International Journal of Mathematical Archive-6(10), 2015, 39-42
Available online through www.ijma.info ISSN 2229 – 5046

GRAPH CLUSTERING: A LINEAR ALGORITHM, THE CLUSTERS OF THE GIVEN GRAPH HAVE BEEN FORMED BASED ON THE LIMIT WE SPECIFIED AT THE BEGINNING

ISHWAR BAIDARI*[1], AJITH HANAGWADIMATH[2]

[1,2] Department of Computer Science, Karnatak University, Dharwad, India.

(Received On: 19-09-15; Revised & Accepted On: 19-10-15)

ABSTRACT

A graph G is a pair of sets G (V, E), V is the set of vertices and the number of vertices n=|n| is the order of the graph. The set E contain the edge of the graph. Here we present a linear graph clustering algorithm the clusters of the given graph have been formed based on the limit we specified at the beginning. Limit specifies the number of adjacent vertices we include in one cluster.

Keywords: *Graph, Cluster-Nodes, Limit, Cluster Edge, Cluster Connecting Edge.*

1. INTRODUCTION

The concept of clustering can be considered the most important problem. It deals with finding a structure in collection of unlabeled data. A cluster is therefore a collection of objects which are similar between them and are dissimilar to the objects belongings to other clusters. The process of identifying in terms of grouping the data elements is called clustering also called data classification [1]

Graphs are formed by a set of vertices and set of edges denoted by G (V, E) Edges connect between pairs of vertices. Graph clustering is the task of grouping the vertices of the graph into clusters taking into consideration the edge structure of the graph in such a way that there should be many edges within each cluster and relatively far between the clusters. Graph clustering in the sense of grouping the vertices of a given input graph into clusters. Here we presented a linear graph clustering algorithm the clusters of the given graph have been formed based on the limit we specified at the beginning. Limit specifies the number of adjacent vertices we include in one cluster.

The construction and correction of the algorithm is based on following attributes

- vertex u ∈ G.V the attribute u.color holds the color of the vertex u.
- vertex u ∈ G.V the attribute u.cluster holds the number of the clusters
- edge (u, v) ∈ G.E the attribute (u,v).color holds the color of the edge (u, v).
- edge (u, v) ∈ G.E the SetLabel function assigns a label depend on following condition such as,
    - If the edge connects two vertices present in the same cluster then the label will be set as ‘Cluster_Edge’.
    - If the edge connects two vertices present in different clusters then the label will be set as ‘Cluster_Connecting_Edge’.
- In line 20 of the algorithm we are creating set data structure according to the value of the variable ‘Cluster Count’. **For ex:** If the value of the variable ‘Cluster_Count’ is 1 then the statement ‘*Set Cluster’* will create a set with name **‘Cluster_1”**

2. ALGORITHM

```
Cluster(G)
1    for each vertex u ∈ G.V
2    {
3        u.color = White
4        u.cluster = Nil
5    }
```

Corresponding Author: Ishwar Baidari[1]*
[1,2] *Department of Computer Science, Karnatak University, Dharwad, India.*

```
6   for each edge (u, v) ∈ G.E
7   {
8       (u, v).color = White
9       Set Label ( (u, v), Undetermined)
10  }
11  Cluster_Count = 0
12  Cluster_Limit = 3
13  for each vertex u ∈ G.V
14  {
15          if (u.color == white)
16          {
17                  Cluster_Count = Cluster_Count+1
18                  u.color = Black
19                  u.cluster = Cluster_Count
20                   ClusterSet = ClusterSet ⋃ {u}
21                  Limit = 1
22                  for each vertex v ∈ G.Adj[u]
23                  {
24                          if ( v.color == white )
25                          {
26                             v.color = Black
27                             v.cluster = Cluster_Count
28                             (u,v).color = Green
29                             SetLabel ((u,v), Cluster_Edge)
30                              ClusterSet=ClusterSet ⋃{v}
31                             Limit = Limit + 1
32                          }
33                          else
34                          {
35                             (u, v).color = Red
36                             SetLabel ( (u,v), Cluster_Connecting_Edge)
37                          }
38                          if (Limit == Cluster_Limit )
39                             break
40                  }
41          }
42  }
43  for each edge (u,v) ∈ G.E
44  {
45          if ( GetLabel(u,v)==Undetermined )
46          {
47                  if ( u.cluster==v.cluster)
48                          SetLabel ((u,v), Cluster_Edge)
49                  else
50                  {
51                          (u, v).color = Red
52                          SetLabel ( (u, v), Cluster_Connecting_Edge)
53                  }
54  }
```

3. WORKING OF THE ALGORITHM

- The for loop in lines 1-5 initializes all the vertices of the given graph. The u. color attribute is set to 'White' for all the vertices in line 3. And the u.cluster attribute is set to 'Nil' for all the vertices in line 4.
- The for loop in lines 6-10 initializes all the edges of the given graph. The (u, v).color attribute of all the edges is set to 'White' in line 8 and the label of all the edges is set to 'Undetermined' in line 9.
- In line 11 a variable 'Cluster Count' is declared and initialized with zero. This variable gives the total number of clusters formed.
- In line 12 a constant 'Cluster Limit' is declared and initialized with value 3. This value gives the upper limit on the size of a cluster.
- The for loop in lines 13-42 scans every vertex u ∈ G.V of the given graph and forms clusters accordingly.

  - The condition in line 15 checks whether the color of vertex u==white, if so, then the variable **'Cluster_Count'** is incremented by 1 in line 17. The **u.color** attribute of vertex u is set with **'Black'** in line 18.The **'u.cluster'** attribute is assigned the value **Cluster_Count** in line 19 And the vertex 'u' is added to the set **'Cluster_Cluster_Count'** in line 20.
  - In line 21, a variable *'Limit'* is declared and initialized with 1.
  - The for loop in lines 22-40 scans the adjacency list of vertex 'u' and determines the cluster member vertices.
    The condition in line 24 checks whether the color of the vertex 'v' ∈ adjacency list of vertex 'u' is White, if so then, the color attribute of vertex 'v' is set to 'Black' in line 26. The **'v.cluster'** attribute is assigned the value **Cluster_Count** in line 27. The (u, v).color attribute for the edge (u, v) is set to Green in line 28. The Label 'Cluster_Edge' is set to the edge (u,v) in line 29. The vertex 'v' is added to the set 'Cluster Set' in line 30 and the value of the variable 'Limit' is incremented by 1 in line 31.
    If the condition in line 24 fails then the color attribute of edge (u,v) is set to 'Red' in line 35 and the label of the edge (u, v) is set to 'Cluster_Connecting_Edge' in line 36.
  - The condition in line 38 checks whether the value of the variable 'Limit' is Equal to 'Cluster_Limit'. If so the control comes out of for loop of lines 22-40(due to break) and continues with the next iteration of the for loop of lines 13-42.
- The for loop in lines 43-54 is used to determine the edges as either 'Cluster_Edges' or 'Cluster_Connecting_Edges'
  - The condition in line 45 checks whether the label of the edge (u, v) is 'Undetermined' , if so then the condition in line 47 checks whether both the vertices 'u' & 'v' belong to the same cluster, if so then, the label 'Cluster_Edge' is set to the edge (u,v) in line 48.
  - If the condition in line 47 fails then, the color attribute of the edge (u, v) is set to Red in line 51 and the Label 'Cluster_Connecting_Edge' is set to the edge (u,v) in line 53.

4. TIME COMPLEXITY

The total time taken by the algorithm is O (V+E).

Example: Consider the graph given below:

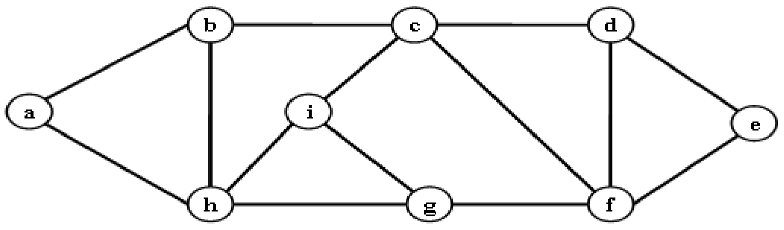

Figure-1.

The clusters formed for the above graph according to the algorithm are as shown below:

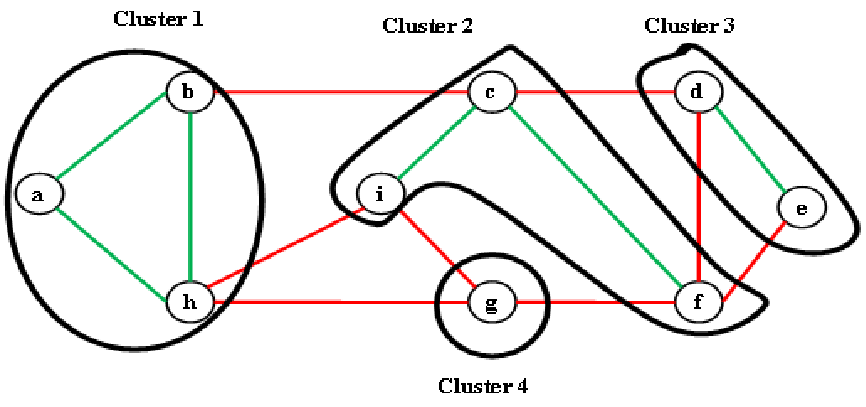

Figure-2.

Set Cluster_1 = {a, b, h}
Set Cluster_2 = {c, i, f}
Set Cluster_3 = {d, e}
Set Cluster_4 = {g}
Cluster_Count = 4

REFERENCES

1. Satu Elisa Schaeffer, *Graph clustering journal of Computer Science*, Review I (2007) 27-64.
2. Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein, 3rd Edition, *Introduction to Algorithms*, The MIT Press Cambridge, Massachusetts London, England.

Source of support: Nil, Conflict of interest: None Declared