

COL106: Data Structures and Algorithms

Lecture 25

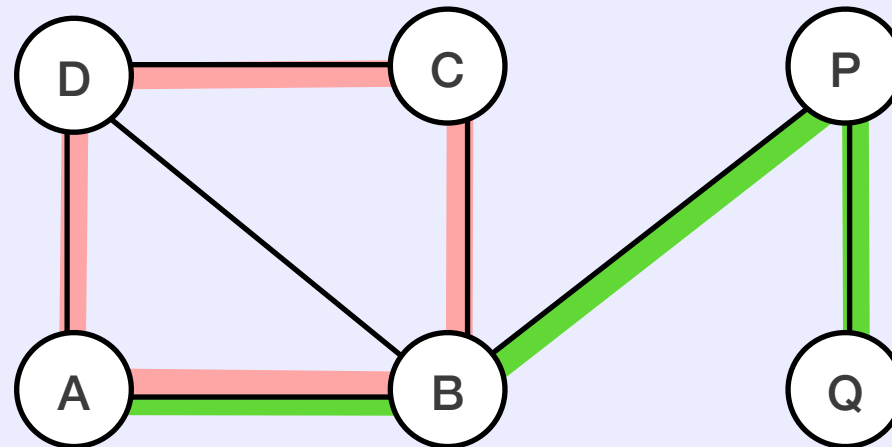
Keerti Choudhary
Department of Computer Science, IIT Delhi

Today's Lecture

1. Graph terminologies
2. Revisit BFS Trees
3. Application of BFS
 - Finding Shortest Path in unweighted graphs
 - Verifying Bipartite graphs
 - Computing smallest cycle in Graph

Graphs

A pair (V, E) of vertices and edges such that E is a subset of un-ordered pairs of V .



$$|V| = 6$$

$$|E| = 7$$

Path:

A sequence $(a_0, a_1, a_2, \dots, a_k)$ of two or more vertices such that

- (a_{i-1}, a_i) is edge for $i \leq k$.
- all vertices are distinct.

Eg: (A,B,P,Q)

Cycle:

A sequence $(a_0, a_1, a_2, \dots, a_k)$ of three or more vertices such that

- (a_{i-1}, a_i) is edge for $i \leq k$.
- $a_0 = a_k$, and all other vertices are distinct.

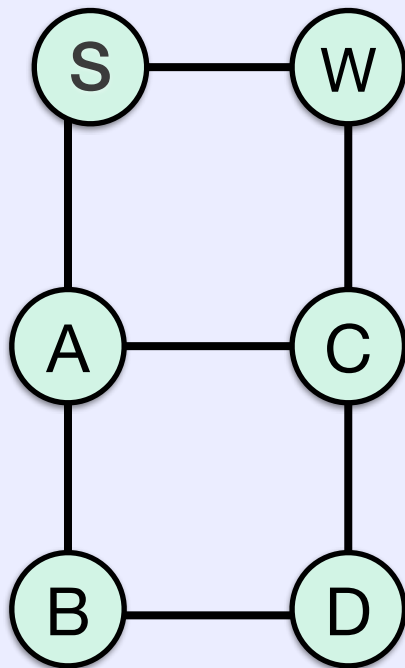
Eg: (A,B,C,D,A)

Length of a path : Number of edges in it

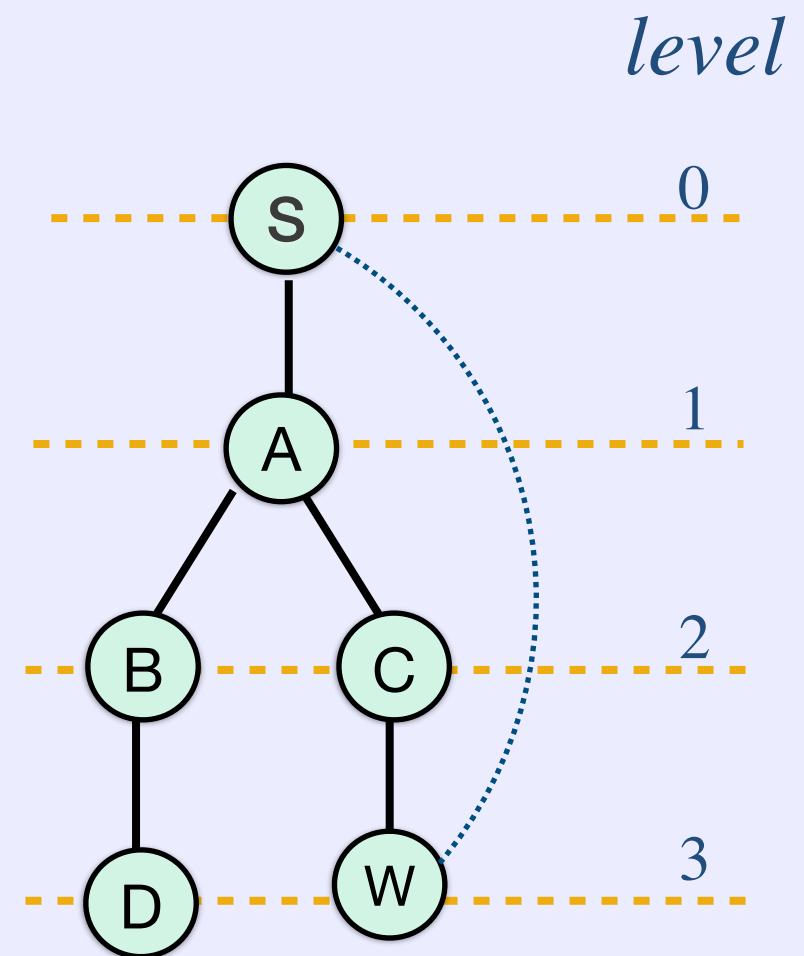
Length of a cycle : Number of edges in it

Spanning Tree

A spanning *tree* of a connected graph is a “spanning subgraph which is a tree”.



Graph G

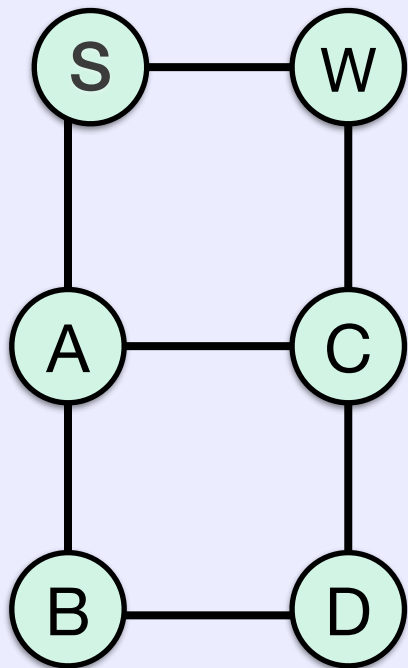


Spanning tree of G

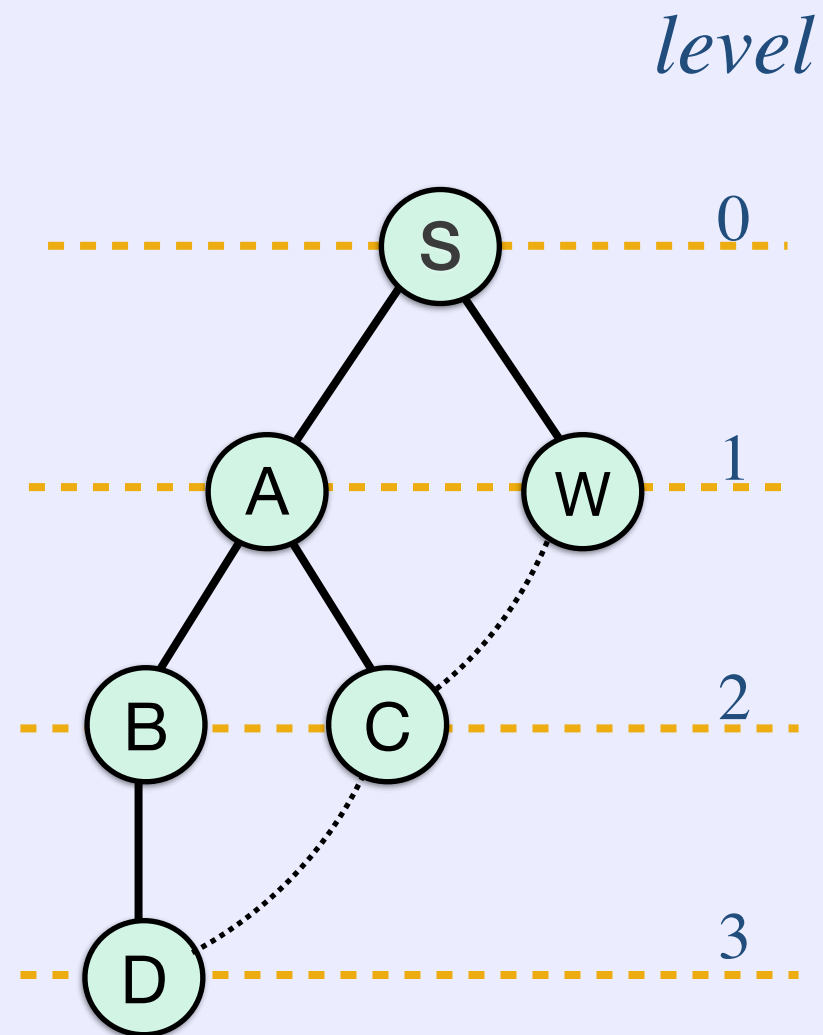
Question: Is this a BFS tree?

BFS Tree

A *tree* obtained by traversing vertices of graph by Breadth First Search traversal.



Graph G



BFS tree, T
along with non-tree edges.

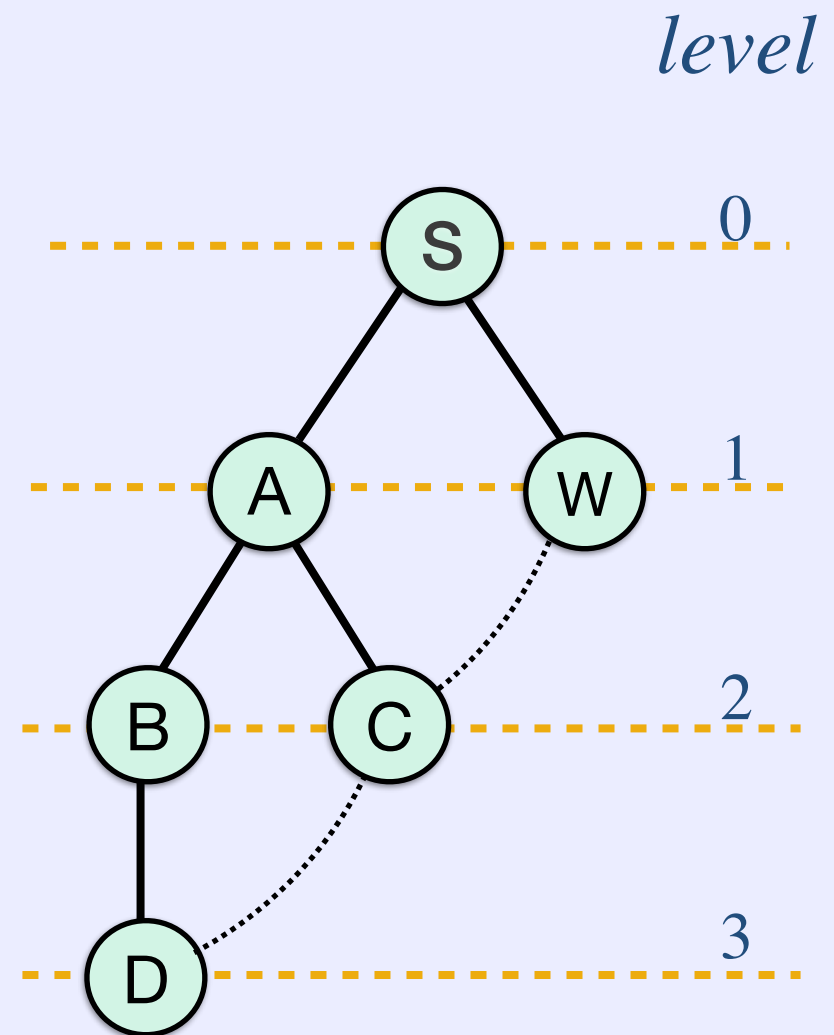
Recall : A BFS tree of a connected graph is computable in $O(|V| + |E|)$ time.

Application I: Finding Shortest Path

Theorem : If G is unweighted and T is a BFS tree rooted at s , then

Set of vertices
at level i in
BFS tree = Set of vertices
at distance i
from s

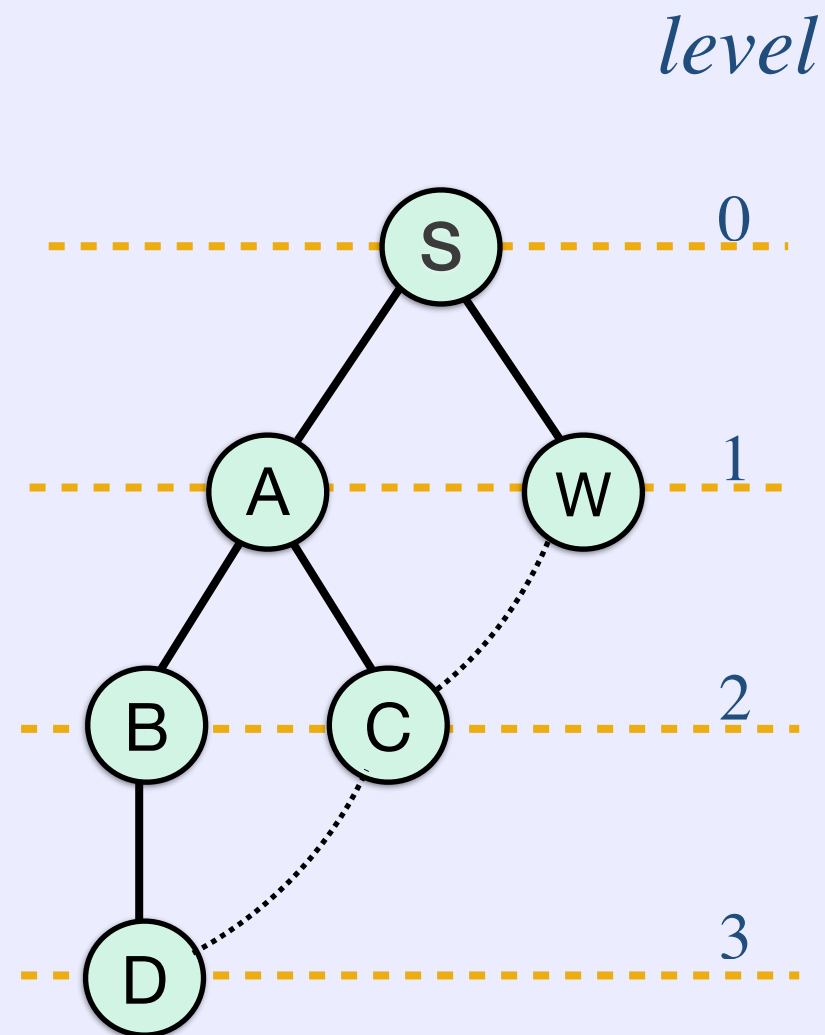
Proof: Use Induction.



BFS tree, T
along with non-tree edges.

Non-tree Edges with respect to BFS Tree

Claim: For any non-tree edge (x, y) , level of x and y differs by at most 1.



BFS tree, T
along with non-tree edges.

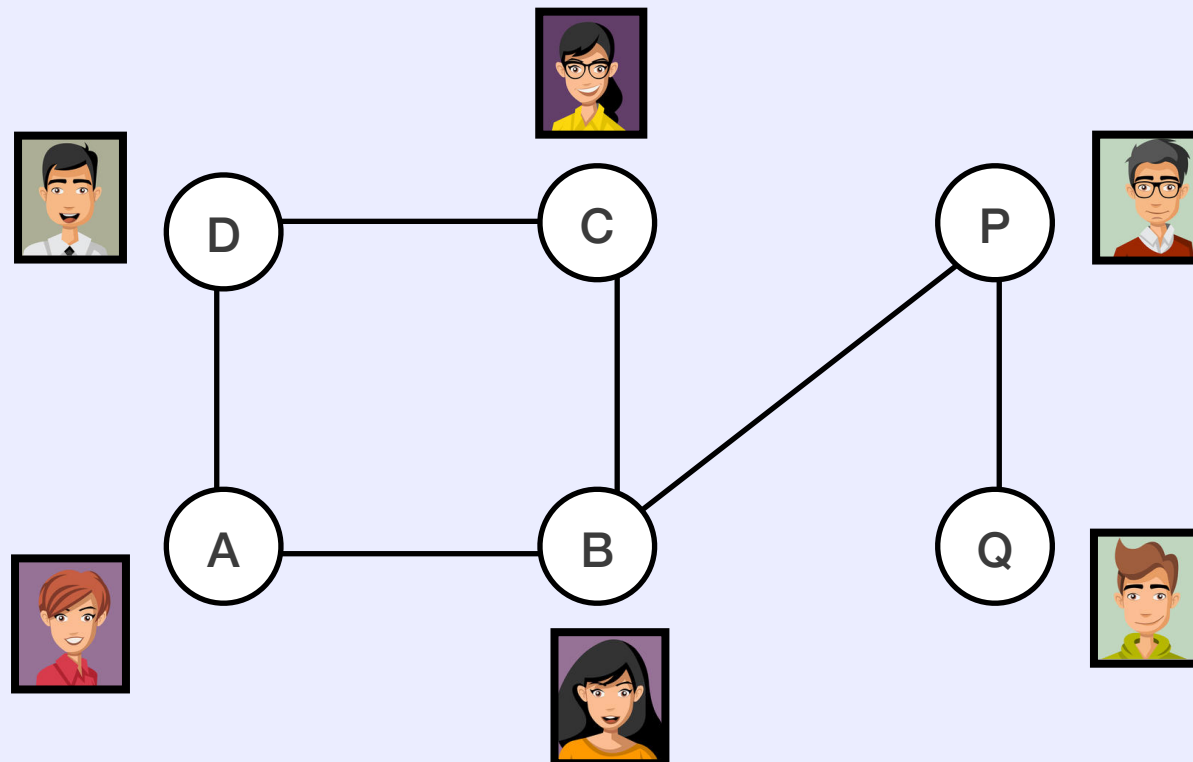
Application II: Graph Partitioning

Given: There are n students in COL106 course represented by vertices in a graph.
We have an edge between two students iff they are friends.

Question: Can you **partition (divide)** students into two classrooms X and Y , so that

- No two students in X are friends, and similarly
- No two students in Y are friends.

Example:



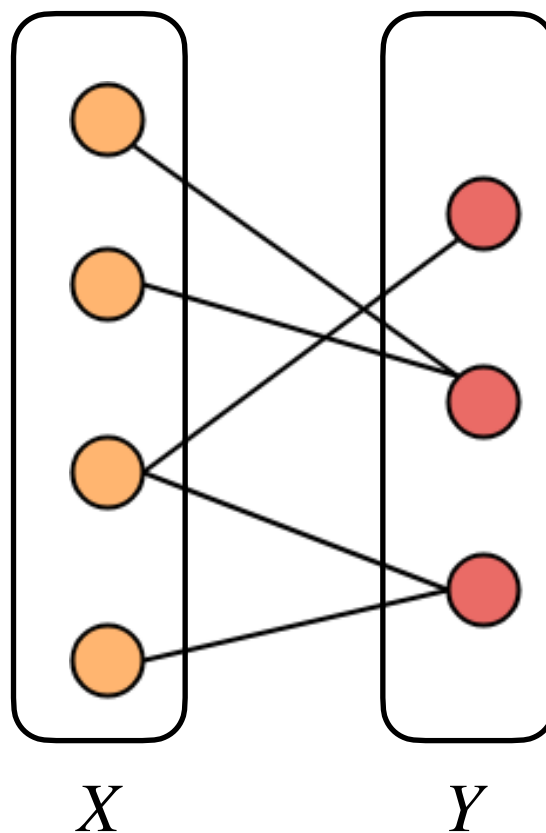
$$X = \{D, B, Q\}$$

$$Y = \{A, C, P\}$$

Bipartite Graph

Definition: A graph $G = (V, E)$ where V can be partitioned into two sets X and Y such that each edge in E has one endpoint in X and other endpoint in Y .

Example:



*How can you
check if a graph
is bipartite?*



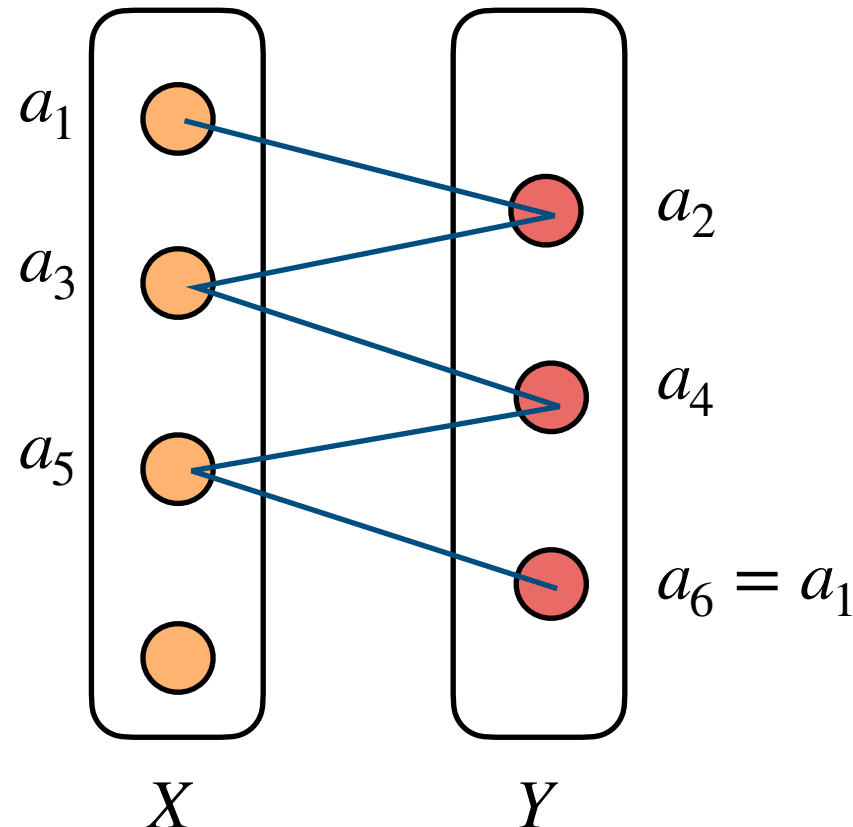
Odd length Cycles

Lemma: *A bipartite graph cannot contain cycles of odd length.*

Proof: Let G be a bipartite graph with partitions (X, Y) .

Suppose there is an odd length cycle, say $(a_1, a_2, a_3, a_4, a_5, a_6 = a_1)$.

Suppose $a_1 \in X$.



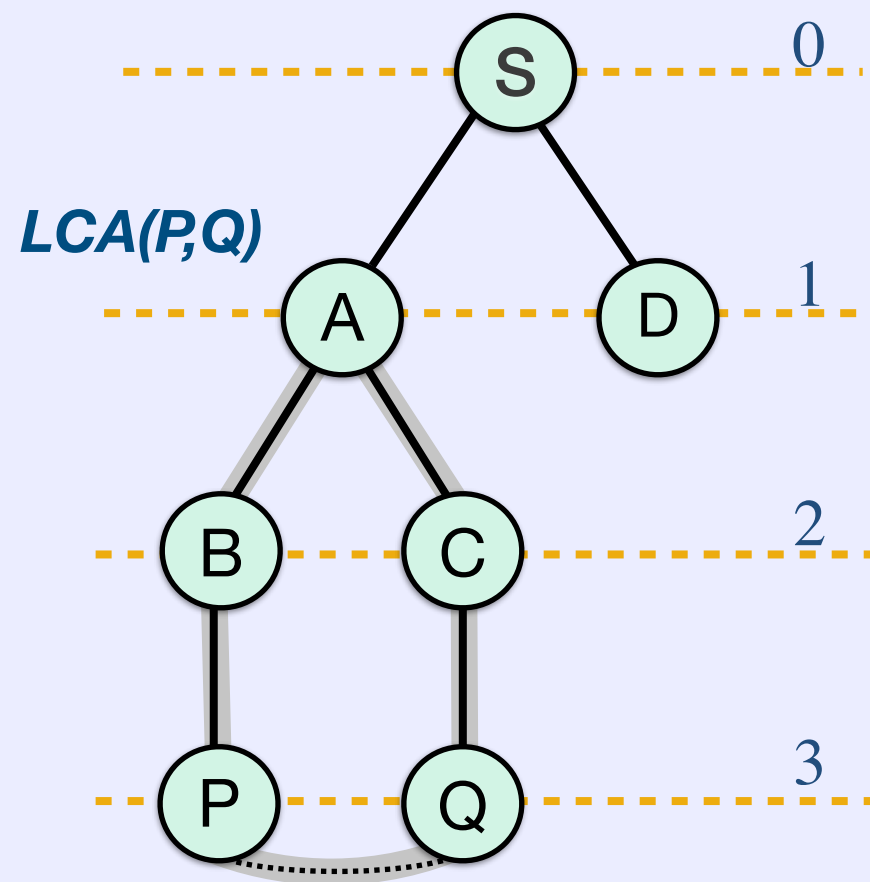
Contradiction!
Vertex $a_1 = a_6$
cannot lie in
both X and Y .

*You can argue
similarly for any
cycle of odd length!*

Verifying Bipartiteness

Theorem: Let T be a BFS tree of a graph G . Then, G is *bipartite* iff there is no non-tree edge with respect to T , whose endpoints are in same level.

Proof ($\Rightarrow$):



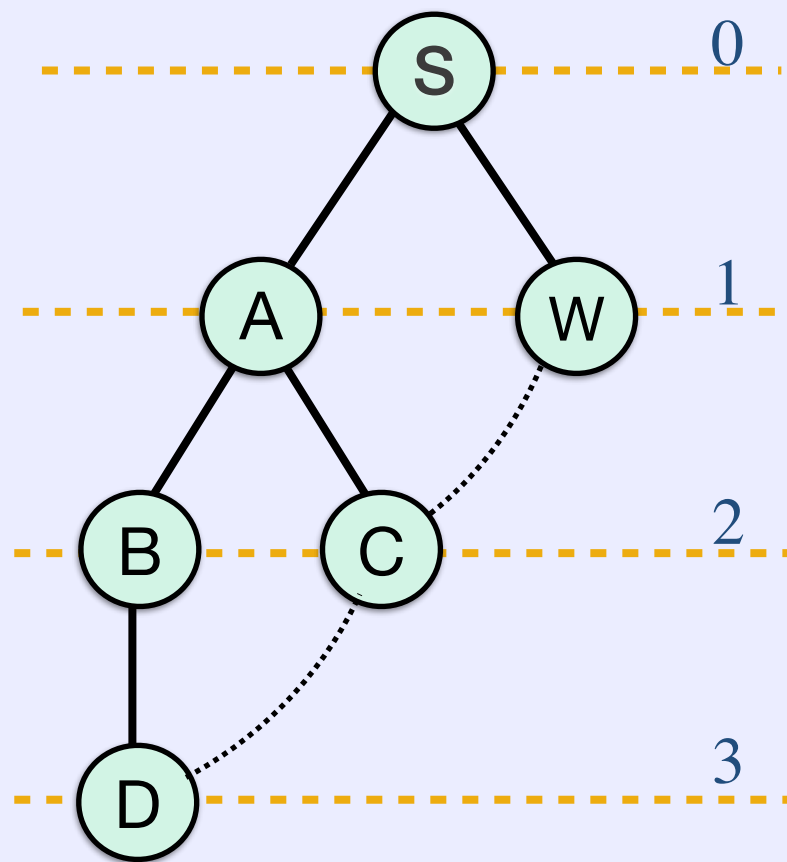
LCA: Lowest common ancestor

1. Let G be a bipartite.
2. Suppose there is an edge (P, Q) such that P, Q are in same level.
3. Consider the cycle formed by
 - Edge (P, Q) ,
 - Path from P to $LCA(P, Q)$
 - Path from Q to $LCA(P, Q)$
4. This is an odd length cycle which contradicts the that G is bipartite.

Verifying Bipartiteness

Theorem: Let T be a BFS tree of a graph G . Then, G is *bipartite* iff there is no non-tree edge with respect to T , whose endpoints are in same level.

Proof ($\Leftarrow$):



Put vertices in odd level in set X ,
and vertices in even level in set Y

Homework Questions

Question 1:

Prove that a graph is bipartite iff it doesn't contain any odd length cycles.

Question 2:

Prove or disprove the following:

“Let T be a spanning tree of G satisfying that for any non-tree edge (x, y) , levels of x and y differs by at most 1 with respect to T .

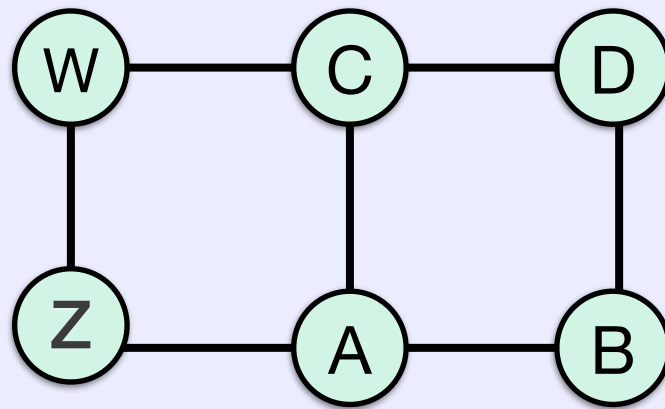
Then T must be a BFS tree of G .”

Question 3:

Prove that in worst case there can exist exponentially many BFS trees for a graph G .

Application III: Finding Smallest cycle

Goal: Given a graph $G = (V, E)$, find a cycle of smallest length in it.



Smallest cycle in G
has length 4.

We will show using BFS trees the smallest length cycle can be computed in

$O(|V| \cdot |E|)$ time

Homework Question

Question:

Let s be a vertex in G .

Design an algorithm to compute smallest cycle containing s in $O(|V| + |E|)$.

Hint: Use BFS(s)