VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

Name: ***Sancheet Kumar Baidya***

Register No: ***22MCB0029***

Subject: **Social Network Analytics *Lab***

Subject Code: **MCSE618L**

Faculty Name: ***Prof. DURGESH KUMAR***

Submission Date: ***28-MAY-2023***

IMPLEMENTATION OF COMMUNITY DETECTION ALGORITHMS ON A SOCIAL NETWORK

While humans are very good at detecting distinct or repetitive patterns among a few components, the nature of large interconnected networks makes it practically impossible to perform such basic tasks manually. Groups of densely connected nodes are easy to spot visually, but more sophisticated methods are needed to perform these tasks programmatically. Community detection algorithms are used to find such **groups of densely connected components** in various networks.

M. Girvan and M. E. J. Newman have proposed one of the most widely adopted community detection algorithms, the **Girvan-Newman algorithm**. According to them, groups of nodes in a network are tightly connected within communities and loosely connected between communities.

In this article, you will learn the basic principles behind community detection algorithms, their specific implementations, and how you can run them using Python and NetworkX.

Community Detection Example Applications

Because networks are an integral part of many real-world problems, community detection algorithms have found their way into various fields, ranging from social network analysis to public health initiatives.

- A known use case is the detection of terrorist groups in social networks by tracking their activities and interactions. Similarly, groups of malicious bots can be detected on online social platforms.
- Community detection can be used to study the dynamics of certain groups that are susceptible to epidemic diseases. Other types of diseases can be studied similarly to discover common links among patients.
- One of the most recent use cases, community evolution prediction, involves the prediction of changes in network structures.

Community Detection Techniques

There are two main types of community detection techniques, agglomerative and divisive.

Agglomerative methods generally start with a network that contains only nodes of the original graph. The edges are added one-by-one to the graph, while stronger edges are prioritized over weaker ones. The strength of an edge, or weight, is calculated differently depending on the specific algorithm implementation.

On the other hand, **divisive methods** rely on the process of removing edges from the original graph iteratively. Stronger edges are removed before weaker ones. At every step, the edge-weight calculation is repeated, since the weight of the remaining edges changes after an edge is removed. After a certain number of steps, we get clusters of densely connected nodes, a.k.a. communities.

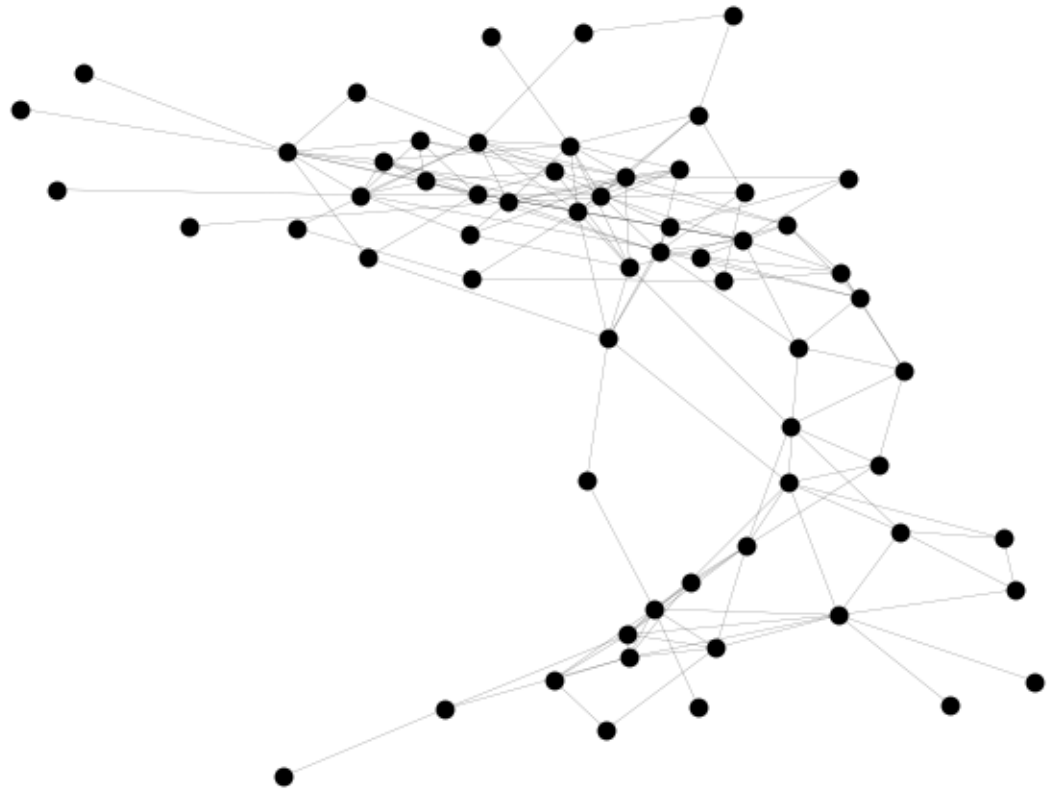
Community Detection in NetworkX

- **Girvan-Newman algorithm**: The Girvan-Newman algorithm detects communities by progressively removing edges from the original network.
- **Fluid Communities algorithm**: This algorithm is based on the simple idea of fluids interacting in an environment, expanding and pushing each other.
- **Label Propagation algorithm**: Label propagation is a semi-supervised machine learning algorithm that assigns labels to previously unlabeled data points.
- **Clique Percolation algorithm**: The algorithm finds k-clique communities in a graph using the percolation method.
- **Kernighan-Lin algorithm**: This algorithm partitions a network into two sets by iteratively swapping pairs of nodes to reduce the edge cut between the two sets.

Implementated Algorithms

I have implemented Givan- Newman, Fluid Communities and Clique Percolation Algortihm.

Given Dolphins Social Network can be represented as :



Clique Percolation Algorithm

The Clique Percolation Algorithm (CPA) is a community detection algorithm that identifies overlapping communities in networks based on the concept of k-cliques. It was proposed by Palla et al. in 2005 as an alternative approach to community detection.

The algorithm operates in the following steps:

1. Define k: The algorithm requires specifying the parameter k, which determines the size of the cliques that will be considered as building blocks of communities.
2. Find k-cliques: Identify all k-cliques (subgraphs with k nodes where each node is connected to every other node) in the network. This can be done using existing algorithms such as the Bron-Kerbosch algorithm.
3. Construct the clique graph: Create a clique graph, where each k-clique is represented as a node and there is an edge between two nodes if they share k-1 nodes in common. This clique graph represents potential communities.
4. Find connected components: Identify the connected components in the clique graph. Each connected component corresponds to a potential community.
5. Merge overlapping communities: If two or more communities share k-1 nodes in common, they are considered to overlap. Merge these overlapping communities to form larger communities.
6. Output the communities: The resulting communities are the final output of the Clique Percolation Algorithm.

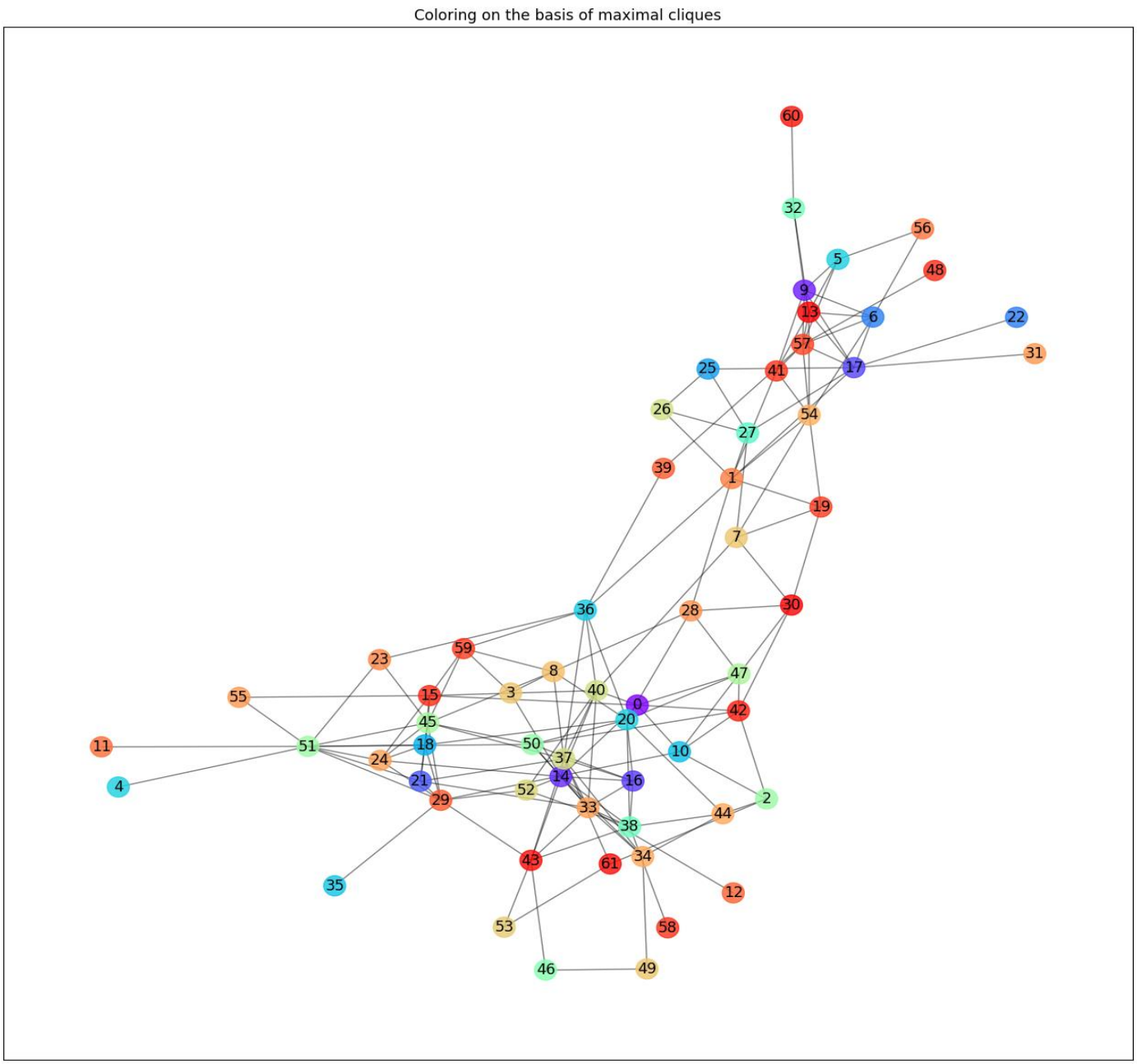
By using k-cliques as building blocks, the CPA captures overlapping community structures, as nodes can belong to multiple communities that share k-1 nodes. The choice of k determines the size and granularity of the detected communities. Smaller values of k lead to smaller and denser communities, while larger values of k result in larger communities with potentially weaker connections.

One advantage of the Clique Percolation Algorithm is its ability to detect communities with arbitrary shapes and sizes, as it does not assume any particular structure or connectivity pattern. It can be applied to a wide range of networks, including social networks, biological networks, and technological networks.

However, a limitation of the Clique Percolation Algorithm is that it can be computationally expensive, especially for large networks, as it involves enumerating and analyzing all k-cliques. Additionally, the algorithm does not provide an automatic way to determine the optimal value of k, which needs to be specified by the user.

Overall, the Clique Percolation Algorithm offers a valuable approach for detecting overlapping communities in networks and can complement other community detection methods.

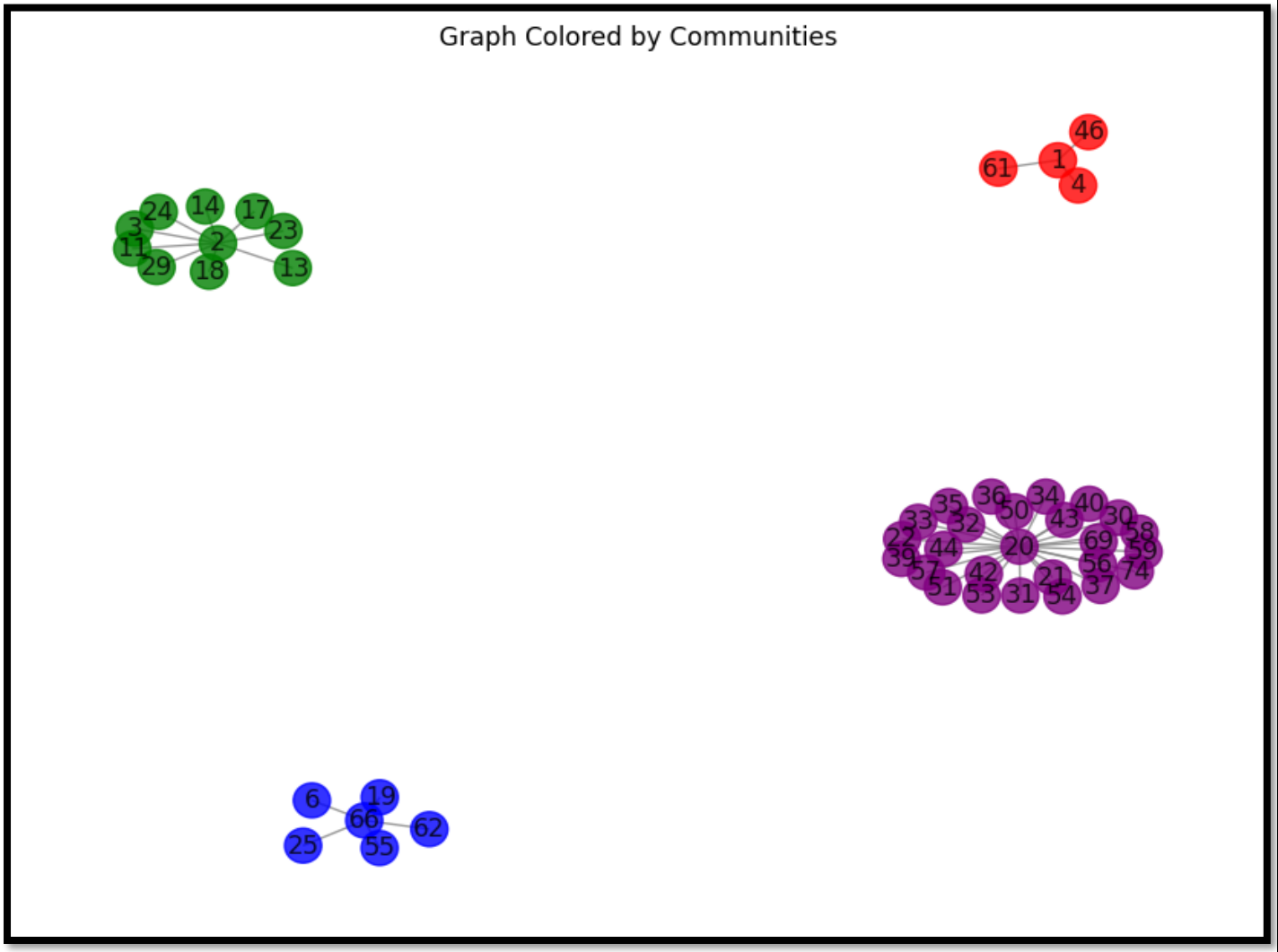
- **Identifying all the maximum cliques and representing them in a graph as**



- **Identifying the communities in the graph**

```
Community 1: {1, 4, 61, 46}
Community 2: {2, 3, 11, 13, 14, 17, 18, 23, 24, 29}
Community 3: {66, 6, 19, 55, 25, 62}
Community 4: {20, 21, 22, 30, 31, 32, 33, 34, 35, 36, 37, 39, 40, 42, 43, 44, 50, 51, 53, 54, 56, 57, 58, 59, 69, 74}
```

➢ **Coloring the nodes based on their communities**



Girvan-Newman Algorithm

The Girvan-Newman algorithm, also known as the edge betweenness algorithm, is a hierarchical community detection algorithm based on the concept of edge betweenness centrality. It was proposed by Mark Girvan and Mark Newman in 2002.

The algorithm aims to identify communities in a network by iteratively removing edges with the highest betweenness centrality. The betweenness centrality of an edge measures the number of shortest paths between pairs of nodes that pass through that edge. Edges with high betweenness centrality are considered to be important in connecting different parts of the network and are more likely to be "bridges" between communities.

Here are the main steps of the Girvan-Newman algorithm:

1. Calculate the betweenness centrality for all edges in the network.
2. Identify the edge(s) with the highest betweenness centrality. These edges are considered to be the most "central" in the network.
3. Remove the edge(s) with the highest betweenness centrality from the network.
4. Recalculate the betweenness centrality for the remaining edges.
5. Repeat steps 2-4 until no edges are left in the network or a stopping criterion is met.
6. The resulting disconnected components of the network after edge removal represent the detected communities.

The Girvan-Newman algorithm is based on the intuition that communities can be identified by the presence of edges with high betweenness centrality, which act as bridges connecting different parts of the network. By iteratively removing these bridges, the algorithm gradually breaks down the network into smaller components, which are potential communities.

One way to determine the optimal number of communities is by using a measure called modularity. Modularity quantifies the quality of the community structure by comparing the actual number of edges within communities to the expected number of edges in a random network. The algorithm can be run multiple times, each time calculating the modularity value for the resulting community structure, and selecting the partition with the highest modularity as the final result.

The Girvan-Newman algorithm provides a hierarchical view of communities since it identifies communities at different levels of granularity. By removing edges with the highest betweenness centrality in each iteration, it captures both large-scale and small-scale community structures.

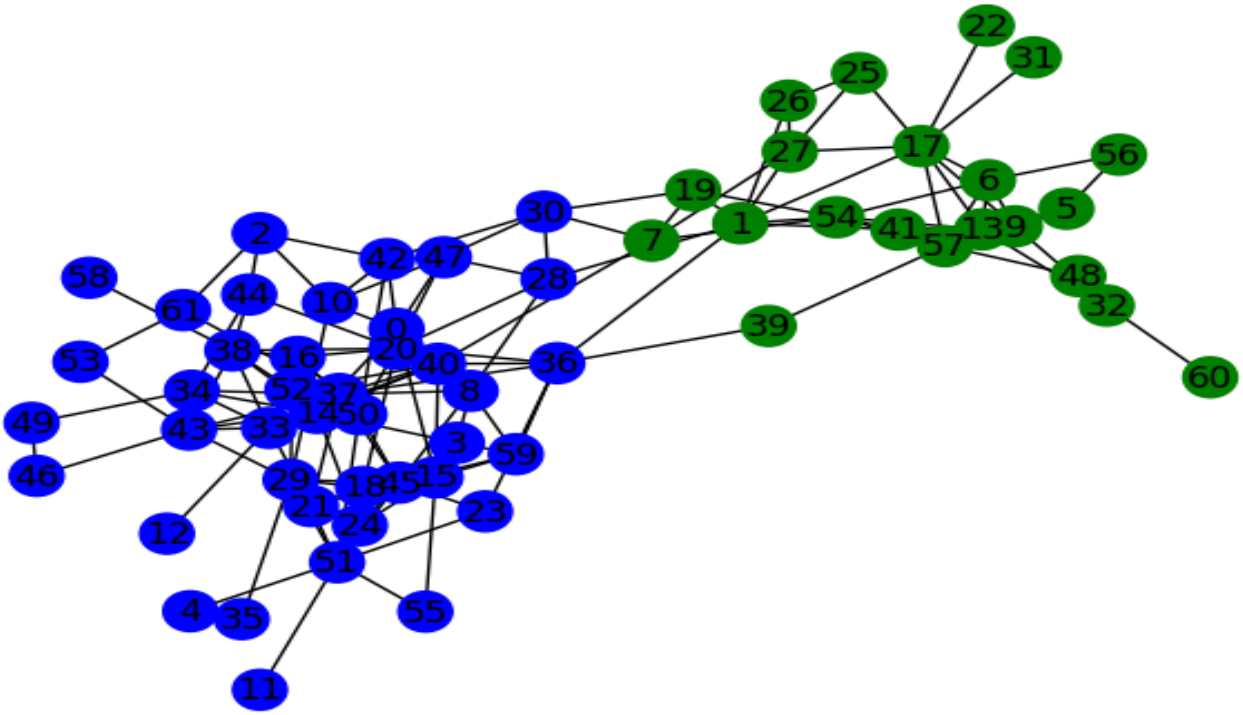
However, one limitation of the Girvan-Newman algorithm is its high computational complexity, especially for large networks, as it requires calculating betweenness centrality for all edges. Alternative methods, such as fast algorithms or heuristics, have been proposed to address this issue while still approximating the original algorithm's results.

- Forming communities based on the above algorithm

[0, 2, 3, 4, 8, 10, 11, 12, 14, 15, 16, 18, 20, 21, 23, 24, 28, 29, 30, 33, 34, 35, 36, 37, 38, 40, 42, 43, 44, 45, 46, 47, 49, 50, 51, 52, 53, 55, 58, 59, 61],

[1, 5, 6, 7, 9, 13, 17, 19, 22, 25, 26, 27, 31, 32, 39, 41, 48, 54, 56, 57, 60]

- Visulalizing those communities


22
31
25
26
27
17
56
19
6
30
1
54
41
5
2
7
139
57
42
47
48
58
28
44
10
32
61
0
39
53
38
16
20
40
36
60
34
52
37
8
49
14
50
43
33
3
59
46
29
18
45
15
21
24
23
12
51
4
35
55
11

Fluid Communities Algorithm

The Fluid Communities algorithm is a community detection algorithm that aims to capture overlapping communities in complex networks. It was proposed by Palla et al. in 2005 and is based on the concept of network flows.

The algorithm begins by assigning each node to its own initial community. It then iteratively updates the community assignments by considering both local and global information. The key idea behind the Fluid Communities algorithm is to simulate a flow of a fluid (representing information or influence) through the network and observe the dynamics of the flow to identify communities.

Here are the main steps of the Fluid Communities algorithm:

1. Initialization: Each node is initially assigned to its own community.
2. Flow simulation: A fluid is introduced into the network and allowed to flow along the edges. The flow is driven by local forces and global forces. Local forces encourage the fluid to stay within the node's community, while global forces attract the fluid towards nodes with similar characteristics.
3. Community update: After the flow simulation, the community memberships of nodes are updated based on the observed flow dynamics. Nodes are more likely to belong to communities through which the fluid has passed more frequently.
4. Iteration: Steps 2 and 3 are repeated iteratively until a stopping criterion is met. This could be a maximum number of iterations or when the community assignments stabilize.

The Fluid Communities algorithm allows nodes to belong to multiple communities simultaneously, capturing the overlapping nature of communities. The strength of a node's membership in a community is determined by the flow of the fluid through that community.

One advantage of the Fluid Communities algorithm is that it does not require the number of communities to be predetermined, as the algorithm can naturally adapt to the network's structure. Additionally, it has been shown to be effective in detecting overlapping communities in various types of networks, including social networks and biological networks.

It's worth noting that there may be variations and refinements of the Fluid Communities algorithm proposed by different researchers, building upon the original concept.