Breadth First Search (contd.)

BFS(s):

```
Discovered[s] = True
For all v ∈ V\{s}:
    Discovered[v] = False

L[0] = {s}      // List implementation
   ↖ Mark s blue.
i ← 0
T ← ∅           // Empty tree.
While L[i] is not empty:
    L[i+1] ← [ ]     // Empty list
    For each u ∈ L[i]:
        For each edge (u,v) incident on u:
            If Discovered[v] == False:
                Discovered[v] ← True
                T ← T ∪ {(u,v)}     ↖ mark v Red
                                      if i+1 is odd
                                      else Blue.
                L[i+1].append(v)
    i ← i+1
Return (L,T)
```

Check if v is disc. and "colour is the same".

$$\sum_i \sum_{u \in L[i]} N_u \le \sum_{u \in V} N_u = \sum_{u \in V} \deg(u) = 2m$$

Question: Which steps change if we have a queue implementation?

Applications of BFS:

1. Shortest distances in an unweighted graph.
2. $s \sim t$ connectivity (undirected)
3. Testing bipartiteness

   → What is a bipartite graph?
4. Connectivity in directed graphs.

Testing Bipartiteness.

$G = (L, R, E)$

$E \subseteq L \times R$

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

<ins>Proof:</ins> A graph is bipartite $\Rightarrow$ It contains no odd cycles.

(L, R, E)

For the sake of contradiction, assume there is an odd cycle in the bipartite graph.

$$\underset{\in L}{v_1} - v_2 - \cdots v_k - v_1 \quad \Big| \quad k \text{ is odd.}$$

W.L.O.G assume that $v_1 \in L. \Rightarrow v_2 \in R$

Following the series of implications, $\{v_{2p} \in R$ and $v_{2p+1} \in L\}$

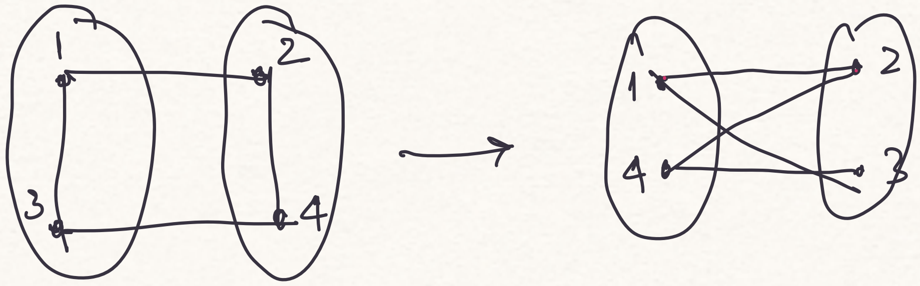
$v_k \in L$ and $v_1 \in L$ and $(v_k, v_1) \in E \Rightarrow G$ is not bipartite.

This contradicts the given fact that G is bipartite.

(V,E)

If G has no odd cycles then G is bipartite

(L,R,E)



BFS tree for a tree starting from node 's'$\in V$ is the rooted tree with s as root.

Algorithm for bipartiteness:

__Input__: $G=(V,E)$

__Output__: If G is bipartite, output (L,R,E)
else, output "No".

- Run BFS and obtain layers $L_0, L_1, \dots, L_{\underline{d}}$ (d = even)
- Let $L = L_0 \cup L_2 \cup \cdots \cup L_d$ (even layers)

  and $R = L_1 \cup L_3 \cup \cdots \cup L_{d-1}$ (odd layers).

- If $\exists$ an edge between a pair of vertices in L or in R, then output "No".

  Else, output (L,R,E)

$\rightarrow \quad (u,v)$ $\qquad u \in L_i \leftarrow$ Blue

$v \in L_j \quad j > i+1$

Blue

Properties of BFS:

If $(u,v) \in E$ Then $u, v \in$ adjacent layers or same layer.

Connectedness in Directed graphs

"Strongly connected"

↳ If for every pair of vertices in G, say u and v, there is a directed path from u to v and a dir. path from v to u.

For an undir. graph, G is connected if $\exists$ a path between all pairs of vertices in G.

Question: Is the given directed graph $G = (V, E)$ strongly connected?

G and G_{rev} → Flip the orientation/direction of each of the edges.

For a pair of vertices: $u, \underline{v} \in V$

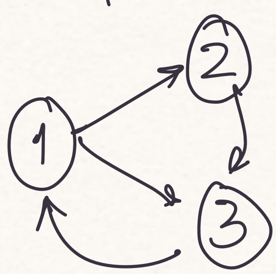
Run BFS from u

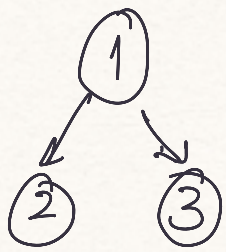
Run BFS from v

From each vertex running BFS tells us what we can discover from that vertex.

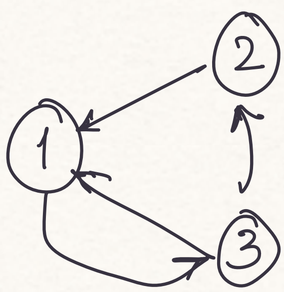
Run BFS in G_{rev}, this tells us which vertices can discover the source.

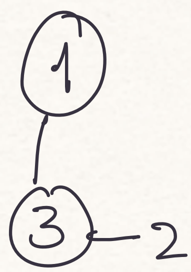
G





G_{rev}





What if s discovers every other vertex and s' doesn't (while s' can discover s).