Graphs (DFS / Dijkstra)

Thursday, December 3, 2020 5:00 PM

Reminder: *lab 8 is due next Monday.
*HW 9 " " " Thursday.

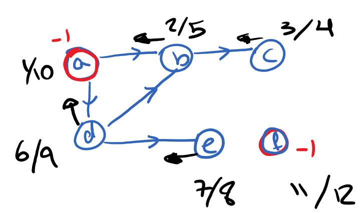
*Final is next Thursday (Dec 10th)

*lab 7 is due tonight.

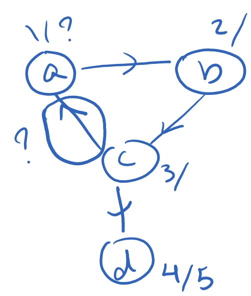
DFS: O(V+E)

Kristinann

```
time = 0
DFS(V)
[redacted]
for vi <- V
    if vi.parent = null
        vi.parent = -1
        DFS_visit(vi)
[redacted]
```

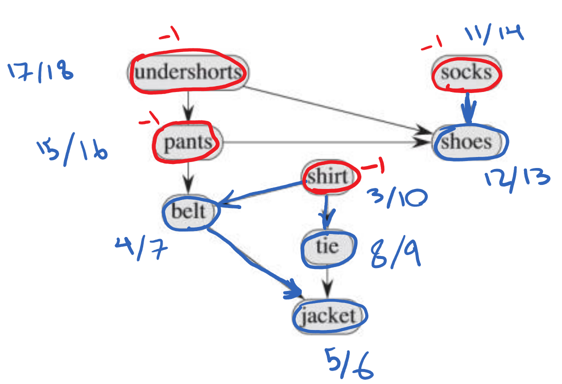


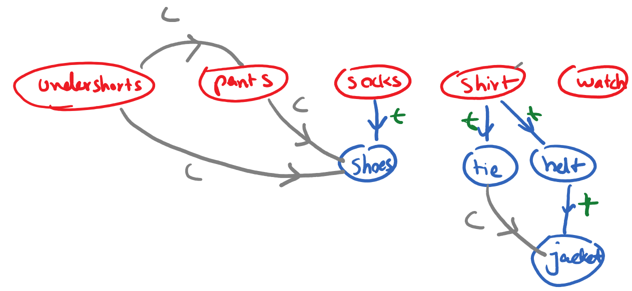

```
DFS_visit(vi)  -> time++
    vi.start = time
    for vj <- vi.adj
        if vj.parent = null
            vj.parent = vi

            DFS_visit(vj)
        else if vj.end = null
            print "cycle detected"
    time++
    vi.end = time
```



Topological Sorting / ordering: Only works for DAG (Directed Acyclic Graphs)

Example: watch → jacket → belt → tie → shirt → shoes → socks → pants → undershorts

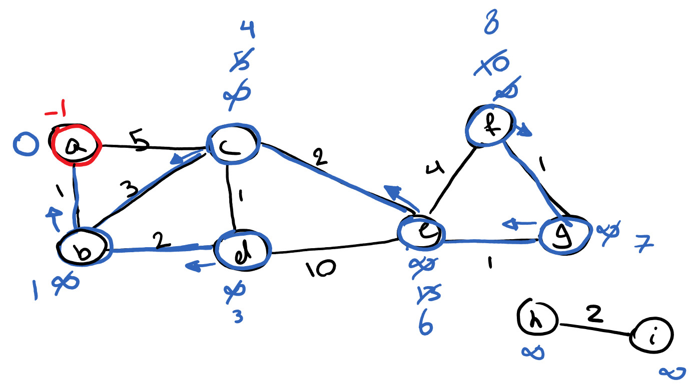




Dijkstra's algorithm: Finds the shortest distance from one initial vertex to all the reachable vertices.

Greedy Algorithm

Example:



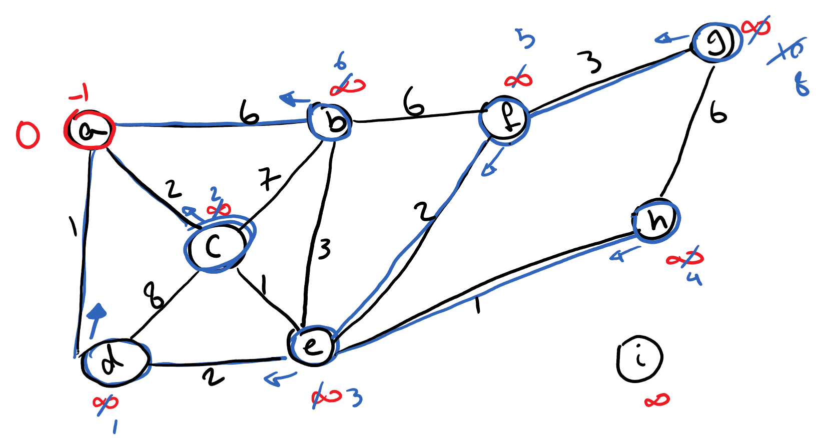
$a \to e$ (6) ✓

$e \to c \to b \to a$ ✓

a/b/d/c/e/g/f

order: a/d/c/e/h/f/b/g

Example



$v_i.dst = \inf$

Dijkstra(s)

s.p

s.dst

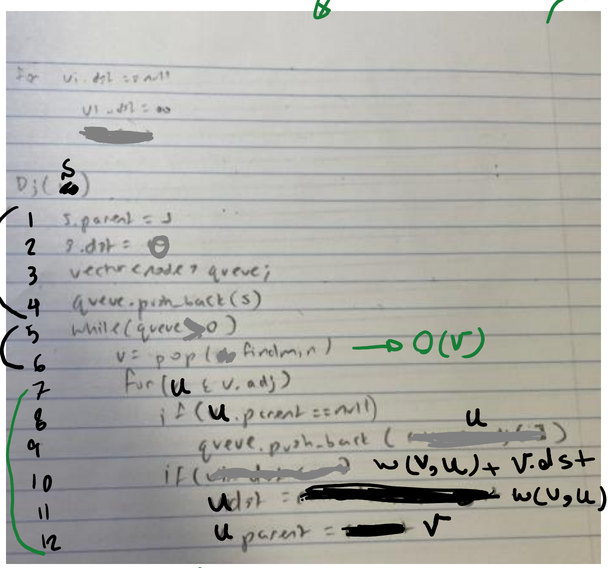
$w(e,d) = 7$

Q.push(s)

while Q.size > 0

Q.findmin.pop() (a)

Edward :)



```
for v: v.dst == null
    v.dst = ∞
[redacted]

Dj(s)
1  s.parent = s
2  s.dst = 0
3  vector<node> queue;
4  queue.push_back(s)
5  while(queue > 0)
6      v = pop(findmin)
7      for(u ∈ v.adj)
8          if(u.parent == null)
9              queue.push_back(u)
10         if(w(v,u) + v.dst < u.dst)
11             u.dst = [redacted] w(v,u) + v.dst
12             u.parent = [redacted] v
```

$\Theta(1)$ (lines 1–4)

$O(v^2)$ (lines 5–6)

$\deg(v)$ (lines 7–12)

$\to O(V)$

$\Theta(1) + O(V^2) + O(E) = \boxed{O(V^2 + E)}$

simple undir graph $\to O(V^2)$

$E \leq \frac{V(V-1)}{2}$

complete graph

$\overbrace{\text{line 5}}^{\text{while}} + \overbrace{\text{line 7–12}}^{\text{for}} = \sum_{i=1}^{|V|} \text{forloop} = \sum_{i=1}^{|V|} \deg(v_i) = \begin{cases} 2|E| & \text{undir} \\ |E| & \text{dir} \end{cases}$

while loop

)

Better version (using minheaps)

```
Dj(s)
    s.parent = s
    s.dst = 0
    ~~minheap.insert(s)~~ → Buildminheap(V)  all vertices
    while(minheap.size > 0)
        v = minheap.removeRoot → O(log V)
    for u ∈ v.adj
        if(u.parent == null)
            minheap.insert(u) → O(log V)
        if(w(v,u) + v.dst < u.dst)
            u.dst = w(v,u) + v.dst
```

$\Theta(1)$ / $O(V)$

$O(V\log V)$

while + $O(E\log V)$

$\Rightarrow \cancel{\Theta(1)}\ O(V) + O(V\log V) + O(E\log V)$

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

$= \boxed{O((V+E)\log V)}$ ✓

$O(E\log V)$

if (w(v,u) + v.dst < u.dst)

u.dst = w(v,u) + v.dst

u.parent = v

minheapify (u) $\longrightarrow O(\log V)$