Deepanshu 2019CS50427

Page :
Date :

Homework 19

Ans =>

```
Algorithm connectedComponents (Graph G)
    for each vertex u ∈ V(G)
        colour [v] ← white                  } Init all
        componentNumber [v] ← 0             } arrays.

    count ← 1
    i ← 0
    while (i<n)
        if (componentNumber[vertices[i]] ≠ 0)
            i++
        else
            BFS ( G, vertices[i] , componentNumber, count)
            count ++
    return componentNumber
```

```
Algorithm BFS ( G, s, componentNumber, count)
    colour [s] ← gray
    componentNumber [s] ← count
    Q ← [s]
    while Q ≠ ∅ do
        u ← head [Q]
        for each v ∈ adj[u] do
            if (colour [v] ← white) then
                colour [v] ← gray
                componentNumber [v] ← count
                Enqueue (Q, v)
        Dequeue (Q)
        colour[u] ← black.
```

Deepanshu 2019CS50427

Page :

Date :

Runtime Analysis of the Algorithm

- for doing BFS, every vertex is visited once and ~~we are~~ it is $O(|V|+|E|)$. Now since we are doing BFS in connected component $|E| \geq |V|-1$

  $O(|V|+|E|) = O(|E|)$.

- Also, we are traversing the ~~connectedcom~~ componentNumber array once which has a size of $|V|$.

  So, total running time of algorithm is $O(|V|+|E|)$.

Ans-2 ⇒ If G has an odd cycle, it cannot be bipartite

Proof by contradiction

Assume that there exist an odd cycle in a bipartite graph.
Call the vertices of that odd cycle as $v_1, v_2 \text{———} v_{2k+1}$.

Now since graph is bipartite, there exist 2 sets (call them A & B) such that every edge in graph is from vertex in set A to vertex in set B.

Without loss of generality, assume $v_1 \in A$. This implies that $v_2 \in B$ (from definition of bipartite).
This implies that $v_3 \in A$ which implies $v_4 \in B$.
Observe that odd index vertex belongs to A. Thus, $v_{2k+1} \in A$
Now $v_1 \in A \Rightarrow v_{2k+1} \in A$
But v_1 & v_{2k+1} are adjacent vertices. And belongs to same set. which contradicts the definition of bipartite. Hence, our assumption is wrong and "If G has an odd cycle, it cannot be bipartite".

Hence Proved.

Deepanshu 2019CS50427

Page :
Date :

If all cycles in a graph are of even length, then graph is bipartite

Proof by construction

We have to come up with 2 disjoint sets and each vertex should be present in exactly one of the set satisfying the bipartite condition of graph.

It is enough to prove for connected graph since it will be applied to every component.

Start with 2 empty sets, A & B

Pick any vertex. call it v_1. Put v_1 in A.

- Pick all the vertices at odd distance from v_1 and put them in B. (distance here means shortest distance)
- Pick all the vertices at even distance from v_1 and put in A.
- By doing this, all vertices of connected graph will be in exactly one of the sets.
- All the edges will be from vertex in A to vertex in B because if there is an edge with the vertices of set A (or B) it would lead to odd cycle in graph (by argument in part i) which is not possible.
- For graph having more than 1 component, repeat the process separately for all component and combine all the vertices in A_i's and B_i's (A_i's & B_i's are sets for different components).
- Thus A & B will be required set satisfying all the conditions of bipartite graphs.

Hence, if all cycles in a graph are of even length then graph is bipartite.

Hence Proved.

Deepanshu 2019CS50427

Algorithm checkBipartite (G, s)

```
for each vertex u ∈ V[G] - {s}          } Init all
    colour [u] ← white                    } vertices
    d[u] ← ∞
    π[u] ← NIL

colour[s] ← gray                         } Init BFS with source s.
d[s] ← 0
π[s] ← NIL
Q ← {s}

while Q ≠ φ do
    u ← head[Q]
    for each v ∈ adj[u] do
        if colour[v] = white then
            colour[v] ← gray
            d[v] ← d[u] + 1
            π[v] = u
            Enqueue(Q, v)
    for each v ∈ adj[u] do
        if colour[v] == gray and d[u] = d[v]
            return false     [ u & v are in same level
                               and are directly connected
                               by an edge ]
    Dequeue[Q]
    colour[u] ← black

return true
```

Runtime.

In worst case, we will have to execute whole of BFS (if graph is bipartite). Hence, time complexity is

$O(|V| + |E|)$.

Page :
Date :

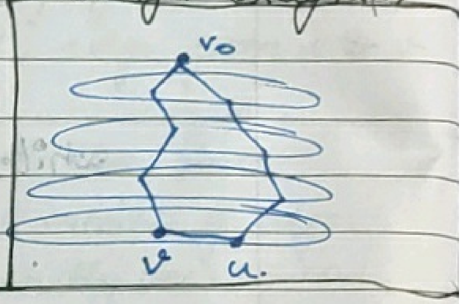
Deepanshu 2019CS50427

we claim that graph is not bipartite if two vertices in same level are directly connected.

Proof by implication.

Let v and u be at same level in BFS Tree.
Let v_0 be the nearest common ancestor. Note that they will always have a common ancester because they are ~~no~~ vertices of tree (BFS Tree).

Rough Diagram



Let v_0 be k levels above v and u.

Now since v and u are already connected by an edge, a cycle of length $2k+1$ is formed.

And since a cycle of odd length is found, graph is not bipartite.

If ~~no such~~ this is not the case, then graph formed is bipartite.

Proof by construction
we have to come up with 2 sets (call them A and B). such that $A \cap B = \phi$, $A+B=V$ and there is no edge between vertices in A (same for B). Define A and B as follows -

$A = \{v : d[v]$ from s is odd$\}$.
$B = \{v : d[v]$ from s is even$\}$.
Since we came up with 2 such sets, graph is bipartite.