Shortest Paths in Unweighted Directed Graph

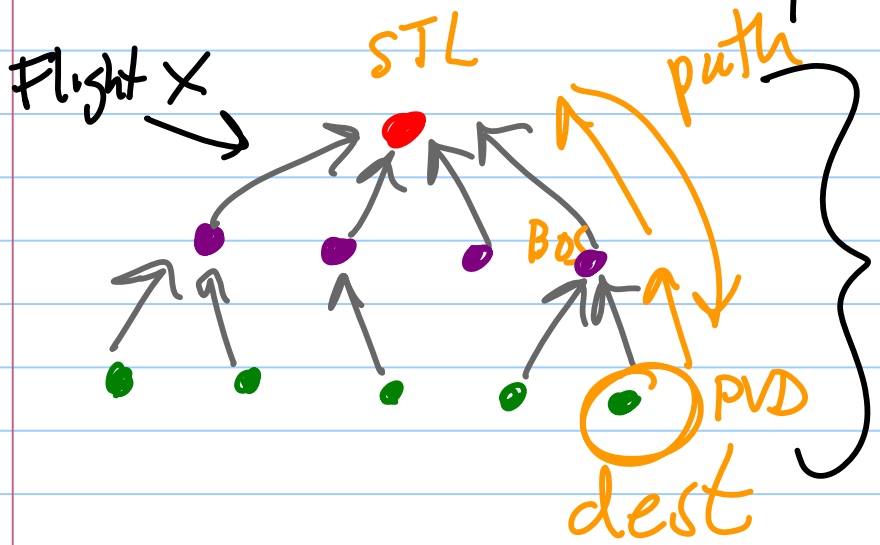


Single source shortest path - find path to all other vertices



Keep going until the desired destination is reached

How do we represent the solution?

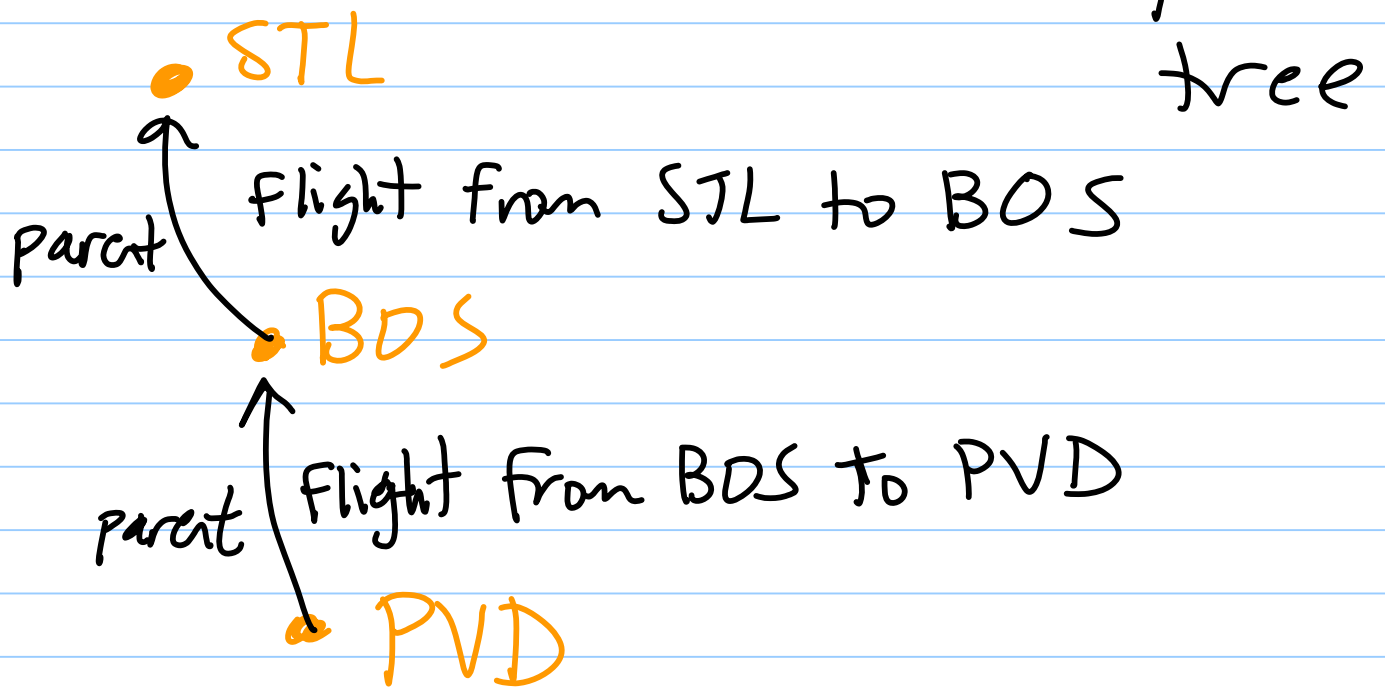


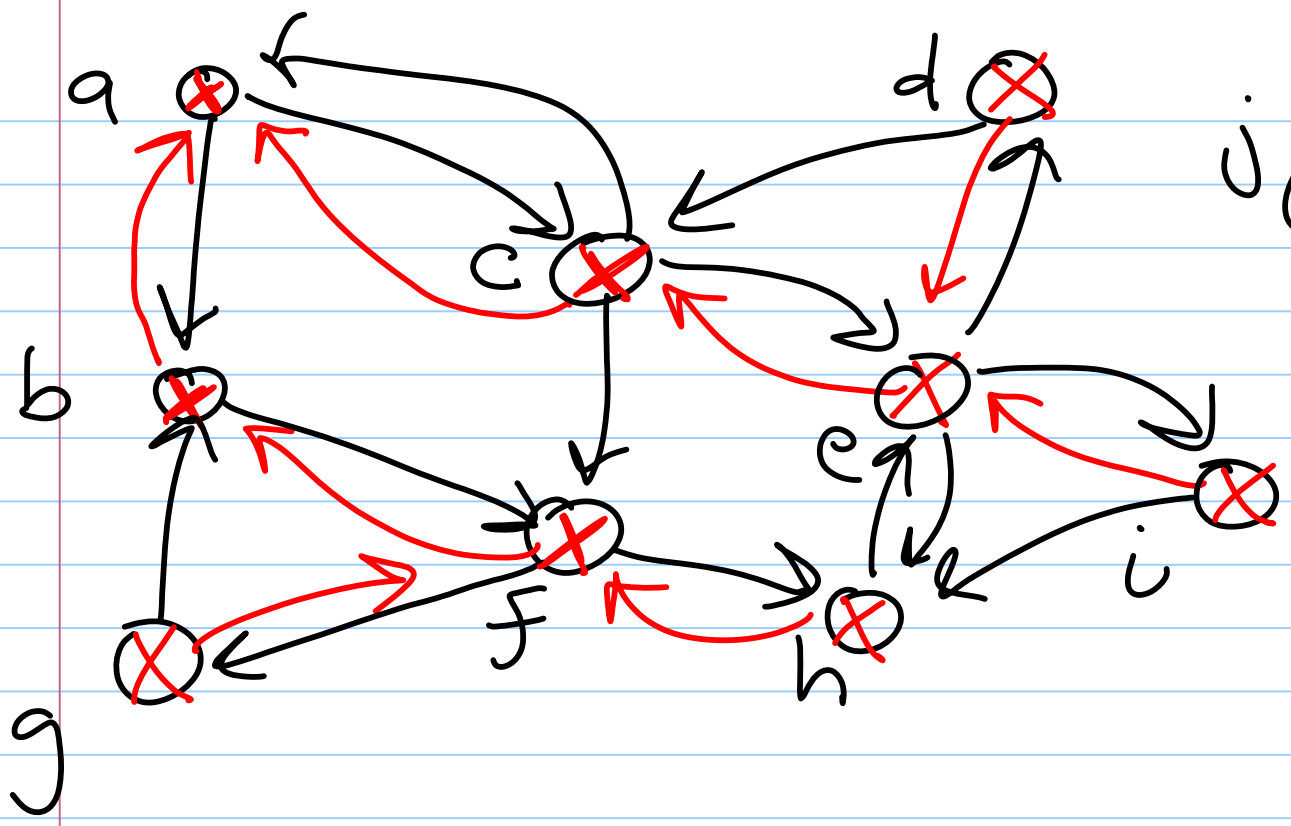
shortest path tree

Source is root

[represent implicitly by storing the edge from parent.

Shortest path tree



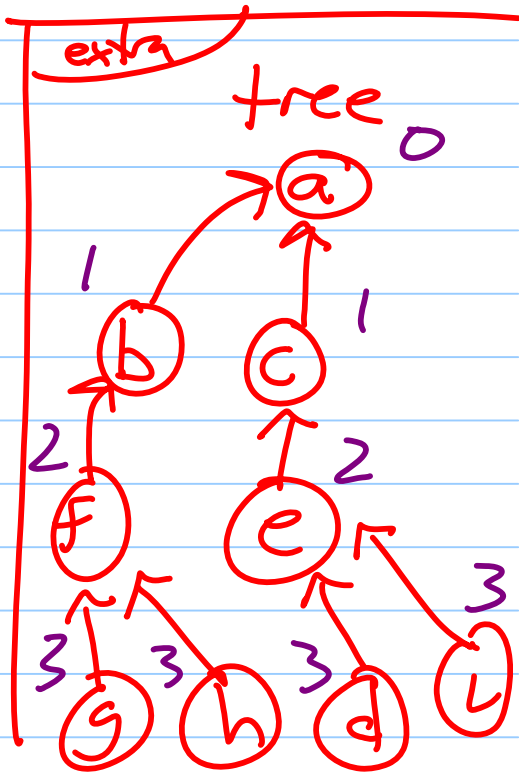


Source a

parent

extra

tree



Queue : ~~a b c f e g h d i~~

Time complexity

entire adj $O(n)$: every vertex is placed into & removed from queue ≤ 1 time

adj list $O(n+m)$: iterate over outedges of each vertex at most once each

$O(n+m)$ adj list, $O(n^2)$ adj matrix

Dijkstra's Shortest Path Algorithm

Note Title 11/20/2007

First let's review breadth first search (bfs) that solves the single-source shortest path algorithm for an unweighted graph

```
bfs(V source)
    For each u∈V
        u.discovered = False
        u.dist = ∞
        u.parentEdge = null
    source.discovered = true
    source.dist = 0
    queue.enqueue(source)
    while (!queue.isEmpty())
        u = q.dequeue()
        for each edge e in outgoing edges from u
            v = e.dest
            if (!v.discovered)
                v.discovered = true
                v.dist = u.dist + 1
                v.parentEdge = e
                queue.enqueue(v)
```

initialization

Upon completion of bFs

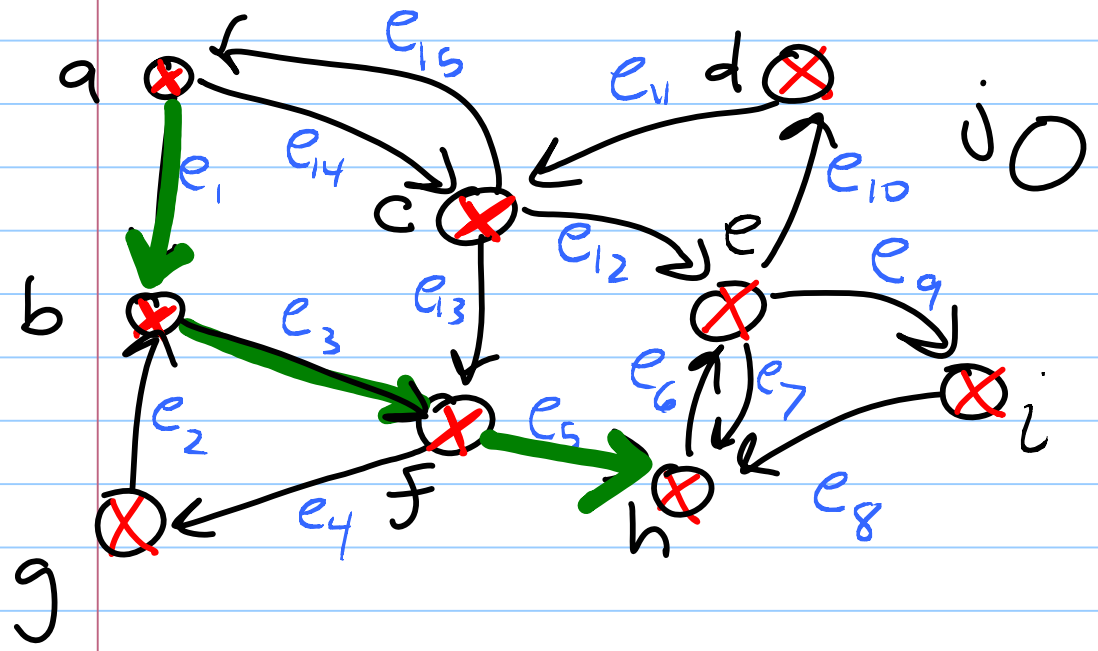
any vertices not discovered are not reachable from source

For all discovered vertices v
v.dist is # of edges in a shortest path from source to v

Following parent references from v to s gives a shortest path from s to v (in reverse order)

Time complexity O(n+m) with adj list

Example from last class

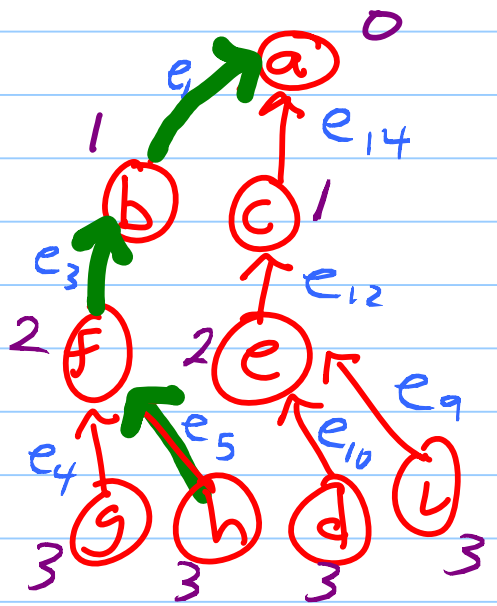


x means it was discovered

Shortest path from a to h

e_1, e_3, e_5

Shortest path tree



V.dist shown in purple