International Journal of Contemporary Research in Computer Science and Technology (IJCRCST) e-ISSN: 2395-5325
Volume 6, Issue 1 (August '2023)

PERFORMANCE EVALUATION OF COMMUNITY DETECTION ALGORITHMS IN SOCIAL NETWORKS ANALYSIS USING LOUVAIN LABEL PROPAGATION AND INFOMAP ALGORITHMS

D.Dhanalakshmi,
Research Scholar,
Department of Computer Science,
Periyar University,Salem,
Tamilnadu,India.

Dr.G.Rajendran,
Head and Assistant Professor,
Department of Computer Science,
Tiruvalluvar Government Arts College,
Rasipuram, Tamilnadu, India.

Abstract: Community detection is a growing field of interest in the area of Social Networks. Many community detection methods in recent years, there are many real networks that are closely formed according to a community network. Many research efforts to develop algorithms and methods that can efficiently find the hidden pattern in the community network. This paper describes various community discovery algorithms such as Non overlapping community detection algorithms, Traditional Algorithm. Finally, research shows a comparison of Louvain algorithm, Label Propagation algorithm, Infomap algorithm concerning these parameters: Modularity of communities, Computational time, Normalized Mutual information, Execution Time, Accuracy. The identified communities by all the community detection algorithms for these four datasets Zachary's Karate Club network, Amazon network, YouTube network, Football network data sets are described in this research.

Keywords: Community Detection, Social network, Label propagation algorithm, Louvain Algorithm, Infomap Algorithm, Normalized Mutual Information

I.INTRODUCTION

A network of socially connected people through different links/relations or connection, the links may be social sites, applications, physical interactions business platforms, educational platforms, research citations. The representation of the social network gives a complex view to determine any specific node or community. Therefore, some algorithms are used on the network population to detect the communities or other network related components. Social networks have been studied fairly extensively over the last couple of decades in the general context of analyzing interactions between people in order to deter- mine the important structural patterns in such interactions. The trends in recent years have been focused on online social networks enabled as an Internet application. Some examples of such networks are Amazon, YouTube. These social networking services have been rapidly growing in popularity for they are no longer constrained by the geographical limitations of a conventional social network connecting people through face-to-face contact, or personal friendships [1].

- The use of social networks like Facebook and YouTube during recent years shows that these networks are not only a means to spend time but also an evolution in human interactions. In other words, there is no doubt that online social networking websites have changed our ways of communication and caused us to spend plenty of time wandering over their web pages. Therefore, social network analysis might be of interest to different groups, including sociologists to analyze users' social behavior, computer engineers to design more efficient services and also security authorities to assess propagation channels of information and gossips,
- Decision makers in enterprises to use the underlying information in social interaction context to assist them for decision making in various contexts.
- One of the most important tasks when studying these networks is community detection. Community structure has a long history in social science, but it has become the focus especially in the fields of physics and computer science in recent years by Newman and Girvan's research.
- In addition, by the emergence and massive success of social media, a new environment has been created for the community. The determination of these communities is useful in the context of a variety of applications in social network analysis, including customer segmentation, recommendations, link inference, vertex labeling, and influence analysis.

The key point is the recognition of hidden structures in real social networks, which is necessary to analyze the organization of complex networks [2][3]. Complex systems are usually built in blocks that have specific roles and functions. Such blocks become a set of vertices with high density of internal links and at the same time low density of external links in the network representation. These blocks are

called communities or modules and are commonly used in various structures of interrelated objects [4]. Suitable representation may help to extract important information about the network, simplify perception and emphasize hidden structural characteristics of linked objects. Figure 1 shows an example of network partitioned into communities with high density of internal links and low density of external links between communities.

1.1. Community definition: A community could be loosely described as a collection of vertices within a graph that are densely connected among themselves while being loosely connected to the rest of the graph.

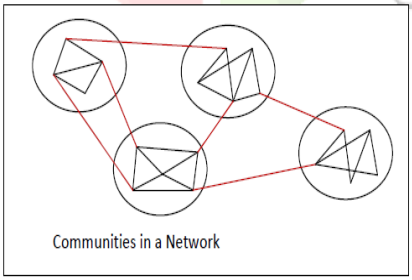

Figure 1: Communities in a Network

A community can be defined as a group of entities closer to each other in comparison to other entities of the dataset. Community is formed by individuals such that those within a group interact with each other more frequently than with those outside the group. The closeness between entities of a group can be measured via similarity or distance measures between entities. "similarity breeds connection". They discussed various social factors which lead to similar behavior or homophily in networks. The communities in social networks are analogous to clusters in networks. An individual represented by a node in graphs may not be part of just a community or a group, it may be an element of many closely associated or different groups existing in the network. For example a person may concurrently belong to college, school, friends and family groups. All such communities which have common nodes are called overlapping communities [5].

1.2. Community Detection: Community detection is basic issue that encounter during social network. Community is structural component that represent interaction of nodes and for that initially we have link predicted between any nodes to generate the community. Community detection is the extraction of related nodes and put it in specific groups. Detected Community is generally used to analyses marketing strategy of organization; find the interaction of employee in organization, to get interest of customer for specific area [6]. Using community detection algorithm, Organization can promote their product on specific area where it is not used mostly and get reason why product is not used. Sometime in organization, we can analyses effect of one node on other connected node. To analyses polling result, it is more efficient to use community detection methods[7]. Community detection methods divide each node in a group that satisfies specific properties and these groups are disjoint set of social network. Hierarchical structure of such groups and networks represent complex community structure.

In social real world networks, discovering a community implies that discovering a cluster of peoples associated with various units like images, comments, videos, blogs or the other posts. Community Discovery is of high significance in Biology, Sociology and computing disciplines where the system is usually depicted as graphs. Research that a network graph have the community structure if the vertices of the network graph are merely classified as group of vertices so that every group of vertices is closely interconnected. Discovering communities in such a composite graph could be a tough job. This research examining the various community discovery techniques and ways for discovering the good community structure in a real world network. Various community detection algorithms have been proposed with different objective functions, such as (Louvain)algorithm, algorithm map equation (InfoMap) , label propagation algorithms (LPA) , The community structure allows the division of large-scale networks and makes the subsequent analysis easier[8][9].

II. TRADITIONAL COMMUNITY DETECTION ALGORITHMS:

As we have noted in this module, no consensus has yet developed around a formal definition of community. Accordingly, a variety of community detection methods have been developed (e.g., Traditional algorithms, overlapping community detection detection algorithms in order to meet the assumptions of different definitions.

2.1. Traditional Algorithms: Basically, many clustering algorithms have been checked against the detecting community. These are old and classic algorithms, have been used in many applications apart from the social networks. traditional algorithms are hierarchical clustering algorithm, Girvan-Newman Algorithm,Partition Clustering described below,

2.1.1 Hierarchical clustering algorithm: Finding communities within an arbitrary network can be a computationally difficult task. The number of communities, if any, within the network is typically unknown and the communities are often of unequal size and/or density. Despite these difficulties, however, several methods for community finding have been developed and employed with varying levels of success [10]. Hierarchical clustering techniques can be categorized into two classes:

2.1.2. Agglomerative algorithms: It is a bottom-up technique because at the start it considers each node as a separate cluster and iteratively merges them based on high similarity and ends

up with the unique community. It is a procedure that finds the community of any vertex, although the authors also presented a more general procedure to identify the full community structure of the graph. Communities are defined locally, based on a simple criterion involving the number of edges inside and outside a group of vertices. One starts from a vertex-origin and keeps adding vertices lying on successive shells, where a shell is defined as a set of vertices at a fixed geodesic distance from the origin. The first shell includes the nearest neighbors of the origin, the second the next-to-nearest neighbors, and so on. At each iteration, one calculates the number of edges connecting vertices of the new layer to vertices inside and outside the running cluster. If the ratio of these two numbers (``emerging degree'') exceeds some predefined threshold, the vertices of the new shell are added to the cluster, otherwise the process stops. The idea of closing a community by expanding a shell has been previously introduced. in which shells are centered on hubs. However, in this procedure the number of clusters is reassigned and no cluster can contain more than one hub. Because of the local nature of the process, the L-shell method is very fast and can identify communities very quickly. Unfortunately the method works well only when the source vertex is approximately equidistant from the boundary of its community.

2.1.3.Divisive algorithms: It is a top-down technique because at the start it considers the entire network as a single cluster and iteratively splits it by eliminating links joining nodes with low similarity and ends up with unique communities. The divisive approaches start while putting the whole network in a cluster and then the split is performed recursively. it is detect inter-community links and remove them, so that the communities get disconnected from each other. In this algorithm, edges are removed according to their betweenness values. Agglomerative algorithms merge nodes into communities iteratively if their similarity is sufficiently high. The quality of the partitions resulting from these algorithms is often measured by a quality function evaluate how good a partition is. The most popularity quality function is the modularity of Newman and Girvan [3]. In this method one defines a similarity measure quantifying some (usually topological) type of similarity between node pairs. Some commonly used measures are,

2.1.4 The cosine similarity: The cosine of two non-zero vectors can be derived by using the Euclidean dot product formula:

$$p.q = \| p \|_2 \| q \|_2 \cos\theta$$

Let P and Q be two vectors of attributes, then the cosine similarity is represented as

$$similarity = \cos(\theta) = \frac{P.Q}{\| P \|_2 \| Q \|_2} = \frac{\sum_{i=1}^{n} p_i Q_i}{\sqrt{\sum_{i=1}^{n} p_i^2} \sqrt{\sum_{i=1}^{n} Q_i^2}}$$

2.1.5 The Hamming distance between rows of the adjacency matrix: Let A(G) be the adjacency matrix of a graph G. The rows of A(G) corresponding to a vertex v of G, denoted by s(v), is the string which belongs to$Z2n$, a set of n-tuples. The Hamming distance between the vertices u and v is the number of positions in which s(u) and s(v) differ.

2.2 Partition Clustering

The concept of partition or partitioning means superficially the same as clustering, that is, a separation into mutually disjoint subsets that cover the entire set of interest. Graph partitioning is typically performed to divide a large network into smaller parts for faster/more manageable processing l Community detection aims into understanding the structure of the network in a large scale § Identifying connection patterns l Hence, graph partitioning always needs to output a partition (even if it is not good), while community detection is not required to provide an answer if no good division exists. Graph partitioning is an easy problem to state and understand but it is not easy to solve l Let's start with the simplest problem of dividing a network into two equally sized parts § Graph bisection § Partitioning in an arbitrarily number of parts can be realized by repeated graph bisection § The number of edges between the two groups is called cut size ɔ Can be thought as a generalized min cut problem.

An optimal solution would require examining all the possible partitions of the network in two groups of sizes n1 and n2 respectively. This is: l Such an algorithm would require exponential time and hence, is not appropriate for even moderately small networks § We have to rely on approximation/heuristic algorithms ü They require less time but the solution is not optimal (but acceptable)[11].

2.3 Girvan-Newman Algorithm It is a divisive hierarchical clustering community detection algorithm given by Michelle Girvan and Mark Newman[12]. The steps of the algorithm are:

The steps in the Girvan-Newman algorithm are as follows.

- **Edge-Betweenness Calculation** At this step, we calculate the edge betweenness value of all edges of the graph used. The calculation of edge betweenness value done by using Breadth First Search (BFS) approach. The use of BFS to calculate the edge betweenness is done to determine the number of shortest paths from the source to each node and the distance from the source assigned to each node (node marking). Furthermore, the number of shortest paths

through the edge is calculated for each edge starting from the last node visited from the use of BFS.

- **Removal edge with the highest edge betweenness value** The removal of the edge begins at the edge that has the highest edge betweenness value in the graph used. However, if there is the same value of highest edge betweenness in a graph, then the selection for edge removal will be made randomly.
- **Re-calculation of edge betweenness value** At this step, after the edge with the highest edge betweenness value is removed, then re-calculation of edge betweenness value at each edge.

III.OVERLAPPING COMMUNITY DETECTION ALGORITHMS:

Many traditional algorithms on community detection are working on finding the communities which are disjoint where the same user can not belong to the other community. But in a social network, the user may belong to more than one community based on his hobbies, work culture, family group, old friends' group, etc. Hence, many researchers started their research on overlapping community detection [13]. The following are some of the useful algorithms in this category.

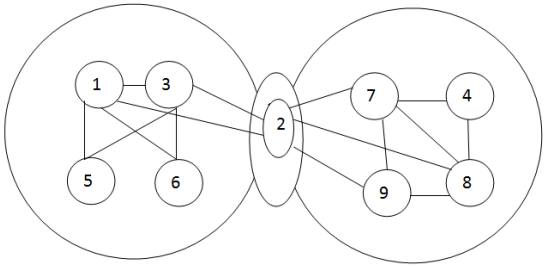

Figure 2: Overlapping community detection

3.1 Clique percolation: Clique Percolation Method (CPM) Algorithms considered as a fourth group of algorithms. All the sub graph nodes are connected with each other to form a complete graph. suggested [14] to Detecting the overlapping communities based on the traditional clique based theory algorithm. CPM correctly identifies the most overlapping communities in the large scale networks. But CPM is not suitable for high dimensionality. The CPM find the all k-cliques in the network, depends upon the k-clique size they form a sub graph in the network. Community detection techniques are widely existence for overlapping communities in the network. CPM is suitable for detecting the overlapping communities in the network, even when both the overlapping density and the diversity are high. If the network size increases automatically the computational cost become high. So, CPM is unlikely to apply in the social network.

3.2 Link Clustering: Since links usually represent unique relations among nodes, the link clustering will discover groups of links that have the same characteristics. Thus nodes naturally belong to multiple communities. In order to group the links in a network into clusters, we need to define a similarity metric as well as a clustering method. the original similarity metric limit to link pairs that share a node, which are expected to be more similar than disconnected pairs.for an undirected,unweighted network, we denote the set of node I and its neighbor as n+(i).the similarity s between link(or edges)e_{ik},e_{jk} is defined as,

$$S(e_{ik,}e_{jk}) = \frac{|n+(i) \cap n+(j)}{|n+(i) \cup n+(j)},$$

When e_{ik} and e_{jk} share a common node k. the similarity becomes 0 when two links have no common nodes. Based on the similarity metric, a single-linkage hierarchal clustering is applied to build a dendrogram.clustering dendogram at some clustering threshold, it yields link communities. The choice of the threshold is critical to the outcome of any clustering method.

In order to overlapping community detection algorithms doesnot found the average solutionand complexity of the approach becomes high in many situation.

IV.NON-OVERLAPPING COMMUNITY DETECTION ALGORITHMS:

NON overlapping community detection algorithms In communities, all node are belongs to an only single community. The following are some of the useful algorithms in this category

- Infomap Algorithm
- Label propogation Algorithm
- Louvain Algorithm

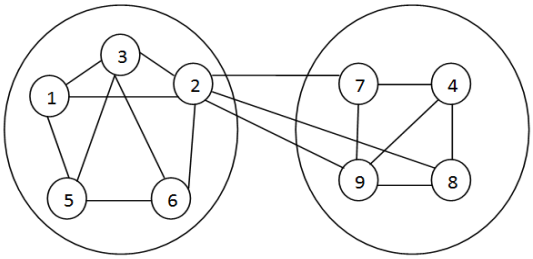

Figure 3: Non overlapping community detection

4.1 INFOMAP

The key idea of the Infomap technique[15] based on closely the Louvain methodology, nearby vertices are merged into components, which eventually are merged into super components and so on. Initially, every individual vertex is appointed to its own component. Then, in arbitrary successive steps, every vertex is shifted to the nearby component that leads to the biggest reduction of the map equation. If no pass leads to a reduction of the map equation, the vertex remains in its native component. This process is reworked, for every time in a new arbitrary subsequent order, till no pass causes a reduction of the map equation. Now the graph is reconstruct, with the components of the last level creating the vertices at

this level, and, precisely as at the preceding level, the vertices are grouped into components. This hierarchic reconstruction of the graph is replicated till the map equation can't be decreased farther. Using this technique, reasonably sensible communities of the graph may be identified in very small duration.

The Infomap algorithm is each node is assigned to a separate community. Then, in random sequential order, each node is moved to a neighboring community, if this movement decreases the map equation value. The repetition is performed each time in a new random sequential order until no movement leads to a decrease in map equation. Then, the graph is rebuilt, and the communities of the last level are replaced by nodes at this level. Then, the procedure is repeated for this level. The network is rebuilt until the map equation result cannot be reduced any more. info map algorithm work as follows,

- Exploits data compression, compressing the movement of a random walker on a network
- Finding structure in networks is equivalent to solving a coding problem
- Optimizes a quality function for networks known as the map equation

Important aims of infomap as follows,

- If we want to understand how network structure relates to system behavior, we need to understand the flow of information on the network
- A group of nodes among which information flows quickly and easily can be aggregated as a single well-connected module/community
- Succinctly describing information flow is a coding/compression problem
- Use a random walk as a proxy for information flow
- Random walk uses all of the information in the network representation and nothing more
- Finding community structure in networks is equivalent to solving a coding problem
- Want a compressed description of a random walk, with unique names for important structures

In a graph G = (V , E), V is the set of vertices and E is the set of edges. The weight of an edge between two vertices, u and v, is denoted as wu,v , which is 1 in an undirected unweighted graph. The community detection problem in Infomap is to find vertices sets, named communities (or modules), which contain high intramodule information flow but low inter-module flow.We denote the non-overlapping community set C of a graph G as:

$$\cup c_i = v, \forall c_i \in C; c_i \cap c = \theta, \forall ci, cj \in C \qquad (1)$$

For simplicity, we only consider undirected unweighted graphs. The original Infomap work [16] shows that the Infomap algorithm can be applied on both undirected and directed graphs. Therefore, our work can be easily extended to directed graphs. The map equation is the objective function of the Infomap algorithm. It is based on the information flow, and is used to find a compressed representation of a set of random walks through a graph. Its insight is that a succinct representation of a graph walk can be expressed over clusters rather than individual vertices, and the clustering that produces the shortest representation also produces the community detection result of the highest quality in practice. Infomap uses minimum description length (MDL) to measure the quality of detected community results, where a shorter
MDL means a more compressed community structure. The definition of map equation is shown in Equation 2:

$$L(M) = q_{\curvearrowleft} H(Q) + \sum_{m \in M} p^m H(p^m), \qquad (2)$$

Where M is the set of modules, qx is the sum of the exit probability of each module in the graph, H(Q) is the average code length of movements between the modules, p_ is the stay probability for the random walks in a modulem, which is equal to the sum of the exit probability and the visit probability of the random walks, and H(Pm) is the average code length of a module codebook for m. L(M) represents the lower bound on the code length of detected community structure M.

$$L(M) = (\sum_{m \in M} q_m) \log(\sum_{m \in M} q_m) - 2\sum_{m \in M} q_m) \log(\sum_{m \in M} q_m) - \sum_{\alpha \in V} p\alpha \log(P\alpha)$$
$$+ \sum_{m \in m} (q_{m} + \sum_{\alpha \in m} p\alpha) \log(q_m + \sum_{\alpha \in m} p\alpha \qquad (3)$$

Where q_m is the exit probability of a modulem, pα is the visit probability of a vertex α during the random walk, and V is the set of vertices in the graph. For an undirected graph, pα corresponds to the relative weight wα that is computed as the total weight of the links connected to the vertex α divided by twice the total weight where qmis the exit probability of a modulem, pα is the visit probability of a vertex α during the random walk, and V is the set of vertices in the graph. For an undirected graph, pα corresponds to the relative weight wα that is computed as the total weight of the links connected to the vertex α divided by twice the total weight

Algorithm 1 Pseudo code for the serial Infomap algorithm [13]

1: input: Network G = (V;E), where V = set of N vertices, E = set of edges. Per-iteration quality improvement threshold _
2: Run PageRank to calculate steady state probability for each vertex.
3: M = ffvig j vi 2 V g
4: L = L(M) in Eq. 1
5: repeat
6: Lprev = L
7: R = random sequence of integers 1 to N
8: for i = 0; i < N; i++ do

9: mnew = bestNewModule(M, vR[i]);
10: Move vR[i] to mnew module, and update M and L.
11: end for
12: until Lprev □ L < _
13: return M

Algorithm 1 illustrates the core algorithm of the fast stochastic and recursive search algorithm implemented in Infomap [17].

The algorithm proceeds in two phases:
Phase 1: (line 2) The visit probability (rank) of each vertex is computed in terms of the network flow.
Phase 2: (lines 5- 12) The space of possible modularizations is greedily searched. The initial modules are singletons — one module per vertex. For each vertex v, the call bestNewModule(M, v) (line 9) checks neighboring modules and greedily selects the one that reduces the minimum description length (MDL) L by the largest amount. The algorithm stops when the change in MDL score in each iteration (Lprev □ L) is less than a threshold . In searching procedure for the best new module of a vertex v (line 9), the algorithm calculates the in-flow and out-flow between the vertex v and its neighbor modules. Finally, the improvement in the MDL for each candidate move can be calculated from the measured in-/out-flow information. The algorithm assigns the vertex v to whichever new module maximizes the MDL improvement.

V. LABEL PROPAGATION ALGORITHM

Propagation: In each node, an entire distribution of labels is maintained and spread to neighbors. We define n 1 × n vectors Pi (n is the number of nodes) which are separate from adjacency matrix A defining the network structure. Each element Pi(c) or Pic holds the current estimation of probability of node i observing label c 2 C taken from a finite set of alphabet C. For clarity of discussion, we assume here that C = {1, 2, . . . , n} (same as node id's) and |C| = n. In Section IV we lift this assumption to increase efficiency of execution. In LabelRank, each node broadcasts the distribution to its neighbors at each time step and computes the new distribution $P_i^{'}$ simultaneously using the following equation:

$$P_i^{'}(c) = \sum_{j \in Nb(i)} P_j(c) / K_i, \forall_c \in C, \qquad (1)$$

where Nb(i) is a set of neighbors of node i and ki = |Nb(i)| is the number of neighbors. Note that, $P_i^{'}$ is normalized to make a probability distribution. In matrix form this operator can be expressed as:
In matrix form this operator can be expressed as:

A X P (2)

where A is the n × n adjacency matrix and P is the n × n *label distribution matrix*. To initialize P, each node is assigned equal probability to see each neighbor:

$$p_{ij=1} \frac{1}{k_i, \forall s.t. A_{ij}} = 1 \qquad (3)$$

Since the metric space A is usually compact, P defined iteratively by Eq. 2 converges to the same stationary distribution for most networks by the Banach fixed point theorem [18]. Hence, a method is needed for trapping the process in some local optimum in the quality space (e.g., modularity Q [19]) without propagating too far.

Algorithm:
Label propagation algorithm (LPA) to identify community in large networks, which runs linearly in the number of edges, thus linearly also in the number of nodes for sparse networks. Initially, each node is assigned a unique label. During the iterative process, each node adopts the label in agreement with the majority of its neighbors. At the end of the algorithm, connected nodes with the same label form a community. This algorithm provides a number of desirable qualities such as no parameters, easy implementation and fast execution for practical networks.

The LPA [19] uses the network structure alone to guide its process and requires neither any parameters nor optimization of the objective function. It starts from a configuration where each node has a distinct label. At every step, one node (in asynchronous version) or each node (in a synchronous version) makes its own decision to change its label to the one carried by the largest number of its neighbors. By construction, as the algorithm converges, each node has more neighbors in its own community than in any of other community. One drawback of LPA is that it returns different solutions (some of them of poor quality) in different realizations.

The basic idea of label propagation is to predict the label information of unlabeled nodes by using the topological relations between nodes from the label information of labeled nodes and finally complete the division of the graph to form a clustering structure. Although this algorithm has the advantages of simple implementation, clear logic, no need to know the number of communities in advance, time complexity is close to linearity, etc., the unstable partition results and strong randomness are the defects of this algorithm. In each iteration of the label propagation algorithm, which community a node belongs to depends on the label with the largest cumulative weight of its neighbor nodes. Therefore, when more than one of the largest neighbor labels appears on a node, one of them will be randomly selected as its own label. This kind of randomness will bring avalanche effect, that is, a small clustering result error at the beginning will be continuously amplified. In addition, the updating order of node labels will also have a great impact. Obviously, the earlier the updating of the most important node will accelerate the process of convergence[20]

Pseudo code LPA:
Algorithm 1 A basic outline of the LPA with possible enhancements.
1: Pre-processing
2: for v ∈ V do
3: ℓ(v) ← v ◅ Initialization,
phase 5.1
4: end for
5: repeat
6:synchronouslyorcasynchronously
phase 5.2
7: sequentially or in parallel
phase 5.3
8: for v ∈ (a random permutation of V) do ◅ Ordering,
phase 5.4
9: (v) ← {u | (u, v) ∈ E} ◅ Neighborhood,
phase 5.5
10: L(□(v)) ←maximum-frequency labels for □(v) ◅
Frequency, phase 5.6
11: if |L(□(v))| = 1 then
12: ℓ(v) ← L(□(v)).pop() ◅ Flat communities,
phase 6
13: else
14: ℓ(v) ← random(L(□(v))) ◅ Selection,
phase 5.6
15: end if
16: end for
17: until ∀v ∈ V: ℓ(v) ∈ L(□(v)) (with |L(□(v))| = 1) ◅
Stopping condition, phase 5.7
18: Post-processing

First, pre-processing can be done on the input graph, such as removing zero-degree singleton nodes or iterative removal of degree-one nodes to clean up the graph. Then, in the initialization phase, a (commonly unique) label is Regardless of whether or not the computation is done in parallel, the label updates may be executed synchronously (i.e., the new label of each node is computed based on the old labels of the neighbors and then all the label updates are applied, requiring the simultaneous storage of both the old and the new labels) or asynchronously (i.e., once a node updates its label, the neighbors that have not yet updated their labels will now observe the new label for the remainder of the iteration)—this is discussed in detail in phase 5.2. In the latter case, the node update order (line 8) becomes relevant; the original formulation uses a random order, but alternatives are discussed in phase 5.4. The new label for each node is chosen among the labels of its neighborhood (the definitions of which may vary, as discussed in phase 5.5). The original formulation chooses the label with the highest frequency; in case that more than one label has the maximum frequency, one is chosen uniformly at random. The chosen label is sometimes referred to as the maximal label. Modifications to the way in which a label is selected among the candidates are discussed in phase 5.6. Variants that permit assigning more than one label to each node allow the definition of overlapping communities; we discuss these in phase 6. The algorithm terminates when each node holds the same label with the majority of its neighbors (see phase 5.7 Formally, the maximal labels of v are

$$L(v) = \arg\max f(l, v), \tag{2}$$

$$f(l, v) = |\{u \mid u \in T(v) \wedge L(u) = l\}|, \tag{3}$$

Is the number of neighbors of v that have label ℓ, that is, the frequency of that label within the neighborhood? When the algorithm concludes, the communities are deduced from the labels, disambiguating any disconnected label sets: each connected induced sub-graph that shares the same label is considered a community— these can be identified by unmarking all nodes and then executing a DFS from an arbitrary, unmarked node to span over all reachable, unmarked nodes that share its label, iterating over the set of unmarked nodes until none remain.

VI. LOUVAIN ALGORITHM

Louvain algorithm(LV) was proposed In[20] for extracting the community structure of large networks. The algorithm is based on modularity optimization method and it is a heuristic approach. The algorithm is divided into two phases, which used to get repeated in every iteration. In the first phase every nodes was assigned in a community, so the number of community is equivalent to the number of nodes. For each node *x*, first it's (i.e. of *x*) neighbors *y* are considered, then the gain of modularity is quantified for the removal of node *x* from its community and placing it in the community of *y*. The removal of node *x* to community *y* happens only when there used to gain in the modularity, otherwise node *x* used to remain in its own community. The first phase continues till the attainment of local maxima of modularity is achieved. In the second phase it builds a new network which consists of the nodes for which communities are found during the first phase. To achieve the task of second phase it attains the weight of the links in between the two nodes by considering the two nodes in the corresponding two communities and then the sum of the weight of the links is calculated[21].

The Louvain method is a simple, easy-to-implement method for identifying com-munities in large networks. It is disjoint community detection, but as it can implement network data in high efficiency, the method is quite popular and has been applied successfully onto many types of massive network datasets of sizes up to 100 million nodes and billions of links. The method reveals hierarchies of communities and allows to zoom within communities to discover sub communities, sub-sub-communities, etc. Until now it is one of the most widely used methods for detecting communities in large networks. The Louvain method is a optimization method that attempts to optimize the modularity of a partition of the network.

The Louvain algorithm or multilevel algorithm is a hypervisor-based grouping analysis that works on weighted graphs. At first, every vertex is a community. Hereafter, according to the change in local edge density, the community gradually merges with its neighbors. The goal is to create neighborhoods with high fringe density, whereas in the inter-community the concentration is still limited. Louvain's type analysis represents the perceptual notion of edge density in terms of modularity and the scale m from □1 to +1 is set as follows:

$$m = \{\frac{1}{2|E|}\sum(i,j)(w_{i,j} - \frac{\deg ree(\upsilon_j)}{2|E|}), \nu_i \in c_i \wedge \in c_j$$

(13)

In ci and cj represent the community to which vi and vj belong, and wi,j is the weight of (i, j). Although Louvain's type of analysis applies to non-weighted graphs, the outcome is invariably a weighted graph, in which the weights are dependent on the local densities of the edges. Non-weighted graphs are considered graphs with original weight of 1. The maximization of modularity is implemented through a set of two distinct stages. During the initial stage, every vi is joined with each one of its adjacent in a grouping C, and the change in the modularity Dm is computed as the change of the new type minus the old. Eventually, vi is delegated to cj, resulting in a larger Dm. Note that in the second stage, we build a new graph which merges the vertices that belong to the common grouping to one vertex. Moreover, all vertices linking both groupings together create an vertex of which the weight is the total of the many

The main steps of the Louvain algorithm are as follows:

Step 1: Initialize the community and set each node as a separate community, namely, community 1: (node1), community 2: (node2), and so on
Step 2: Find out all the communities connected to node 1, and calculate the change of modularity after moving node 2 to each neighbor community. Move node 1 to the community, which can increase the modularity to the maximum.
Step 3: Iterate over all the nodes and execute step 2 until there are no nodes to move and get a layer of community partition.
Step 4: Merge each community in step 3 into a new node. ,e relationship between new nodes is the relationship between the original communities. Return to step 1 until all nodes are finally merged into one community. The multilevel community partition is obtained, and the partition with the highest modularity is selected as the final partition result.

Algorithm 1 Pseudo-code of Louvain Method
1: G the initial network
2: repeat
3: Put each node of G in its own community
4: while some nodes are moved do
5: for all node n of G do
6: place n in its neighboring community including its own which maximizes the modularity gain
7: end for
8: end while
9: if the new modularity is higher than the initial then
10: G = the network between communities of G
11: else
12: Terminate
13: end if
14: until

Louvain Method is a hierarchical greedy algorithm. It is composed of two phases, executed alternatively. Initially, each node is in its own community. Then, during phase 1, nodes are considered one by one, and each one is placed in the neighboring community (including its own community) which maximizes the modularity gain. This phase is repeated until no node is moved (the obtained decomposition is therefore a local maxima). Then, phase 2 consists in building the graph between communities obtained during phase 1: there is a node in the new graph for each community and, for two communities C and C0, there is a link of weight w where $\omega = \sum \nu, \upsilon' \in CXCweight(\upsilon, \upsilon')^1$ weight (v; v0). There is also a loop on C of weight $\sum \nu, \upsilon' \in CXCweight(\upsilon, \upsilon')^1$ then; the algorithm starts the phase 1 again with the new graph, grouping communities together, and then phase 2, and so on until the modularity does not improve. The whole process is described on algorithm 1. However, Louvain Method is the fastest and most accurate method

6.1 Modularity gain: Modularity is a mathematical function which measures the structure of networks. It is designed for the measurement of strength of division of a network into communities (also known as groups, clusters or modules). Communities with high value of modularity have dense connections within that community while communities with low value of modularity have sparse or scattered connections between nodes in different clusters. Modularity is defined as the difference of value obtained by the fraction of the edges that are within the module to the expected such fraction if edges were randomly distributed

$$Q = \sum_{c \in C}(\frac{\sum_{in}^{c}}{} - (\frac{\sum^{c}}{\frac{tot}{2m}})^2), \qquad (2)$$

where *m* is the sum of all edge weights in the graph, $\sum_{in}^{c}$ *in* is the sum of all internal edge weights in a community *c*,

calculated as $\sum_{in}^{c} = \sum \omega_{u,v}(u \in c \wedge \upsilon \in c)$, and $\sum_{tot}^{c}$ is the sum of all edge weights, calculated as $\sum_{tot}^{c} = \sum \omega_{u,v}(u \in c \wedge \upsilon \in c)$, The intuition of Equation 2 is that if the modularity value is high, there are many edges inside communities but only a few between communities, indicating a high quality of community detection. δQ, is the gain in modularity obtained by moving an isolated vertex u into a community $c \in C$ [1], which can be computed by:

$$\delta Q_{u \to c} = \frac{1}{2m}(w_{u \to c} - \frac{\sum_{tot}^{c} * w(u)}{m}),$$

Where $wu \to c$ is the total weight of edges connecting a vertex u and a community c, and $w(u)$ is the weighted degree of u. The Louvain algorithm iterates multiple stages for computing a hierarchical clustering of the vertices in G. Each stage consists of two phases. In this phase (named vertex movement), each vertex is considered in turn and moved to a community with the maximum modularity gain given by Equation 3. The vertex will remain in its current community if no positive modularity gain can be achieved. This indeed employs a greedy strategy to compute graph clustering that optimizes the modularity given by Equation 2. The algorithm continues iterations until no further gain can be obtained or if the gain falls below some predefined threshold. In the second phase, the communities generated in the first phase are represented as a new graph with vertices of each community merged into a single new vertex. If there are multiple edges between different communities, these coalesce into one edge between the new vertices. The weight of a new edge is the sum of the weights of all the edges merged into it. The new aggregated graph is then iteratively input to the next stage of the algorithm. This whole process continues until the graph is stable (no modularity change)[22].

The modularity is optimized by means of two phases or steps that are repeated iteratively:
1. each node of the network is assigned to its own community and the modularity is optimized locally.
2. nodes in the same community are grouped and a new network is build where nodes are communities from the previous step.
The links between nodes of the same community are represented with a selfloop on the community node of the new network and links between nodes in different communities are represented as weighted links between community nodes.
Figure 4 shows a sample iteration of the algorithm.

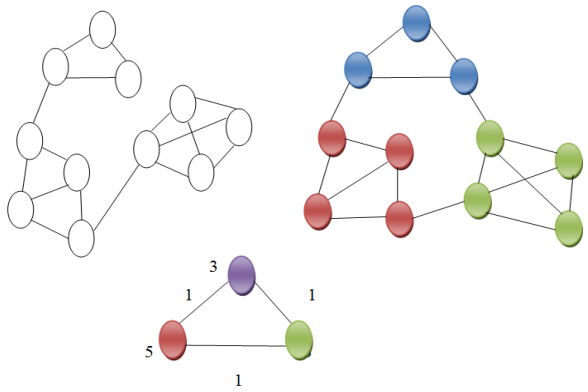

Figure 4: First iteration of the Louvain algorithm on a simple network

VII. DATA SETS USED IN COMMUNITY DETECTION

For study of the various community detection technique, we need to carry out those algorithms onto the different benchmark data sets. Community detection technique generally use some data sets given below:

(1). Zachary's Karate Club network-

This graph is a benchmark graph for community detection algorithms. Zachary is the members of a university karate club in 1977 uses the social network information of all the candidate of the club.

(2) Amazon network

Amazon is website for an online shopping. Amazon makes a data set on the purchasing pattern of the items. In this connection, items are served as nodes.

(3) YouTube network

On the basis of contents, users make friendships with each other, or users can Make a community with each other.

(4) Football network

Football network is a collection of 115 college teams of U.S. colleges. Edge between two if they played a match against each other during the year 2000.

VIII. EXPERIEMENTAL RESULTS

In this paper, normalized mutual information (NMI) is used as the evaluation measure which is currently widely used in measuring the quality of detected communities. NMI allows us to measure the amount of information common to two different network partitions. Accordingly, if a network has a known community structure, one can explore the efficacy of the algorithm by comparing known real partition with the partition found by that algorithm. The test will be performed using four data sets, consisting of two social network datasets and two other datasets. Each of these datasets will test with all three algorithms, Label Propagation, INFOMAP, and LOUVAIN algorithm.

Modularity of communities

Possible measure for evaluation of community decomposition is the so–called "Modularity" Given an undirected graph $G(E,V)$, where each node is assigned to one of C possible communities, the modularity of the decomposition is defined as

$$Q = \frac{1}{2m}\sum_{i,j\in v}[A_{ij} - \frac{k_i k_j}{2m}]\delta(c_i, c_j)$$

Where:
Aij are the elements of the adjacency matrix of G(E, V)
ki is the out-degree of node i
m = |E|
ɔ(ci , cj) is equal to 1 if i and j belong to the same community, and is
equal to 0 otherwise

The louvian algorithm,infomap algorithm,label propagation algorithm outperforms with values ranging from 0.586 to 0.655. On the other hand, Breadth-First Search and MaxToMin perform poorly in contrast to the other three community-detection algorithms. It is worth mentioning that higher values of modularity are in the football and dolphins datasets, The modularity metric results are also depicted in Figure 1.

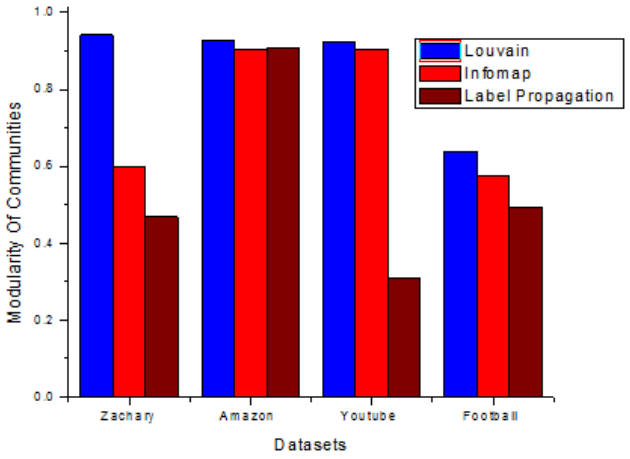

Figure 5: Modularity Of Communities

Computational time: We compare the Louvain algorithm with the info map algorithm and the label propagation algorithm. The Louvain algorithm is one of the fastest and most efficient non-overlapping community detection algorithms, and the info map algorithm is one of the most popular overlapping community algorithms. Fig. 1 shows that our models achieve better performance on all datasets than other baseline algorithms in all networks, especially for larger networks of tens of million nodes. Whereas the infomp algorithm and the labelpropogation algorithm cannot deal with such a large dataset. Besides, when detecting the same size of communities, the Louvain algorithm is faster than compared methods by two to three orders of magnitude. Compared with the infomap, label propogation and louvain algorithm has the advantage of fewer computation times per iteration. Meanwhile, when calculating the movement of a single node, only the modularity gain of the node in the two communities where the movement occurs needs to be calculated. So, the computational efficiency is obviously better than that of the infomap,Label propagation algorithm

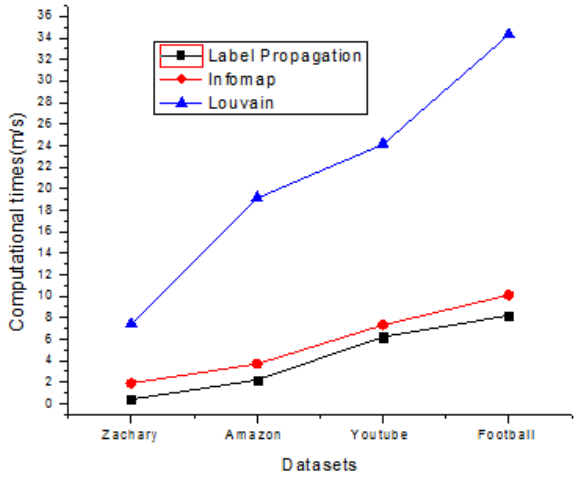

Figure 6:Computational Times

Normalized Mutual information : Normalized Mutual Information (NMI) is a measure used to evaluate network partitioning performed by community finding algorithms. NMI is a variant of a common measure in information theory called Mutual Information. NMI measured as,

$$NMI(Y,C) = \frac{2\times I(Y;C)}{[H(Y)+H(C)]}$$

Where,
1) Y = class labels
2) C = cluster labels
3) H(.) = Entropy
4) I(Y;C) = Mutual Information b/w Y and C

The results of the tested algorithms in respect to the NMI metric for the four distinct datasets are given. Louvain algorithm achieve the best performance in almost all datasets whereas label propagation and infomap have the lower values. Concretely, regarding the dolphins dataset, Propinquity Dynamics and Newman–Girvan have the higher values and MaxToMin with Breadth-First Search have the lower ones. In terms of the football dataset, all the algorithms have almost the same performance with values ranging from 0.903 to 0.926. In the karate dataset, MaxToMin along with Propinquity Dynamics and Newman–Girvan perform equally well, and Breadth- First Search has the worst value, e.g., 0.309. Finally, in Polbooks dataset, the six algorithms have the lowest values

in contrast to the other three datasets, with Newman–Girvan having the best value. These results are also illustrated in Figure 7.

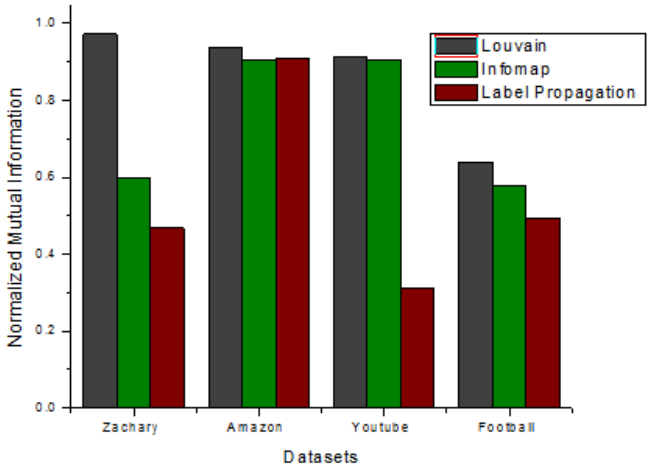

Figure 7:Normalized Mutual Information

Execution Time: Figure 8 illustrates the running time of the weighting Algorithm 4 where n ranges from 0.3 seconds to 34.3 seconds. As it can be seen from this figure, the weighting Algorithm 3 consumes less than 0.3 seconds for finding weights of this network .With increasing the number of node n to 500,000, the consumed time increase near linearly. Therefore, finding weights of all links of a network with 500,000 nodes and around 10 million links requires less than 0.3 seconds. Similarly, for evaluating the scalability of Louvain algorithm,, the average running time of Louvain algorithm approximately linearly with n figure shows that compared with infomap and label propagation algorithm better result from the Louvain algorithm for execution time.

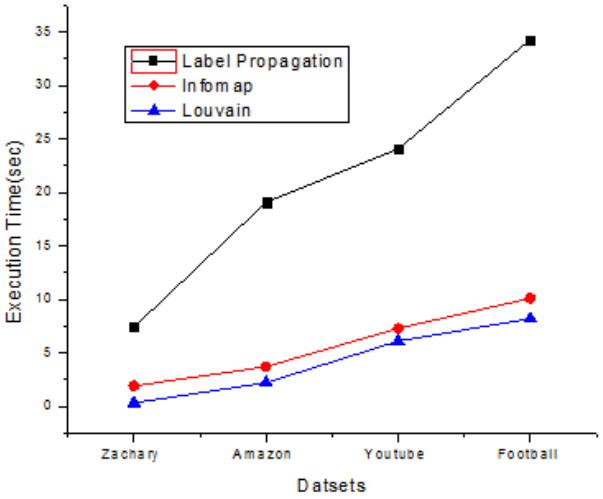

Figure 8: Execution Time

Accuracy: The proposed framework gives very easy platform for expressing relative inclination of an algorithm towards accuracy, which otherwise were very difficult. The advantage of proposed algorithm can be understood with the actual indications in metric values as follows. Performance of all three algorithms indicated by each of the accuracy and quality metrics are ease of understanding, average values of accuracy metrics and quality metrics are considered. Clearly, Louvain algorithm show higher accuracy and quality metrics values in dataset.

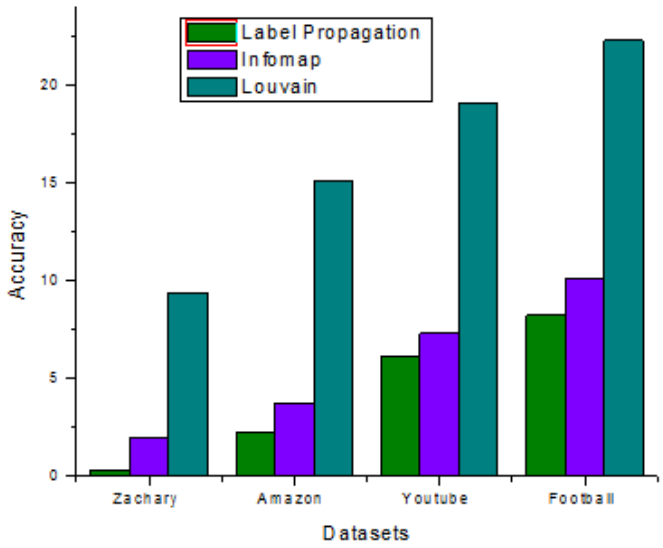

Figure 9:Accuracy

IX.CONCLUSION

With the growth of Social Networks this area become area having very vast potential of research to find more better and fast algorithms for community detection. Use of community detection algorithms provides the need of research in this field. Information provided by community detection may be helpful in development, commercialization and in education. From the results of the analysis of the four datasets used in this research known the Louvain algorithm is superior to Info map and Label Propagation algorithm. Many community detection algorithms are simple and relatively easy but slow, high in Complexity. Compared to infomap, Label propagation algorithms better result in Louvain algorithms which have NMI,Accuracy will be high and efficiency. Louvain is better with respect to execution time.

X.REFERENCES

[1]. R. Aktunc and I. H. Toroslu, "A Dynamic Modularity Based Community Detection Algorithm for Largescale Networks : DSLM," pp. 1177–1183, 2015

[2]. T. V. Batura, "Methods of analysis of computer social networks," Vestn. NGU, Ser. Inform. Technol., 10, No. 4, 13–28 (2012).

[3]. V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, "The Louvain method for community

detection in large networks," J. Stat. Mech. Theory Exp., 108–121 (2008).

[4]. A. N. Churakov, "Social network analysis," SocIs, 1, 109–121 (2001).

[5]. S. Fortunato, "Community detection in graphs," Phys. Rep**.,** 486, 75–174 (2010).

[6]. M. Girvan and M. E. Newman, "Community structure in social and biological networks," Proc. Natl. Acad. Sci. USA, 99, 7821–7826 (2002).

[7]. Ellison, Nicole B. Social network sites: Denition, history, and scholarshipJournal of ComputerMediated Communication 13.1 (2007): 210-230.

[8]. Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. 2008. Fast unfolding of communities in large networks. Journal of Statistical Mechanics: Theory and Experiment 2008, 10 (Oct 2008), P10008. https: //doi.org/10.1088/1742-5468/2008/10/p10008

[9]. Tengjiao Wang, Jingbo Tan, Wenbo Ding, Yanru Zhang, Fang Yang, Jian Song, and Zhu Han. 2018. Intercommunity Detection Scheme for Social Internet of Things: Compressive Sensing Over Graphs Approach. IEEE Internet of Things Journal 5, 6 (2018), 4550–4557. https://doi.org/10.1109/JIOT.2018.2837048

[10]. M. A. Porter; J.-P. Onnela; P. J. Mucha (2009). "Communities in Networks" . Notices of the American Mathematical Society. 56: 1082–1097, 1164–1166.

[11]. Flávio gabriel duarte and leandro nunes de castro,"A Framework to Perform Asset Allocation Based on Partitional Clustering",IEEE Acess, volume 8, 2020

[12]. M.E.J. Newman, M. Girvan, "Finding and evaluating community structure in networks", Phys. Rev. E 69 (2) (2004) 026113.

[13]. U.N. Raghavan, R. Albert, and S. Kumara, "Near linear t ime algorithm to detect community st ructures in large-scale networks", Phys. Rev. E 76 (2007) 036106.

[14]. Y. Su, B. Wang, and X. Zhang, "A seedexpanding method based on random walks for community detection in networks with ambiguous community structures," Sci. Rep., vol. 7, p. 41830, Feb. 2017

[15]. Martin Rosvall and Carl T. Bergstrom*‡," Maps of random walks on complex networks reveal community structure", Department of Biology, University of Washington, Seattle, WA 98195-1800; and Santa Fe Institute, 1399 Hyde Park Road, Santa Fe, NM 87501, 1118–1123, vol 105,no 4,2008

[16]. Martin Rosvall, Daniel Axelsson, and Carl T Bergstrom. 2009. The map equation.The European Physical Journal Special Topics 178, 1 (2009), 13–23.

[17]. Seung-Hee Bae_, Daniel Halperin_, Jevin Westy, Martin Rosvallz and Bill Howe,"Scalable Flow-Based Community Detection for Large-Scale Network Analysis", Department of Computer Science and Engineering, University of Washington, Seattle, Washington

[18]. S. Banach. Sur les op´erations dans les ensembles abstraits et leur application aux ´equations int´egrales. *Fund. Math.*, 1922. M. E. J. Newman. Fast algorithm for detecting community structure in networks. *Phys. Rev. E*, 69:066133, 2004.

[19]. U. N. Raghavan, R. Albert, and S. Kumara, "Near linear time algorithm to detect community structures in large-scale networks," *Phys. Rev. E*, vol. 76, p. 036106, 2007.

[20]. W. Liu, J. Wang, and S. F. Chang, "Robust and scalable graphbased semi-supervised learning," Proceedings of the IEEE, vol. 100, no. 9, 2012.

[21]. V. D. Blondel, J. L. Guillaume, R. Lambiotte, and E. Lefebvre, "Fast unfolding of communities in large networks", *Journal of Statistical Mechanics*, *DOI:10.1088/1742-5468/2008/10/P10008*, 2008.

[22]. S.-H. Bae, D. Halperin, J. D. West, M. Rosvall, and B. Howe, "Scalable and efficient flow based community detection for large-scale graph analysis," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 11, no. 3, p. 32, 2017.

