Step 4 Complexity Analysis

1- Identify a free man $O(1)$

2- For a man $\underline{m}$, identify the highest ranked woman to whom he has not yet proposed. $O(1)$

3- For a woman $\underline{w}$, decide if $\underline{w}$ is engaged, and if so to whom $O(1)$

4- For a woman $\underline{w}$ and two men $\underline{m}$ & $\underline{m'}$, decide which man is preferred by $\underline{w}$

$O(1)$

5- Place a man back in the list of free men. $O(1)$

1. Identify a free man

	get	put
use an array	$O(1)$	$O(1)$
" a stack	$O(1)$	$O(1)$
" a queue	$O(1)$	$O(1)$
" linked list	$O(1)$	$O(1)$

2. Identify the highest ranked woman to whom <u>m</u> has not yet proposed.

Keep an array Next[1..n]

Next[m] points to the position of the next woman he will be proposing to on his preference list

Men's preference list: <u>ManPref[1..n, 1..n]</u>, where

ManPref[m, i] denotes the ith woman on man <u>m</u>'s preference list.

To find next woman <u>w</u> to whom <u>m</u> will be proposing to:

$$\underline{w} = \text{ManPref}[m, \text{Next}[m]]$$

$O(1)$

3_ Determine woman w's status

keep an array called Current[1..n]

Current[w] is Null if w is not engaged

set to m if she is engaged to m

O(1)

4_ Determine which man is preferred by w.

WomanPref$_i$ =

3	8	4	22	1	-	-	-	-
1	2	3						

WomanRanking$_i$ =

		1	3			
		3				

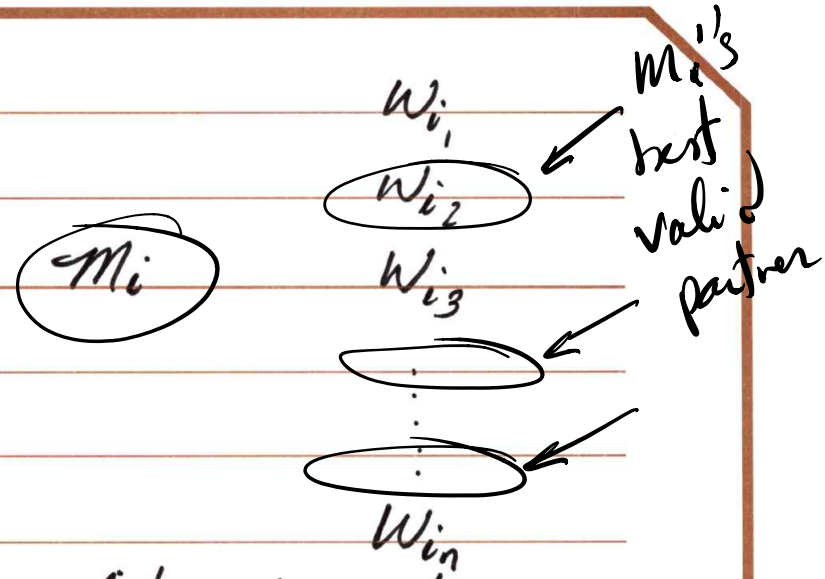
Preparation before entering GS iterations

Create a Ranking array where Ranking [w,m] contains the rank of man m based on w's preference

takes $O(n^2)$

Preparation		G-S iterations
$O(n^2)$	+	$O(n^2)$

overall performance $O(n^2)$



Def. Woman $\underline{w}$ is a valid partner of a man $\underline{m}$ if there is a stable matching that contains the pair (m, w)

Def. m's best valid partner is a valid partner who is the highest in m's pref. list.

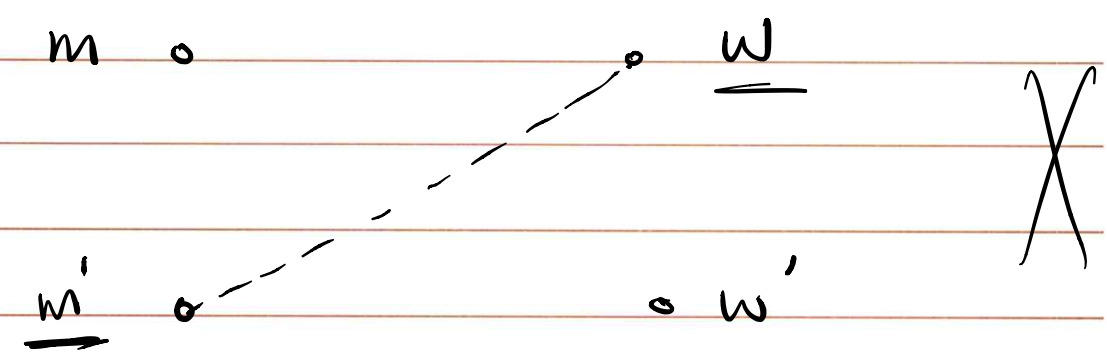
Claim: Every execution of the G-S algorithm (When men propose) results in the same stable matching regardless of the order in which men propose.

Plan: to prove this, we will show that when men propose, they always end up with their best valid partner.

Proof by contradiction:

Say $\underline{m}$ is the first man rejected by a valid partner w.

S'



Claim: When men propose, women end up with their worst valid partners

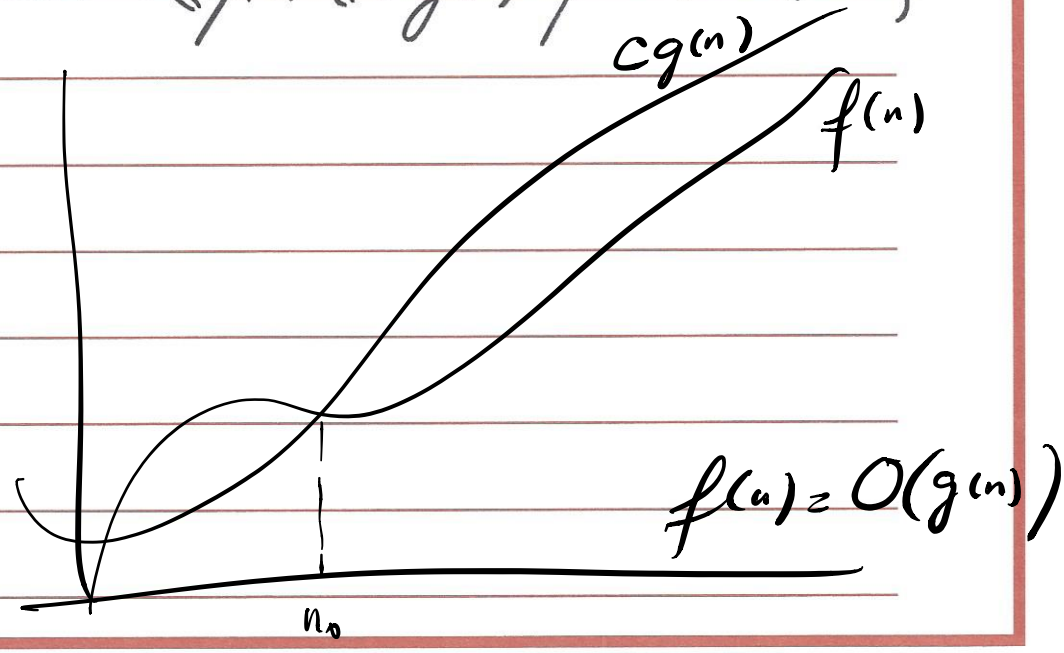
Proof: By contradiction . . . .

Suppose we end up with a matching S where for a pair (m,w) in S, m is <u>not</u> w's worst valid partner.

So there must be another matching S' where w is paired with a man m' whom she likes less.

Review of The Asymptotic Notation

Formally, $O(g(n)) = \{ f(n) \mid$ there exist positive constants c and n_0 such that $0 \leq f(n) \leq c\,g(n)$ for all $n \geq n_0 \}$

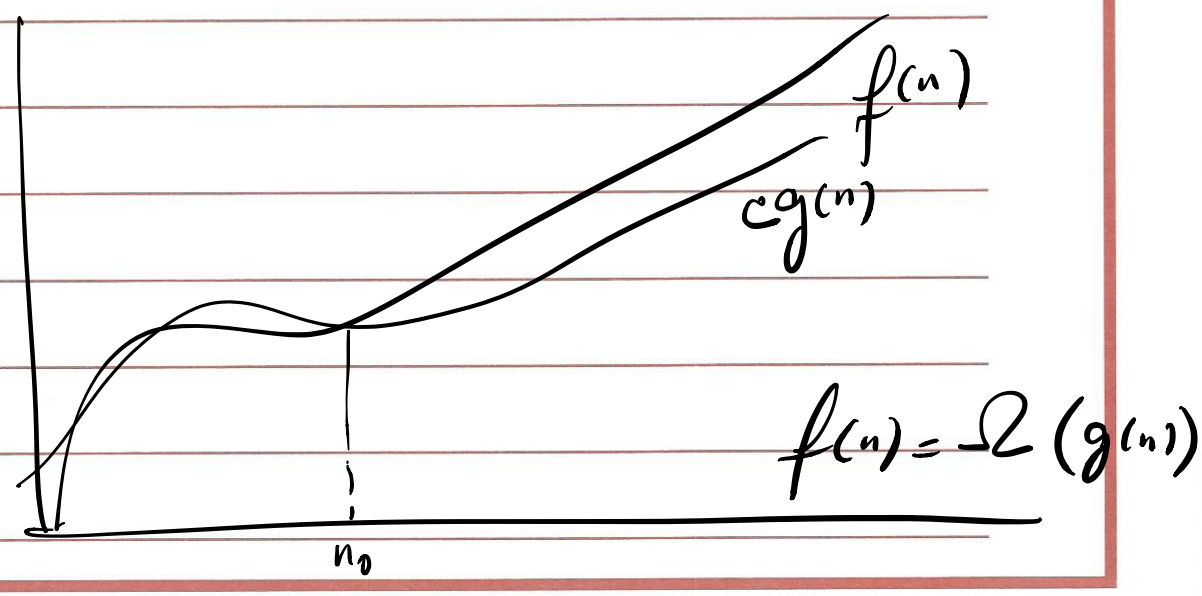


T Any quadratic function is $O(n^2)$

T Any linear " " $O(n^2)$

F Any cubic " " $O(n^2)$

$\Omega(g(n)) = \{ f(n) \mid$ there exist positive constants c and n_0 such that $0 \leq c\,g(n) \leq f(n)$ for $\underline{n \geq n_0} \}$

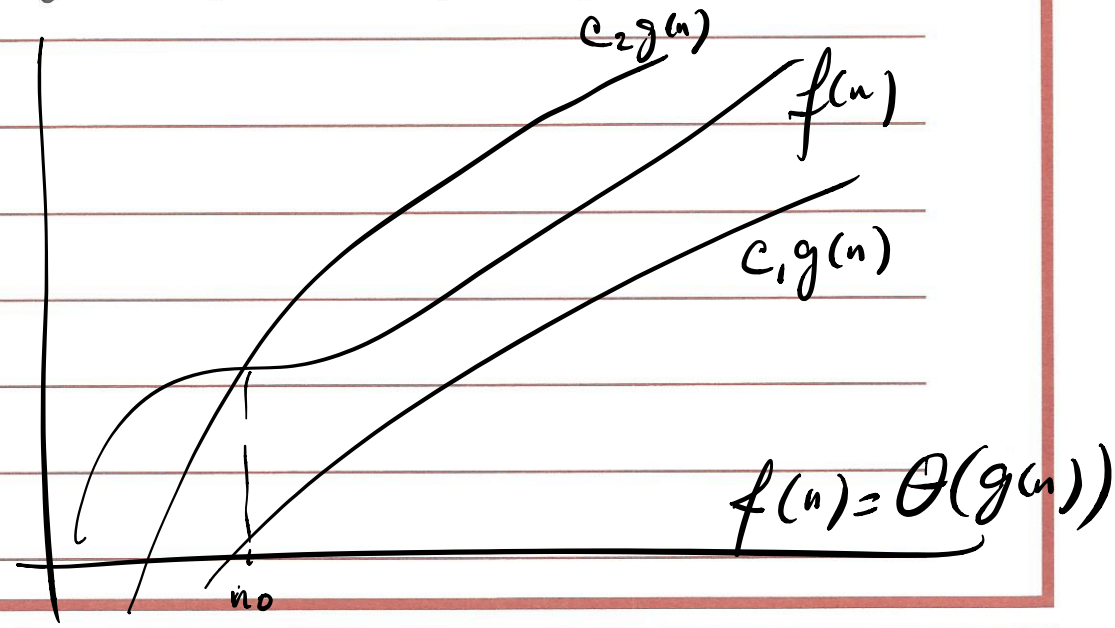


T Any quadratic function is $\Omega(n^2)$

F Any linear " " $\Omega(n^2)$

T Any cubic " " $\Omega(n^2)$

$\Theta(g(n)) = \{ f(n) \mid$ there exist positive constants c_1, c_2, and n_0 such that $0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n)$ for all $n \geq n_0 \}$



T Any quadratic function is $\Theta(n^2)$

F Any linear " " $\Theta(n^2)$

F Any cubic " " $\Theta(n^2)$

	Worst Case	Best Case
Linear Search	$O(n)$, $\Theta(n)$	$O(1)$, $\Theta(1)$
Binary Search	$O(\lg n)$, $\Theta(\lg n)$	$O(1)$, $\Theta(1)$
Insertion Sort	$O(n^2)$, $\Theta(n^2)$	$O(n)$, $\Theta(n)$
Merge Sort	$O(n \lg n)$, $\Theta(n \lg n)$	$O(n \lg n)$, $\Theta(n \lg n)$

Worst case complexity

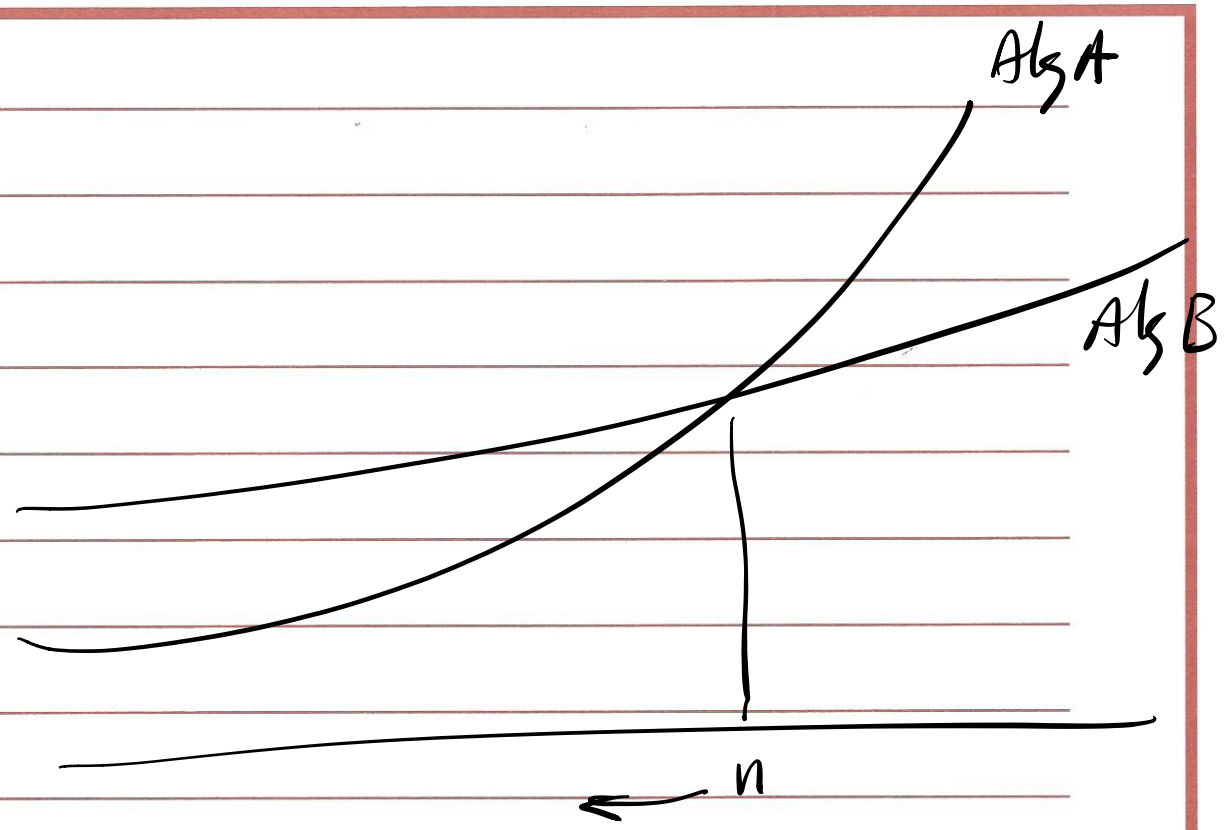
Algorithm A: $\Theta(4^n n^3 \lg n)$

Algorithm B: $\Theta(3^n n^8 (\lg n)^2)$

1- Exponential Component — fastest growing

2- polynomial "

3- logarithmic " — slowest growing

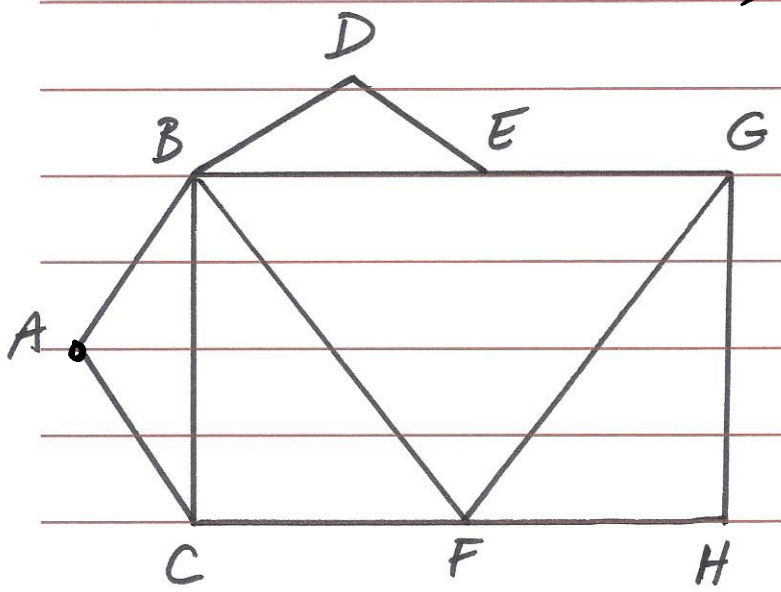


Review of BFS & DFS

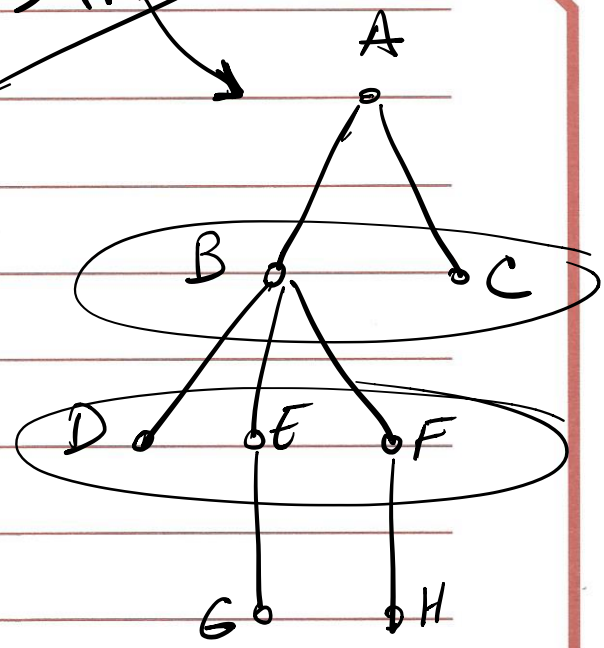
Q: What are we searching for?

- Find out if there is a path from A to B.
- Find all nodes that can be reached from A.

BFS



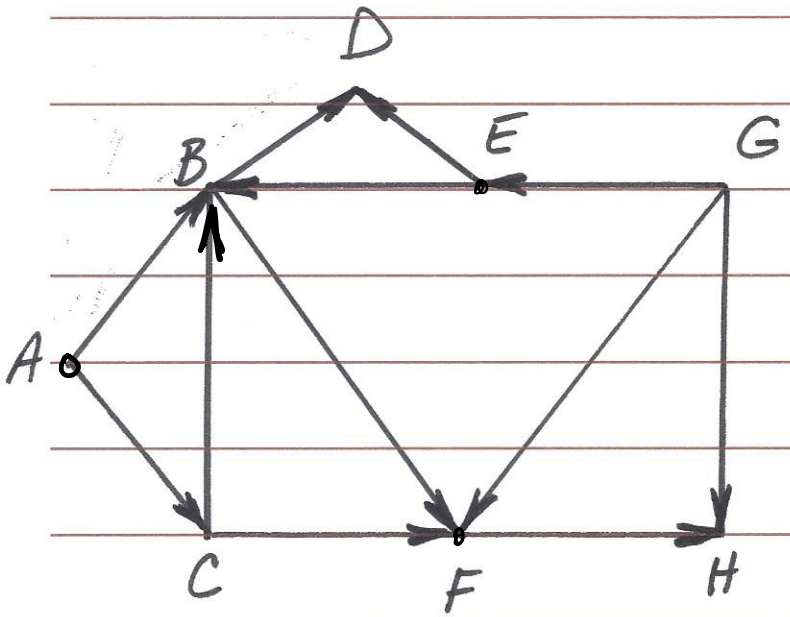
BFS tree!



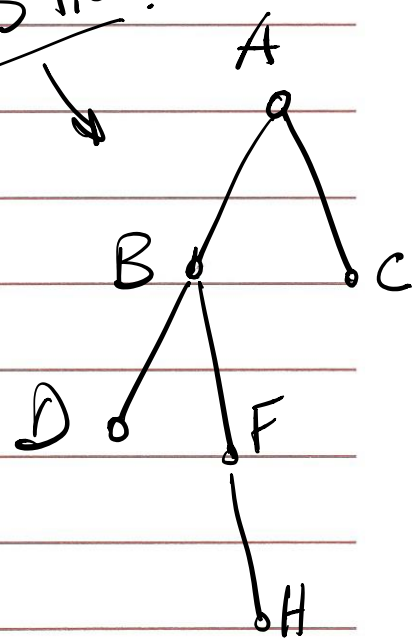
Say G has n nodes and m edges

BFS takes $O(m+n)$

DFS

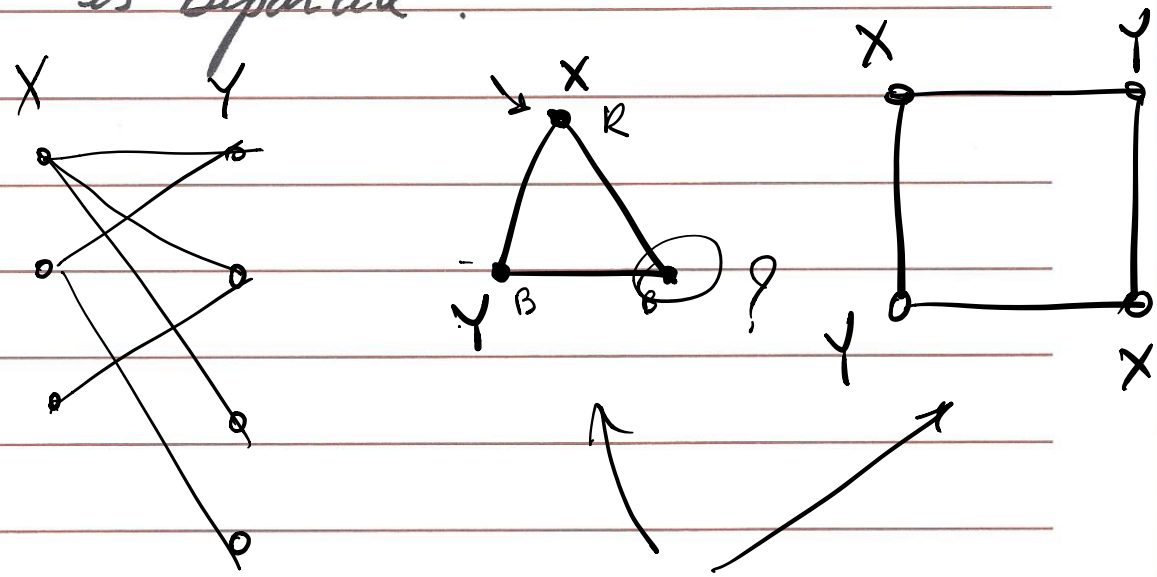


DFS tree!

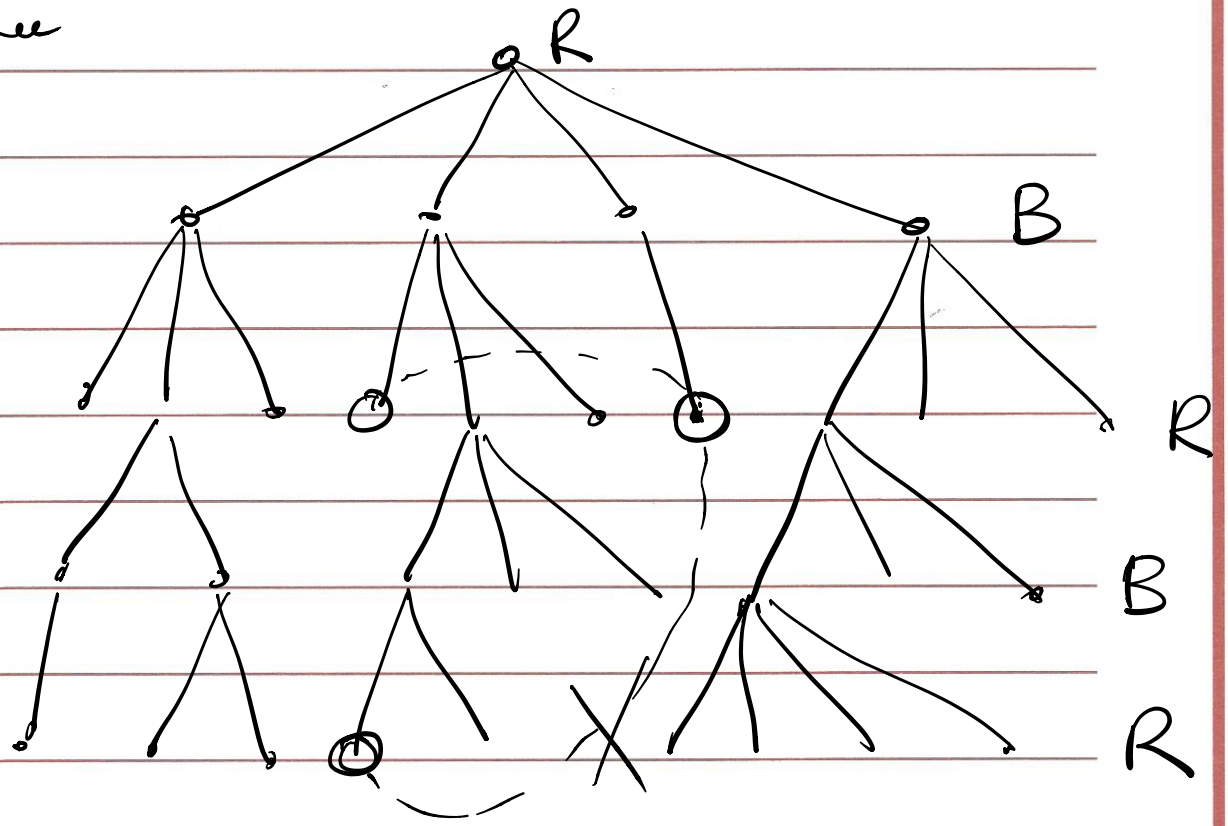


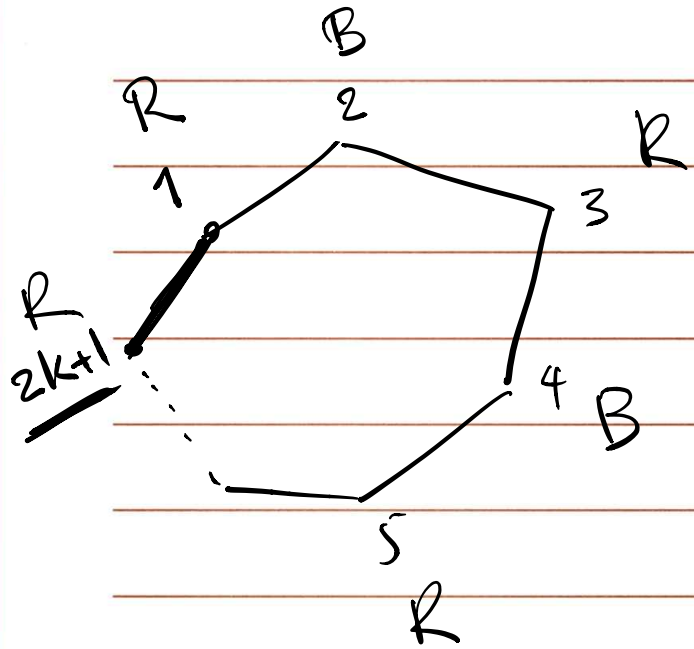
runs in $O(m+n)$

Q: How do you determine if a graph is bipartite?

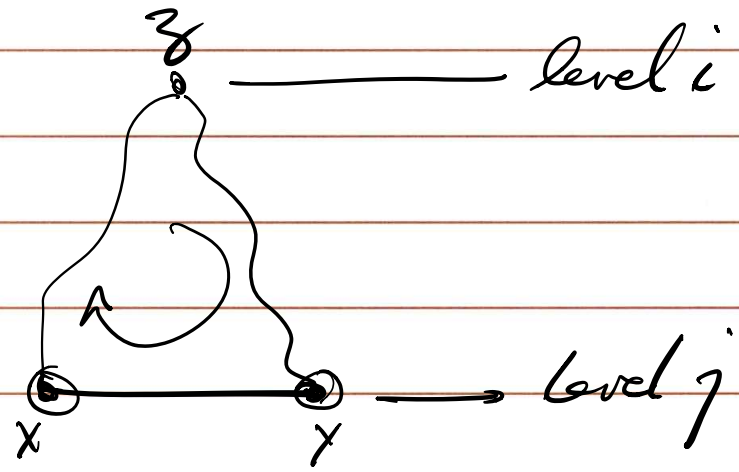


BFS tree





$\Rightarrow$ A graph containing an odd cycle cannot be bipartite.



length of this cycle is $\underline{2*(j-i)+1}$

Solution:

Run BFS starting from any node, say $\underline{S}$. Label each node $\underline{\text{Red}}$ or $\underline{\text{Blue}}$ depending on whether they appear at an odd or even level on the BFS tree.

Then, go through all edges and examine the labels at the two ends of the edge. If all edges have a Red end and a Blue end, then the graph is bipartite.

Otherwise, the graph is not bipartite.

BFS

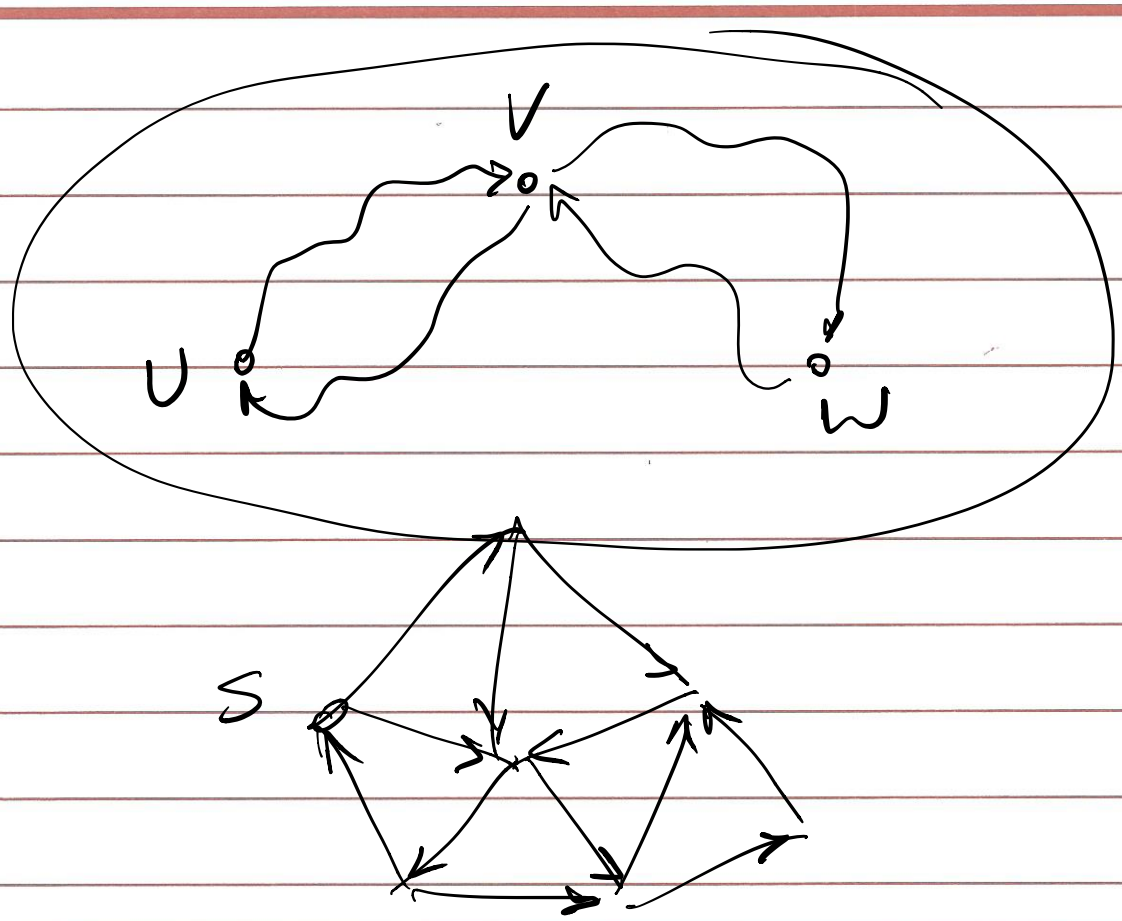
this takes $O(m+n) + O(m)$

$= O(m+n)$

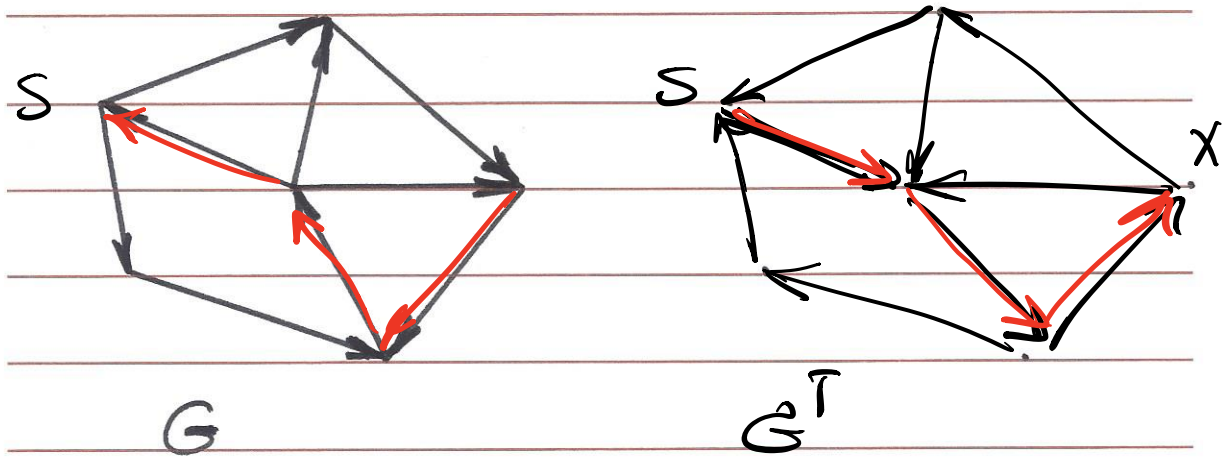
Def. A directed graph is strongly connected if there is a path from any point to any other point in the graph.

Q: How do you know if a given directed graph is strongly connected?

Brute force: $O(n \cdot (m+n)) = O(nm+n^2)$



Transpose of a directed graph



Mutually Reachable Nodes

Solution:

1. Use BFS or DFS to find all nodes reachable from s (an arbitrary node) in G. If some nodes are not reachable from s, stop. The graph is not strongly connected.

Otherwise, continue with step 2.

$O(m+n)$

2. Create G^T (Transpose of G) $O(m)$
3. Use BFS or DFS to find all nodes reachable from s in G^T. If some nodes are not reachable from s, then the graph is not strongly connected. $O(m+n)$

Otherwise, the graph is strongly connected.

overall takes $O(m+n)$