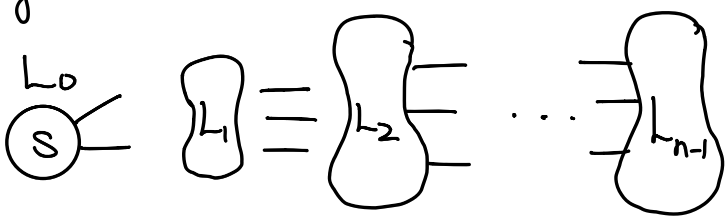
Lecture 34

Breadth-First Search (BFS)

BFS Intuition - Explore outward from source vertex (s) of a graph $G = (V, E)$ in all possible directions, adding vertices one "layer" or "level" at a time



BFS is used for both directed & undirected graphs

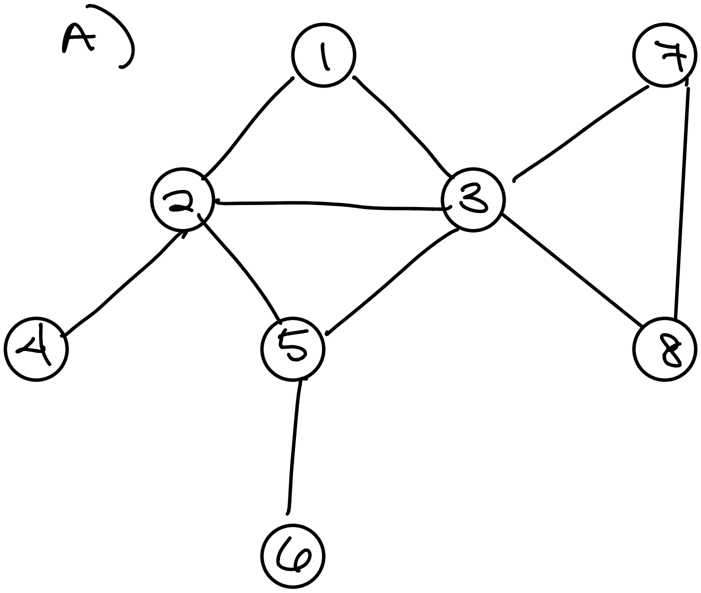
BFS Algorithm Outline

* $L_0 = \{s\}$
* L_1 = all neighbors of L_0
* L_2 = all vertices that do not belong to L_0 or L_1 & and that have an edge to a vertex in L_1
* L_{i+1} = all verticies that do not belong to an earlier layer, and that have an edge to a vertex in L_i

Note:

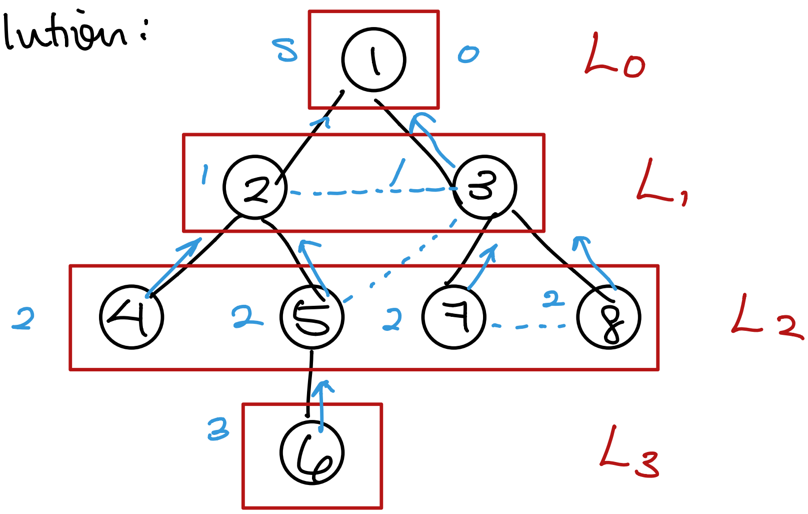
Ex. Use BFS to determine the smallest # of layers or hops b/w the vertices starting from vertex s

A)

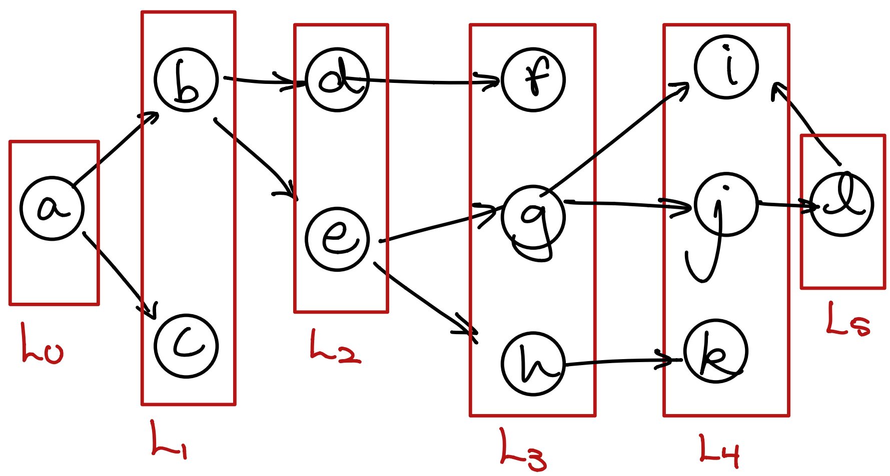


Undirected

Solution:

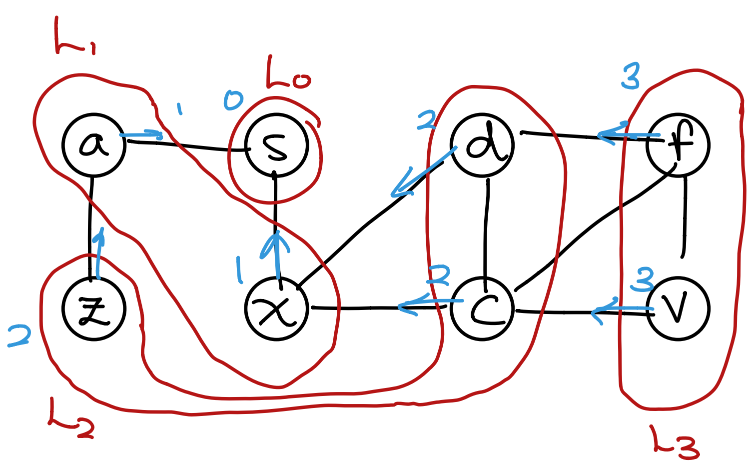


(B) Directed graph



Strongly connected

(C) Undirected Graph



Handout #5

$frontier_0 = \{s\}$

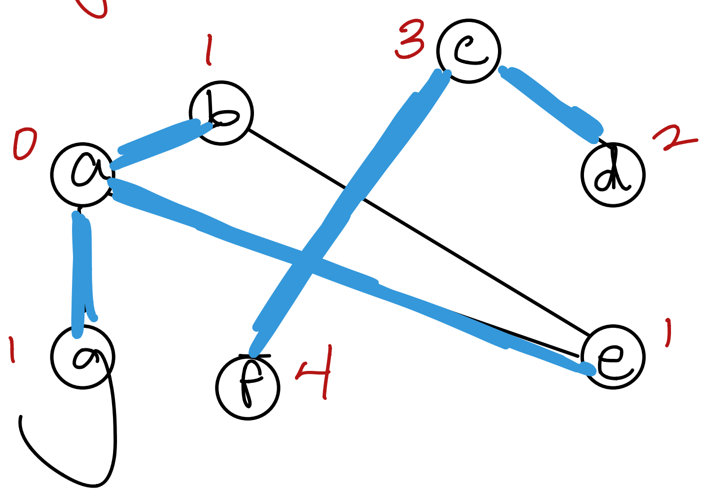
$frontier_1 = \{a, x\}$

$frontier_2 = \{z, d, c\}$

$frontier_3 = \{f, v\}$

Thm: The above implementation of BFS runs in $O(|V|+|E|)$ time if the graph is given by its adjacency representation
or $O(V+E)$

Ex. For the following graph, draw a BFS tree using alphabetical ordering (starting at a). Then list the edges in the order selected.



Edge List: (a,b), (a,e) (a,g)
(b,d)
(d,c)
(c,f)

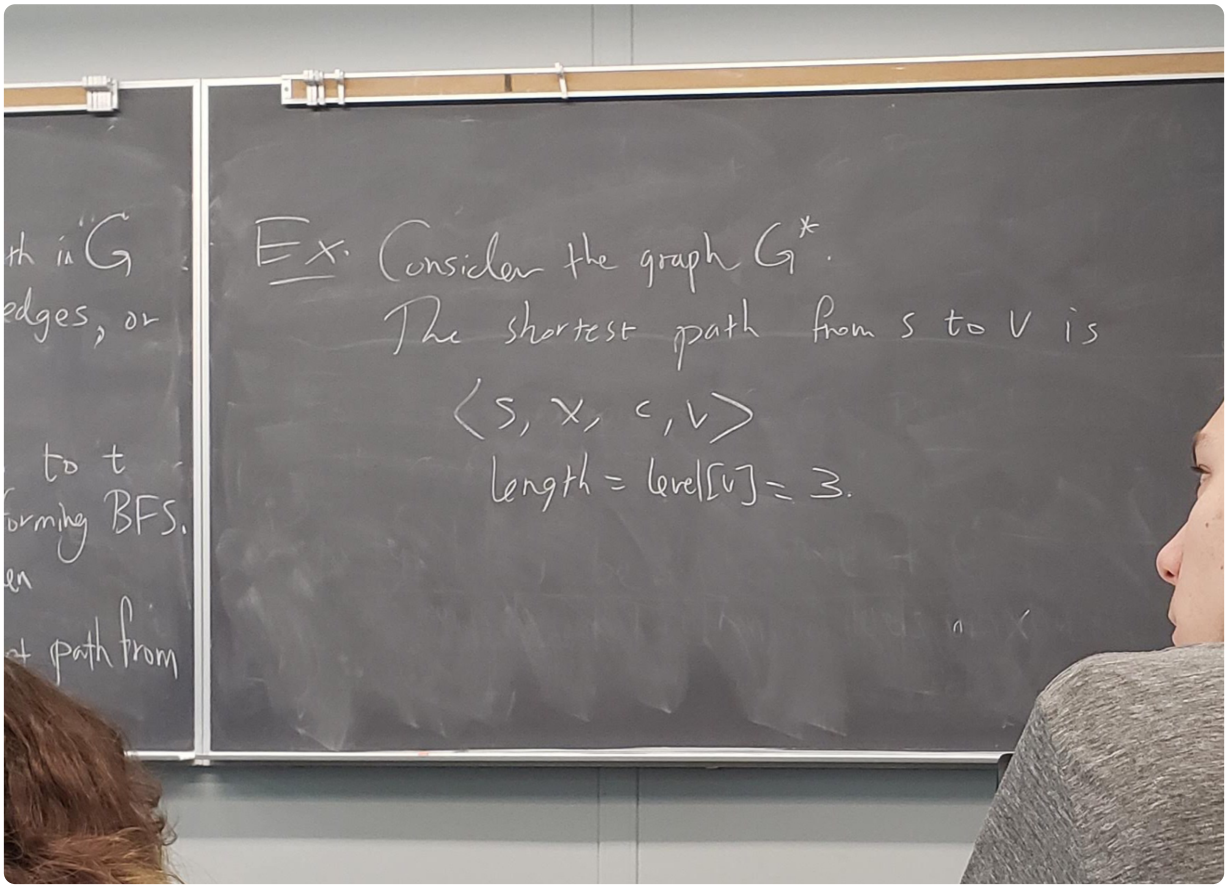
An application: Shortest Path

Problem: Given 2 verticies of G, find a path in G b/w them w/ the minimum # of edges, or report that no such path exists

FACT: There is a path from s to t IFF t appears in some layer while performing BFS.

Idea: We perform BFS starting at vertex s, then <s, ..., parent [parent [v]], parent [v], v> is a shortest path from s to v

The length of this path is level[v]



Homework:

https://u.osu.edu/alzalg.1/files/2019/11/hw14.pdf