Revisiting BFS:

BFS(s):

Queue: FIFO

Discovered [s] = True

For all $v \in V \setminus \{s\}$:

  Discovered [v] = False.

$L[0] = \{s\}$ // list } $L \leftarrow$ empty queue

$i \leftarrow 0$ ✗

$T \leftarrow \phi$ // Empty tree.

While $L[i]$ is not empty: While L is not empty:

  $L[i+1] \leftarrow []$ // empty list ✗ $u \leftarrow L.$~~pop~~(u)

  For each $u \in L[i]$: ✗

    For each edge $(u,v) \in E$ incident on u: ✓

      If Discovered [v] == false :

        Discovered [v] $\leftarrow$ True

        $T \leftarrow T \cup \{(u,v)\}$

        $L[i+1]$. append (v) ✗ L. append (v).

  $i \leftarrow i+1$. ✗

---

n - layers ⇒ ≤ n iterations of while. ✓

$\sum_{v \in V} |N(v)| = 2m$.

O(m+n) running time.

Revisiting DFS:

Stack $S \leftarrow \perp$. // empty stack

DFS(s):

S.push(s).

While S is not empty:

$u \leftarrow$ S.pop()

If Explored[u] == False :

Explored[u] $\leftarrow$ True

For each $(u,v) \in E$:

S.push(v).

$R \leftarrow \{\}$

DFS(u):

Explored[u] $\leftarrow$ True

$R \leftarrow R \cup \{u\}$

For each $(u,v) \in E$:

If Explored[v] == False:

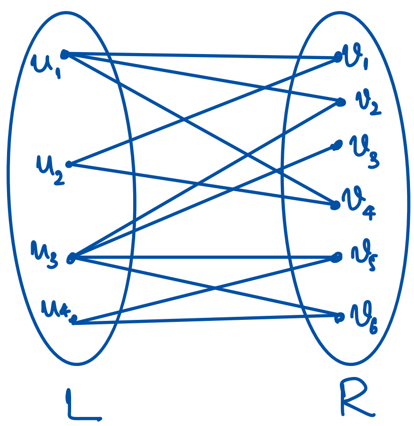
DFS(v) .

DFS(s)

Stack: { Last In First Out }

Applications: Testing bipartiteness.

Bipartite graphs: (L, R, E)



Lemma: A graph is bipartite if and only if it contains no odd cycles.

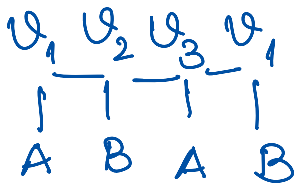
Pf: $(\Rightarrow)$ A graph is bipartite $\Rightarrow$ no odd cycles.

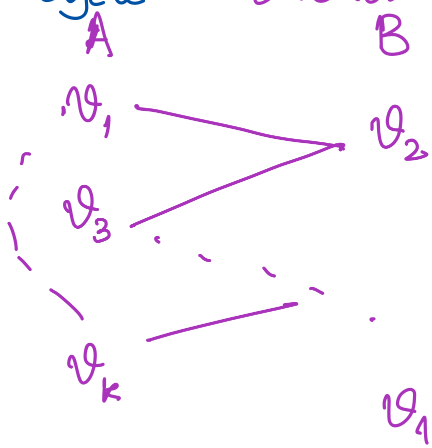
Proof by contradiction.

Assume that G is bipartite and it contains odd cycles.

$v_1, v_2, \ldots, v_k, v_1$ be a cycle.

$\uparrow$ k is odd





Case-1: $v_1 \in B$

or

Case-2: $(v_k, v_1) \in E(G)$

$v_k, v_1 \in A$.

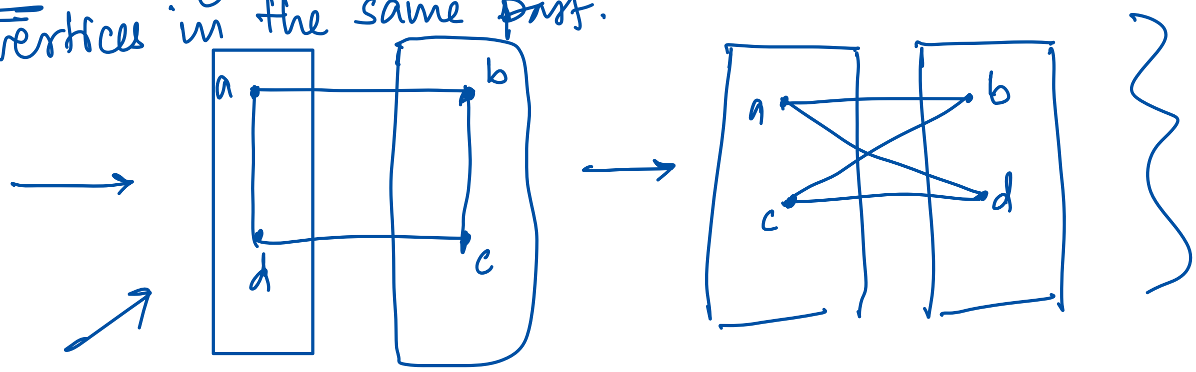
$(\Leftarrow)$ No odd cycles

$\Downarrow$

Graph is bipartite.

$\rightarrow$ The only cycles are even cycles.

and $\exists$ edges between pairs of vertices in the same part.



Claim: If there are only even cycles then there are no edges between pairs of vertices in the same part.

Claim: