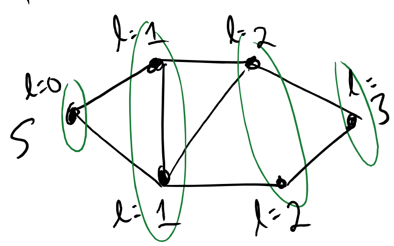
S.KIMMEL

Shortest Paths

Input: Graph $G=(V,E)$, $s \in V$

Output: $\forall\ v \in V$, $\ell(v)$ = shortest path from s to v

$\ell(v) = \infty$ if s, v not connected



Applications: Bacon #
Linked in Degree

Idea, explore layers.

$\ell[v] = \infty \quad \forall\ v \in V$ // will store shortest paths

$Ex[v] = \text{False} \quad \forall\ v \in V$ // mark True when "explored"

$A = \{\}$; ← What type of data structure is A?

A.add(s)

★ QUEUE: FIFO ★

$\ell[s] = 0$

or STACK: FILO

$Ex[s] = \text{True}$

Without red: BFS,
With red: BFS for shortest path

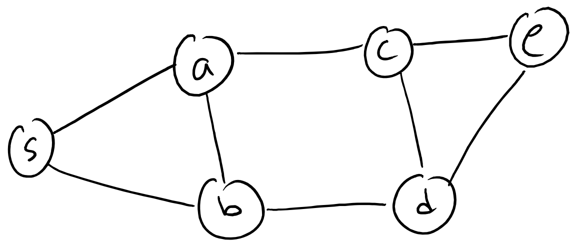
while(A is not empty) {
    v = A.pop

    For each edge (v,w) {
        If(Ex[w] = false) A.add(w) ; Ex[v] = true ; $\ell[w] = \ell[v] + 1$;
    }
}

This is Breadth First Search - slowly move away in layers from initial node

S. KIMMEL

ex:



A

~~s~~
~~a~~
~~b~~
~~c~~
~~d~~
e

ℓ

s = 0	T
a = 1	F
b = 1	F
c = 2	F
d = 2	F
e = 3	F

Now need to figure out...

- is it correct?
- what is the runtime

Q: What strategy can you use to prove correct?

Discuss...

S.KIMMEL

Q:

What is runtime of BFS using adjacency list if n is total # vertices, m is total # edges, n_s is # of vertices reachable from s, m_s is # edges reachable from s.

A. $O(m_s)$ B $O(n+m_s)$ C. $O(n_s \cdot m_s)$

D. $O(n+n_s m_s)$

Answer: B.

1: initializing $Exp[v]$ $\forall v \in V$ takes time $O(n)$

2. (After Initialization)

Seems like should be $n_s \cdot m_s$ b/c while loop runs $O(n_s)$ times, for loop runs $O(m_s)$ times. Can do better!

$\Rightarrow$ Inside while loop, do a constant # of operation for each edge that is examined. We only examine edges of vertices that end up in queue. Each edge only has two vertices connected to it. Each vertex can only be in queue once. (Only get to edges connected to s.) $\Rightarrow$ $\sim 2m_s$ operations

S.KIMMEL

What kind of algorithm is BFS shortest path?

Greedy

Local Search

- look for best choice in nearby area

Dynamic - Since stores solutions to previous problems, one could view as dynamic... but most textbooks would <u>NOT</u> describe as dynamic