Index

Introduction

Definitions

Graphic Sequence: Given a degree sequence S = <d1, d2, …, dn> is a graphic sequence, if and only if every realisation Graph G from the sequence S is a "simple graph", (i.e. no loops and no multiple edges)

Connectedness: A graph is said to be connected if for every pair of nodes in the graph, there exist a path.

Tree: If the graph is connected and it has n – 1 edges then it implies that it is a tree, and these are necessary and sufficient conditions

Cycle: If there exists a pair of nodes u, v ∈ G; such that u and v are connected by more than one path in the graph, then there is a cycle in the graph

Graph Property: If P is a graph property, and S = <d1, d2, …, dn> is a graphic sequence, then S is said to be potentially-P, if at least one of the realizations of S has the property P; it is said to be forcibly-P, if every realization of S has the property P.

Lemmas

Handshaking Lemma: Given a graphic sequence S = <d1, d2, …, dn>, then total number of edges in any realisation G from S will be equal to half of the summation of the degrees in S, hence 2*|E| = ∑di

Euler's Proof: Let's take a set I which has elements in the form of Incident Pair (V, e), where "e" is an edge and vertex "V" is one of its endpoints.

- Each "Vi" will be repeated "di" times in set I
- Therefore, size of the set I will be equal to the sum of the degrees in S … (1)
- However, each edge in the graph belongs to exactly two incident pairs, one for each of its endpoints
- Therefore, size of the set I will be equal to 2*|*E*|. … (2)
- Comparing both the results we get 2*|E| = ∑di

Cycle Lemma: Given that the graph G is connected, If G does not contain any cycle then Number of edges in the graph must be less or equal to number of nodes – 1
i.e. |E| ≤ |V| - 1
Proof by Contradiction: Let's assume that |E|>|V-1|, and the graph does not have any cycle

- We are given a connected graph, so we can create a Tree by connecting initial |V-1| edges
- By the definition of Tree: for each pair u, v in graph they will be connected by a unique path
- Now, we still have some edges left
- Now let's say we add the next edge to between nodes u, v
- Since u, v are now connected by more than one path, it creates the cycle by the definition
- Hence there must be a cycle, Contradiction!

Connected Component Lemma: A graph with n vertices and m edges has at least $n - m$ connected components

Proof: For $m = 0$, A graph with n vertices and no edges has n connected components as each vertex itself is a connected component. So, the graph has at least $n - 0$ connected components hence the claim is true for $m = 0$

Induction Hypothesis: Let's say $k \geq 0$, every graph with n vertices and k edges has at least $n - k$ connected components.

Induction Step: Using the inductive hypothesis we need to prove that G, with n vertices and $k+1$ edges has at least $n-(k+1) = n-k-1$ connected components.

- Now lets say that we got G' by removing some edge from G
- Now G has n nodes and k edges, hence G' has at least $n - k$ components
- Now we can add the edge to G and again get G'
- 1. If by adding the edge we connect two components then, number of components $= (n - k) - 1$, True according the assumption
- 2. If we are adding the edge inside an already connected component then, number of components $= (n - k)$
- In both the we have at least $n - k - 1$ components
- Hence by induction we can say that our lemma is correct

Potentially-Connected

Algorithm

- If any of the element in S is less than 1 return False
- If sum of all elements from degree sequence is greater or equal to 2*(n-1) return True, otherwise return False

```
potentially_Connected(S, n)

   If for each d in S
     If d<1
        Return False
   If SUM(S) is greater than or equal to 2*(n - 1)
         Return True
   Return False
```

Proof of Correctness

n = Number of Nodes

m = Number of Edges

Gi = Graph "i" realised by graphic sequence S

S = {d1, d2, …, dn}

Necessary Conditions:

1. $\forall i, d_i \geqslant 1$ if $n \geqslant 2$

   Proof: If there is a node a node with degree = 0, then it will be a singular node, never be a part of connected component, if number of nodes are $\geqslant 2$
   - Also, if number of nodes are equal to one then that node can have di = 0, since that node itself is the whole graph

2. Number of edges in G must be greater or equal to n – 1

   Proof: Let's say we found a connected graph with number edges less than n – 1
   - Applying Connected Component Lemma,
   - Number of nodes: n
   - Number of edges: m < n-1 implies – m > - (n - 1) … (1)
   - According to Lemma, total number of connected components in this graph would be at least n – m
   - Let's take total number of components = C then,
   - $C \geqslant n - m$
   - C > n – (n - 1) … using Eq. (1)
   - C > 1
   - $C \geqslant 2$
   - This makes the graph disconnected, Contradiction!

- To check that Edge $\geqslant n - 1$, simply sum of all the degree in graphic sequence and check this condition $\sum di \geqslant 2*(n - 1)$. (by Handshaking Lemma)

Sufficiency of the above two conditions: Prove that above two conditions are enough to build at least One connected graph

Trivial Case: If there is only one node in the graph, it will always be a connected graph

Proof: We are given a graph G: {V1, V2, V3, …, Vk}, where n: total number of nodes and m: total number of edges, k: total number of connected components in a graph

- Here, Vi is the ith connected component of the graph, $k > 1$
- Note. For $k = 1$, we have only one connected component which means graph is connected

Condition 1: $\forall i, di \geqslant 1$ using this we can say that every node is connected to at least one other node, since no node has degree equal to zero and the sequence is graphic

- Hence, we can infer that for every connected component Vi, Vi at least has one edge in the graph.
- I.e. Every connected component has at least one edge

Condition 2: $m \geqslant n - 1$ ..... (p)

- Using this condition lets connect all the disconnected components in the graph

Lemma 1: There is at least one Connected component Vi in the G, such that it has a cycle

Proof: Let's assume that there is no component with a cycle, then using Cycle Lemma, we can argue that every component has number of edges less than its nodes – 1.

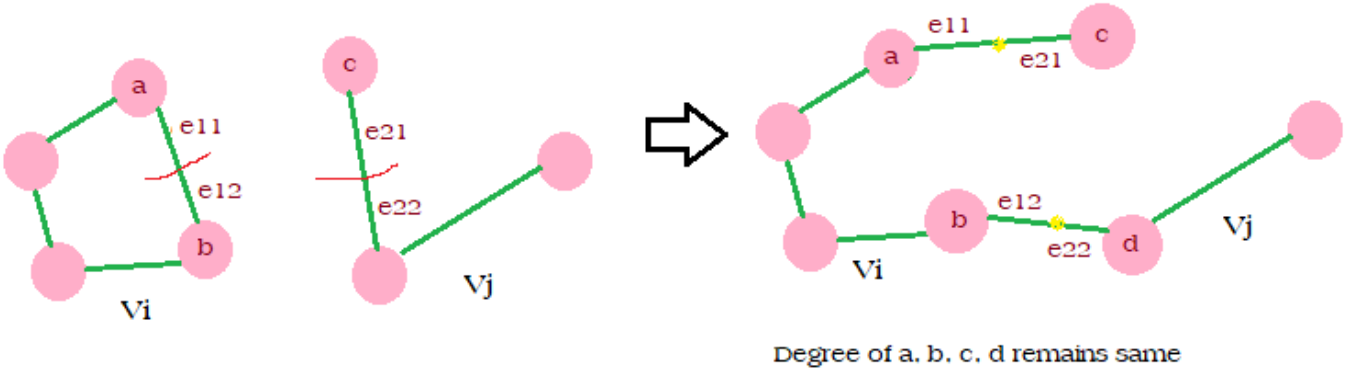
- In other words, if {n1, n2, n3, …, nk} are the number of nodes and {m1, m2, m3, …, mk} are the number of edges in each connected component respectively then,
- $m1 + m2 + m3 + \ldots + mk \leq n1 - 1 + n2 - 1 + n3 - 1 + \ldots + nk - 1$
- But, $m1 + m2 + m3 + \ldots + mk = m$ and, $n1 + n2 + n3 + \ldots + nk = n$
- Hence, $m \leq n - k$, (since -1 is repeated k times) ….. (q)
- Since $k > 1$, from $m \leq n - k$ maximum value of m can be $n - 2$, but we are given that $m >= n - 1$, hence Contradiction!
- Hence Lemma 1 is true

Lemma 2: If There is a Connected component Vi in the G, such that it has a cycle then, we can always connect Vi with any other component Vj in G, without changing the graphic sequence of the Graph.

Proof: We already proved using Condition 1 that Vj will have at least one edge in it.

- Let's take an edge "e1" from the cycle of the Vi, and cut it into two half pieces "e11" and "e12", we can call these "half edges" for the time being
- Let's take any edge "e2" from the Vj, and cut it into two half pieces "e21" and "e22"
- Now connect "e11" with "e21", and "e12" with "e22"
- Since the all the vertex are just inter-changing the edges from one node to another node, and also no other vertices are in picture, graphic sequence is not changed, because total number of degrees of each vertex remain the same (see figure)

- Now let's argue about the connected-ness
  1. Since we are choosing edge from the cycle of Vi cutting that edge into half, will not disconnect the Component Vi
  2. Let's say by cutting the edge of Vj, Vj gets divided into two separate components, we are merging edge of them both into the Vi, hence Vj is completely connected to Vi now, and Number of components in the graph are decreased by 1



Lemma 3: We can apply the lemma 1 and lemma 2 on any number of components and reach to a single connected component, in other words using Lemma 1 and Lemma 2

We can always build at least one connected component.

- Let's say that we connected any two of the connected components using Lemma 1 and Lemma 2
- After updating, the change in the state of graph is as below
- G': {V1, V2, V3, …, Vk-1}, where n: total number of nodes and m: total number of edges, k: total number of connected components in a graph, $k > 1$
- Note. If k = 1, we have only one connected component which means graph is connected
- We already proved in the lemma 2 that the graphic sequence and total number of edges does not change during merging of two components
- Hence the both the Necessity Conditions $\forall i,\ di \geqslant 1$ and $m \geqslant n - 1$ still remain true
- Hence we can again argue the Lemma 1 and Lemma 2 on G' and further reduce one component
- Finally, we will reach the state that K>1 condition does not hold, which means there is only one connected component the graph itself.

Time Complexity

Summing over all the degrees = O(n)

Overall complexity = O(n)

Forcibly-Connected

Algorithm

- We divide S into two subsets S1 and S2 where $S1 \cap S2 = \phi$ and $S1 \cup S2 = S$ and $S1 \neq \phi$ and $S2 \neq \phi$
- If S1 and S2 are also graphic then we return False, otherwise return True
- Iterate over all such possible combinations

```
Is_graphic(S):

   Sort_Desc(S)

   If S(0) == 0
      Return True

   K ← S(0)
   i ← 1
   While K > 0 and i < size(S)
      If S(i) == 0
         Break
      S(i) ← S(i) – 1
      K ← K – 1
      i ← i + 1

   Remove first element from S

   If K > 0
      Return False
   Else
      Return Is_graphic(S)


forcibly_Connected(S, n)

   For each S1, S2 in All_Possible_Combinations(S)
      If is_graphic(S1) and is_graphic(S2) both are True
          Return False
   Return True
```

Proof of Correctness

n = Number of Nodes

m = Number of Edges

G = Graph

Is Graphic: We are simply implementing the **Havel - Hakimi Theorem** to check whether given sequence is graphic or not, return True if the sequence is graphic else False

Proof for Forcibly Connected:

- For a given graphic sequence S, lets divide S into to mutually exclusive and complete sets. i.e. $S1 \cap S2 = \phi$ and $S1 \cup S2 = S$

Lemma A: S is not a forcibly-connected graphic sequence If and only If there is exist some pair of sets S1 and S2 such that both are graphic sequences, where $S1 \cap S2 = \phi$ and $S1 \cup S2 = S$ and $S1 \neq \phi$ and $S2 \neq \phi$

LHS to RHS: If S is not a forcibly-connected graphic sequence then there is exist some pair of sets S1 and S2 such that both are graphic sequences, where $S1 \cap S2 = \phi$ and $S1 \cup S2 = S$

Proof: Since S is not a forcibly-connected

- We can at least two connected components let's say G1 and G2 such that
- $G = G1 \cup G2$ and $G1 \cap G2 = \phi$
- Now for both of this individual Graphs G1 and G2, we can create individual graphic sequences for them, simple by making a set which contains degree of each node in G1 and G2
- Also, combining both the graphs completely we get the original graph back
- Hence both $S1 \cap S2 = \phi$ and $S1 \cup S2 = S$ conditions are satisfied
- Hence we found S1 and S2 given that graph is not forcibly-disconnected

RHS to LHS: If there is exist some pair of sets S1 and S2 such that both are graphic sequences, where $S1 \cap S2 = \phi$ and $S1 \cup S2 = S$, then S is not a forcibly-connected graphic sequence.

Proof: If we find set S1 and S2 such that S1 is a graphic sequence and S2 is a graphic sequence

- Let's say G_1 is a realisation of S1,
- Let's say G_2 is a realisation of S2
- Since S1 and S2 are entirely independent graphic sequences hence $S1 \cap S2 = \phi$
- $G_1 \cap G_2 = \phi$, In other words, G_1 and G_2 are not connected by an edge
- Now we can create $G = G_1 \cup G_2$ resulting into $S1 \cup S2 = S$
- Then G will be the realization of S, since number of nodes and degrees will be same for the G
- Hence G is a disconnected graph
- Hence S is not forcibly connected

Lemma B: By showing the results for only two components we are not losing the any kind of generality

Proof: Let's say there are total "k" number of connected components then,

- We can argue that we choose "x" out of "k" into one subset and "k-x" out of "k" into another subset
- Since it is a graphic sequence union will also be a graphic sequence
- Hence by showing the result for two set of graphic sequences we are covering all the cases.

Hence, using Lemma A and Lemma B we can say that S is not a forcibly-connected graphic sequence If and only If there is exist some pair of sets S1 and S2 such that both are graphic sequences, where $S1 \cap S2 = \phi$ and $S1 \cup S2 = S$ and $S1 \neq \phi$ and $S2 \neq \phi$; without the loss of generality.

Time Complexity

All combinations of making two sets = O(2^n)

Time Complexity of Is_Graphic() = O(n^2)

Overall complexity = O((n^2) * (2^n))

Potentially-Tree

Algorithm

If sum of all elements from degree sequence is equal to 2*(n-1) and also if degree sequence is potentially connected then and only then return TRUE

```
potentially_Tree(S, n)

    If potentially_Connected(S, n) is True
        If SUM(S) is equal to 2*(n - 1)
            Return True
    Return False
```

Proof of Correctness

n = Number of Nodes

m = Number of Edges

{G1, G2, G3, ..., Gs} = SG = Set of Graphs which can be realised by S

- Using the definition of Tree we just need to check that ∃ Gi ∈ SG which is connected and has total number of edges equal to n – 1

1. Checking ∃ Gi ∈ SG which is connected
   Proof: Since we already have an algorithm potentially-Connected which gives us decision whether given a graphic sequence can have at least one connected tree or not, we will be directly using that to check the current requirement. (see the proof of potentially connected graph)

2. Number of edges is equal to n – 1
   Proof: Using the handshaking lemma we can show that the number of edges for given degree sequence will be equal to SUM of all elements divided by 2.
   - Now, m = SUM(S)/2 and m = n – 1
   - Hence, SUM(S) = 2 * (n - 1)
   - This we are also checking in our algorithm

Time Complexity

Summing over all the degrees = O(n)

Time Complexity of potentially_Connected() = O(n)

Overall complexity = O(n)

Forcibly-Tree

Algorithm

If sum of all elements from degree sequence is equal to 2*(n-1) and also if degree sequence is potentially connected then return TRUE

```
forcibly_Tree(S, n)

    If forcibly_Connected(S, n) is True
        If SUM(S) is equal to 2*(n – 1)
            Return True
    Return False
```

Proof of Correctness

n = Number of Nodes

m = Number of Edges

{G1, G2, G3, …, Gs} = SG = Set of Graphs which can be realised by S

- Using the definition of Tree we need to check that ∀ Gi ∈ SG which is connected and has total number of edges equal to n – 1

1. ∀ Gi ∈ SG must be connected
   Proof: For Every G that can be realised by S, we need to prove that ∀ Gi ∈ SG
   - The above statement is equivalent to the definition of forcibly-Connected graph
   - Hence, this property can be checked using forcibly_Connected(S, n) algorithm which we have already introduced

2. Number of edges for ∀ Gi ∈ SG must be equal to n – 1
   Proof: Using the handshaking logic we can show that the number of edges for given degree sequence will be equal to SUM of all elements divided by 2.
   - Now, m = SUM(S)/2 and m = n – 1
   - Hence, SUM(S) = 2 * (n - 1)
   - Since for every Gi in SG, the SUM(S) will remain same hence number of edges will remain n – 1 for ∀ Gi ∈ SG

Time Complexity

Summing over all the degrees = O(n)

Time Complexity of forcibly_Connected() = O((n^2) * (2^n))

Overall complexity = O (n + ((n^2) * (2^n))