Clustering By Density

Submission deadline: 9-1-2021

Clustering

- A mean to extract information by grouping items into cohesive groups.
- First step in understanding your data.

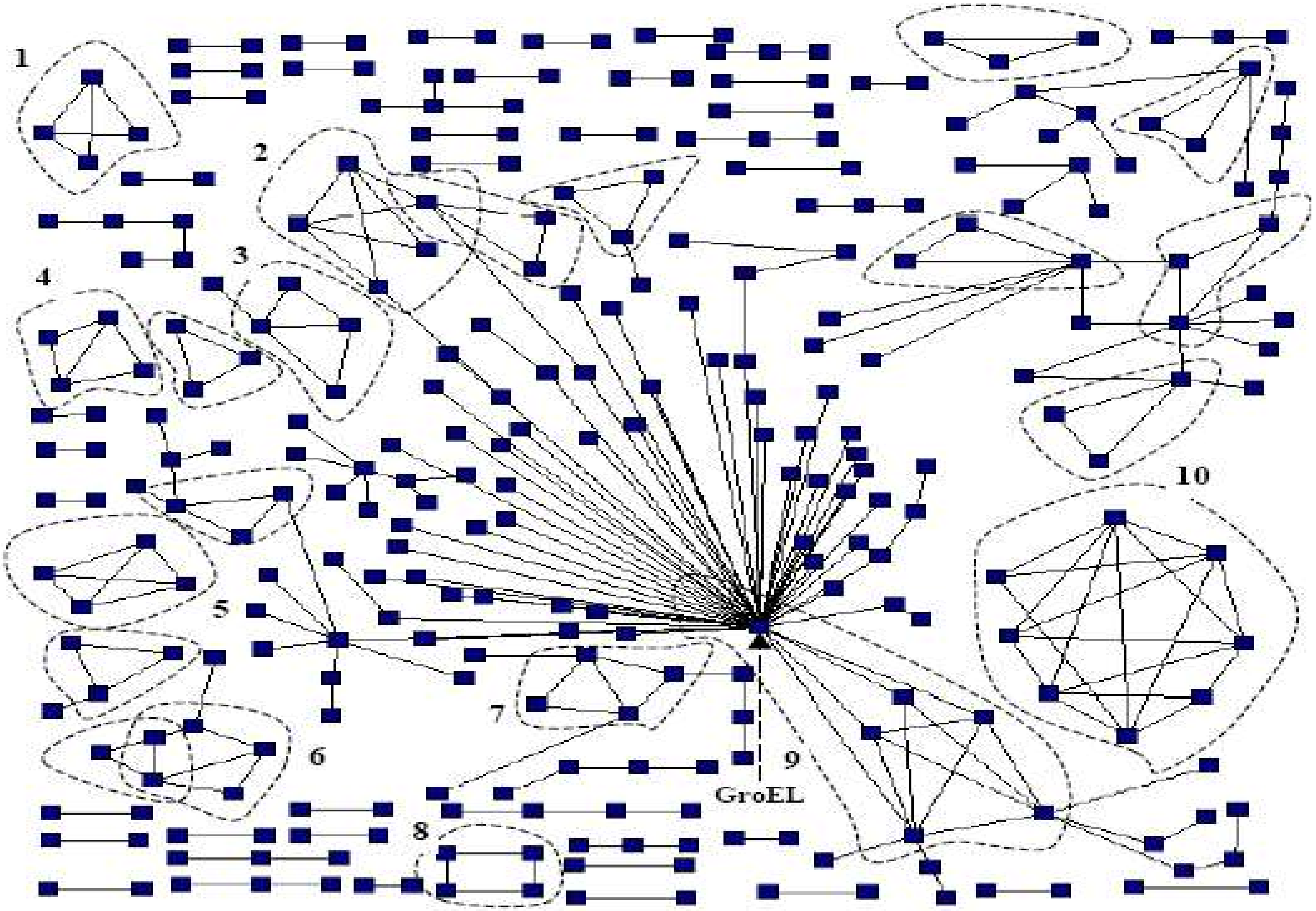
Clustering By Density

- Represent problem as a network / graph

- Each node in graph represent item to be grouped.

- An edge between two node represent that the two items are related to each other.

- Clustering is to find such sub-graphs (group of nodes) whose density is more or equal to the given value.



Protein-Protein interaction network and its major clusters (density > 0.6)

High Density Clusters

- A node that is part of a cluster should be connected to reasonable number of edges within cluster.
- Two nodes belonging to same cluster have more common neighbors than two nodes that are not.
- If a node is part of bigger cluster, its degree within cluster should be more than its degree when it is part of a smaller cluster.

Notations

- An undirected simple graph G = (*N, E)*
- *M* be the associated matrix of *G*
- |N| denotes no. of nodes
- |E| denotes no. of edges
- d denotes density

$$d = \frac{|E|}{|E|_{max}} = \frac{2 \times |E|}{|N| \times (|N| - 1)}$$

Algorithm

- **Input**
  - Associated Matrix of graph (M)
  - Threshold value for density (d’)
  - Threshold value for Cluster property of a node (cp’)

- **Algorithm**
  - Start with a single node as a cluster
  - Grow cluster by adding nodes from neighbors one by one

Algorithm

- Continue expanding cluster as long as following 2 conditions are satisfied
  - Density of cluster > = d’
  - Node is in periphery of cluster
- Remove Cluster from graph
- Apply same procedure to rest of nodes to find other clusters in graph.

- **Output**
  - Generates clusters whose density >= d’

Details

- An undirected simple graph G= (N, E)
- M be the associated matrix of G
- **Weight of an Edge:**

  The *weight of an edge* (u, v) є E is the number of the common neighbors of the nodes u and v .

  M^2 for $u \neq v$ represents the number of common neighbor of the nodes u and v .

Details

- **Weight of a Node:**
  - The *weight of a node* is the sum of the weights of the edges connected to the node.
  - The weight of every node is calculated and then the highest weight node is determined.
  - We start at the highest weight node as the   cluster and then grow it larger.

Details

- **Generating Neighbors:**

Neighbors of a cluster are the nodes connected to any node of cluster but not part of the cluster.

- **Adding Neighbors:**

To guide the cluster formation in a proper way, add neighbor nodes on priority basis.

The priority is determined based on two measures, (1) the sum of the weights of the edges between a neighbor and the cluster,

(2) the number of edges between a neighbor and the cluster.

Details

- **Sorting Neighbors:**

Neighboring nodes are sorted on any one of the two basis and node having large values of measures have highest priority.

- **Adding a Node to Cluster:**

Before adding a node to a cluster, check two things.

- – Make sure that addition of the node to the cluster does not cause the density *d* of the cluster to fall below the threshold density d'.
- – Second is to check whether the node is part of the cluster or part of the periphery.

Details

- **Part of Periphery**
  - To determine whether node is part of periphery we use *cluster property* **'cp'** of node.
  - If a node exist in periphery of cluster it should be connected to reasonable no. of edges within cluster.

Formally '*cp*' of a node w.r.t a cluster of density '*d*' and size $|N_c|$ is

$$\frac{|E_c|}{d \times (|N_c|)}$$

$|E_c|$ is no. of edges between node and the cluster.

Details

- **Adding a Node to Cluster:**

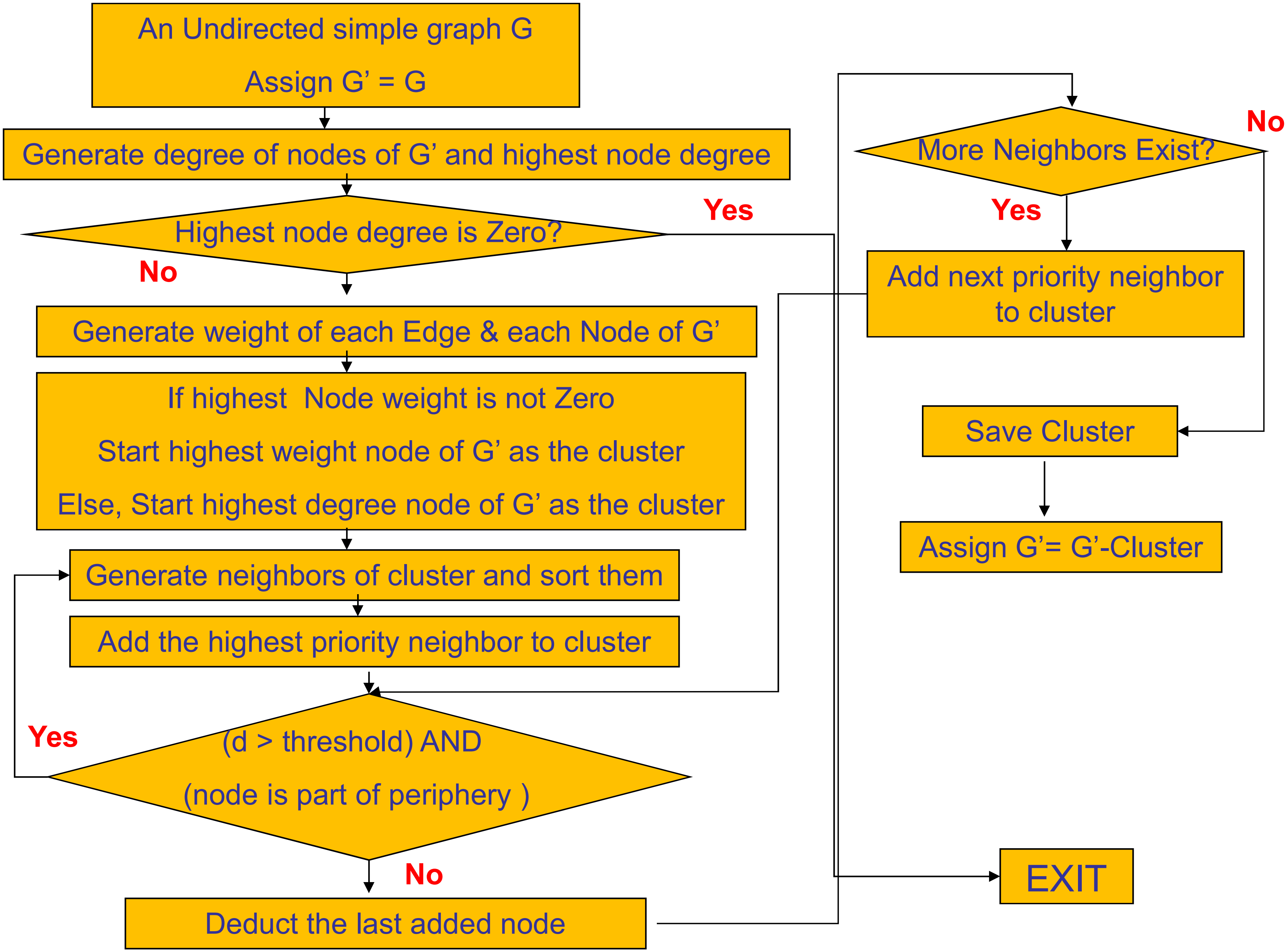
  Don’t add a node to cluster if

  – Its addition cause the density of resulting cluster fall below threshold d’.

    - where $0 \leq d' \leq 1$

  – cp value of node is less than a threshold value cp’.

    where $0 < cp' \leq 1$



An Undirected simple graph G

Assign G’ = G

Generate degree of nodes of G’ and highest node degree

Highest node degree is Zero?

Yes

No

Generate weight of each Edge & each Node of G’

If highest Node weight is not Zero

Start highest weight node of G’ as the cluster

Else, Start highest degree node of G’ as the cluster

Generate neighbors of cluster and sort them

Add the highest priority neighbor to cluster

(d > threshold) AND

(node is part of periphery)

Yes

No

Deduct the last added node

More Neighbors Exist?

No

Yes

Add next priority neighbor to cluster

Save Cluster

Assign G’= G’-Cluster

EXIT