Final version of the accepted paper. Cite as: *"Jayati Gulati and Muhammad Abulaish, A Novel Snowball-Chain Approach for Detecting Community Structures in Social Graphs, In Proceedings of the IEEE Symposium on Computational Intelligence and Data Mining (IEEE-CIDM), Xiamen, China, December 6-9, 2019, pp. 1-8."*

A Novel Snowball-Chain Approach for Detecting Community Structures in Social Graphs

Jayati Gulati
Department of Computer Science
South Asian University, New Delhi, India
gulati.jayati@gmail.com

Muhammad Abulaish, *SMIEEE*
Department of Computer Science
South Asian University, New Delhi, India
abulaish@sau.ac.in

***Abstract*—Community detection in social networks has been a widely explored problem to gain information about the dense groups of user favoring some particular idea or topic, or sharing common interests. Inspired from the process of snowball sampling, this paper presents a novel community detection approach, termed as snowball-chain (SbChain), for identifying communities in social networks. SbChain follows a bottom-up approach to find the most prominent nodes based on the degree of overlapping neighbors and clustering coefficient, that may form some cliques. The novelty of SbChain lies in a single overlapping hyperparameter requirement, λ, to merge snowballs to form a community. λ also helps in deciding the coarseness of communities to be identified from the social graph. The proposed approach is evaluated over different real-world and synthetic benchmark datasets and compared with some state-of-the-art methods in terms of the number of identified communities and their modularity.**

***Index Terms*—Social network analysis, Community detection, Snowball sampling, Clique, Modularity.**

I. Introduction

In recent times, there has been a monumental growth in study of networks like the World Wide Web, online social networks (e.g., Twitter and Facebook), metabolic networks, neural networks etc. [1]. These networks can be modeled as complex graphs and analysis of the various dynamic interactions among their entities might be beneficial to handle various real-life applications. Community detection problem is one of the core problems of the social network analysis, and it mainly aims to find densely connected nodes from a network that preferably form cliques [2]. A k-clique is a complete graph having k number of vertices [3]. A community in a network is represented by a set of nodes with high density links among themselves and low density links among inter-community nodes [4]. The nodes within a community may have similar characteristics as compared to the nodes outside the community.

Since community detection is a well-studied problem, a number of community detection methods has been proposed by different researchers [5]–[7]. Some of the well-known methods include density-based approaches, hierarchical approaches, label propagation-based approaches, and random walks-based approaches. In this paper, we propose a snowball-chain-based approach, SbChain, to identify community from social networks, where entities are represented as nodes and their relations as edges, which are usually unweighted and undirected in nature. The term snowball-chain used in this paper is inspired from the snowball sampling technique [8], in which a random sample of individuals is drawn from a finite population; each of these individual then recommends another k individuals, and this process goes on till the desired number of samples are collected.

In our proposed approach, seed nodes are chosen based on a certain criteria from the initial population. Thereafter, these nodes find their best neighbors and merge with them to form snowball-chains, which eventually lead to the formation of communities. The aim is to find well-connected nodes, as they are more likely to form dense groups. The proposed approach is in line to the Label Propagation Algorithm (LPA) [5], which re-labels a node based on the label frequency count of its neighboring nodes. Like LPA, SbChain merges nodes with their best neighbors to form snowballs. The snowballs roll and grow by merging with their best suited neighbor in every iteration, till an optimized cut is obtained by a high modularity value. The novelty of SbChain lies in its simple approach for finding communities, based on the local and global clustering coefficients and common neighbors. Moreover, it uses only a single hyperparameter λ, whose value is determined empirically to set the level of coarseness of the communities.

The rest of the paper is organized as follows. Section II presents a brief review of the related works in the field of community detection. Section III describes the preliminaries used in the subsequent sections. Section IV presents the SbChain approach and respective algorithms for community detection. Section V presents the experimental setup and evaluation results, followed by section VI, which presents the complexity analysis of the proposed approach. Finally, section VII concludes the paper with future directions of research.

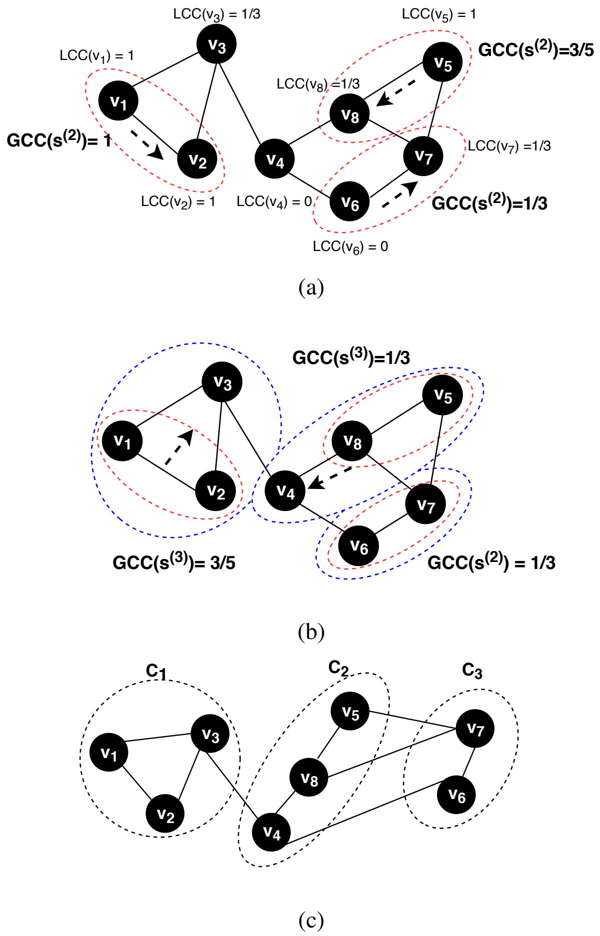
II. Related Work

Initial works on finding communities based on random walk were Markov clustering [9], Walktrap [10], etc. Seed set expansion [11] is a locally optimized random walk-based algorithm to find overlapping communities. Initial seeds are found in the seeding phase that are further expanded using PageRank scheme. The proposed work in [12] is inspired from [13], and it uses the structure of the network and edge weights to find overlapping communities. It is a two-phase approach which starts with identifying communities using random walks. If the probability of visiting nodes is higher, then they tend to group together. Further, the clusters are refined by calculating overlapping coefficient between each pair of cluster and merging them.

As the real-world data tends to lie in several groups, finding overlapping communities becomes an important research area. One of the common node based overlapping approach is Clique Percolation Method (CPM) which forms communities from k-cliques [14]–[16]. Maximal union of adjacent *k-cliques* forms communities. And two k-cliques are adjacent if they have $k-1$ nodes in common [15]. Other node-based overlapping algorithm is reported in [17], where a fitness function is calculated based on internal and external degrees in a subgroup. The neighbors that contribute to the fitness function are added to the subgroup, and those negating the function are removed. Thus, a local maxima for each node is obtained. Overlapping communities are also determined by link-based strategies because links have a unique identity of belonging to various communities [18]. The studies in [19], [20] are based on Link Clustering (LC). In [19], LC calculates the link similarity of the neighbor links and constructs a transformation matrix, and hierarchical clustering technique is applied to generate a dendrogram with partition density. The maximum density value can be determined to decide the best cut.

Over the years many hierarchy-based community detection algorithms have been proposed by various researchers. Newman and Girvan proposed an approach based on removing the edges having high edge betweenness. The optimized community cut was decided on the value of modularity Q [21]. A similar work by Newman in [22] is based on agglomerative clustering, where the nodes that maximize the modularity are combined together. Another work in [23], proposes global maximization of modularity function by using spectral clustering. The input graph is represented in the Euclidean space and *k-means* clustering is applied to detect communities. A local community detection approach in [24] starts by finding the high degree node locally, termed as local degree central node, and the degree of this node is either greater or equal to the degree of its neighboring nodes.

Among the node similarity based approaches, a work in [25] uses local information to find nodes similar to a seed node from the degree of their common neighbors. The similarity function finds the neighbor with maximum value and adds it to the community of the given node. This maximum valued node further finds its best neighbor until all of the nodes are visited. Another work in [7] extends the above work by creating levels of similarity function along with label propagation, called the Stepping LPA-S. On one level, similarity is calculated between nodes to form sub-networks with same labels. Further, similarity of these sub-networks is calculated based on another similarity function. Finally, sub-networks are combined to form communities. The original Label Propagation Algorithm (LPA) [5] proposed to change each node's label to the most frequently used label in its neighborhood. The process continues till all the nodes are updated and is verified by a modularity cut.



(a)

(b)

(c)

Fig. 1: Growth of communities by `SbChain` approach

Some recent studies in community detection [26] suggest label propagation based on a benefit score of immediate neighbors using boundary nodes. Also, a similar study in [27], finds a preference network where the connected components form communities. This network is built by finding preference nodes using maximum common neighbors between a selector node and its immediate neighbors or by using spread capability, calculated by gossip algorithm in [28]. Another similarity-based approach called Community Detection Algorithm based on Structural Similarity (CDASS) is proposed in [29]. The CDASS works in two phases; in the first phase, low similarity edges are removed, causing the network to break into several disconnected components. These communities are later merged to form final communities. In the second phase, best communities are identified from the final communities based on an evaluation function, calculated using internal and external edges, internal degree, and total degree.

III. Preliminaries

A graph $G(V, E)$ is defined as a set of n nodes $V = \{v_i, v_j, ...v_n\}$ and and set of edges $E = \{e_{ij} =$

TABLE I: Notations and their descriptions

Notation	Description
$\mathcal{N}(v_i)$	Set of immediate neighbors of a node v_i
$k_i = \|\mathcal{N}(v_i)\|$	Degree of a node v_i
a_{ij}	Adjacency matrix value for nodes v_i and v_j
$LCC(v_i)$	Local clustering coefficient of a node v_i
$\mathcal{N}_{best}(\mathcal{V})$	Best scoring neighbor set of a set $\mathcal{V} \subseteq V$
$s^{(n)}$	A snowball containing n elements
$\mathcal{N}(s^{(n)})$	Neighbor set of a snowball $s^{(n)}$
$GCC(s^{(n)})$	Global clustering coefficient of a snowball $s^{(n)}$

$(v_i, v_j)|\ v_i, v_j \in V$ & $\exists$ a link between v_i and $v_j\}$. The aim is to find the seed nodes which will grow to form snowballs, and eventually lead to formation of communities. A list of symbols used in this paper and their brief descriptions is presented in table I.

For a given graph G, the process starts by arranging nodes in non-increasing order of their respective local clustering coefficient, LCC[1], as given in equation (1).

$$LCC(v_i) = \frac{2 \times |\{\, e_{jk} \in E \mid v_j, v_k \in \mathcal{N}(v_i) \,\}|}{k_i(k_i - 1)} \tag{1}$$

The nodes are picked-up one-by-one (in order). If v_i is the first selected node, then its best matching node denoted by $\mathcal{N}_{best}(v_i)$ from $\mathcal{N}(v_i)$ should fulfil two criteria: (i) $\mathcal{N}_{best}(v_i)$ should have the maximum value of LCC among $\mathcal{N}(v_i)$, and (ii) the cardinality of overlap between $\mathcal{N}(v_i)$ and $\mathcal{N}(\mathcal{N}_{best}(v_i))$ should be maximum among neighbors of v_i, i.e, $\mathcal{N}(v_i)$, given by equation (2).

$$weight = \frac{|\mathcal{N}(v_i) \cap \mathcal{N}(\mathcal{N}_{best}(v_i))|}{min\{|\mathcal{N}(v_i)|, |\mathcal{N}(\mathcal{N}_{best}(v_i))|\}} \tag{2}$$

Further, seed nodes combine to form snowballs. The score and neighbor sets of the snowballs are updated as they grow. These snowballs grow in every iteration and eventually form communities.

Definition 1. *(Seed node). A node $v_i \in V$ is said to be a seed node if it works as a attractor in the first round of snowball formation.*

A seed node follows non-redundant node startegy, i.e., once a seed node v_i finds its best neighbor $\mathcal{N}_{best}(v_i)$, then both, v_i and $\mathcal{N}_{best}(v_i)$ are not allowed to join any other node in the current iteration. This strategy results in disjoint communities.

Definition 2. *(Snowball). A snowball $s^{(n)}$ is a connected component formed by enumerating nodes as a set, where n is the number of nodes contained in it. It is formed either by combining a seed node v_i with $\mathcal{N}_{best}(v_i)$ or by combining two or more snowballs.*

It is important to note that there may exist one or more snowballs with a similar value of n, however they can be differentiated by their respective set of elements. It can be seen from fig. 1, a snowball $s^{(2)}$ is formed by joining a seed node v_1 with $v_2 = \mathcal{N}_{best}(v_1)$ in the first iteration. In the second iteration, $s^{(2)}$ joins another node v_3 to form $s^{(3)}$ represented by equation (3).

$$s^{(3)} = s^{(2)} \cup \{v_3\} = \{v_1, v_2, v_3\} \tag{3}$$

The neighbor set and the score of a snowball are updated as per the definition 3 and 4, respectively.

Definition 3. *(Neighbors of a snowball). The neighbor set of a snowball represented by $\mathcal{N}(s^{(n)})$ is the combined set of neighbors of the nodes comprised by the snowball $s^{(n)}$, i.e., $v_1, v_2, ..., v_n$, given by equation (4).*

$$\mathcal{N}(s^{(n)}) = \mathcal{N}(v_1) \cup \mathcal{N}(v_2) .... \cup \mathcal{N}(v_n) \tag{4}$$

Definition 4. *(Score of a snowball). The score of a snowball represented by $GCC(s^{(n)})$ is the global clustering coefficient calculated by considering the subnetwork formed by the nodes $v_1, v_2, ..., v_n$ and their immediate neighbors. It is calculated by equation (5).*

$$GCC(s^{(n)}) = \frac{3\ \times Number\ of\ triangles}{Total\ number\ of\ triplets} \tag{5}$$

Fig. 1 shows the changing GCC value for snowballs with each iteration. A snowball keeps expanding till its $weight$, given by equation (2) is greater than or equal to the overlapping parameter λ value.

Definition 5. *(Community set). A set of community may comprise of single nodes or snowballs or both, which cannot be further combined with each other and have maximum modularity value among all the iterations.*

IV. Proposed Approach

In this section, we present the functional details of the proposed snowball-chain-based community detection approach, `SbChain`. This approach works well for a network which is undirected and unweighted. It starts in a bottom-up manner by finding the nodes that may be a part of cliques, and keeps adding nodes to grow the cliques to form snowballs. The snowballs keep expanding until convergence, i.e., when the community set of an iteration is same as the communities identified in the previous iteration. The set with largest modularity value forms the final set of communities.

A. `SbChain` Algorithm

The `SbChain` algorithm begins with finding the initial set of neighbors and local clustering coefficient (eq. (1)) for each node which is represented as a set. These $\langle nodes, values \rangle$ pairs are added to $N^{(0)}$ and $scoreList^{(0)}$, respectively, where superscript (0) represents the zero iteration, as shown in step 4 and 5 of the Algorithm 2. The maximum number of iterations for this algorithm is set to the number of nodes in the network. However, it never runs for maximum iterations as it converges when the community set formed in consecutive iterations are identical. As each iteration i begins, nodes (or snowballs)

[1] For an undirected graph each edge is counted twice, hence there exists a factor of two in the numerator.

Algorithm 1: $BestNeighbor(\mathcal{V}, N, scoreList)$

```
Input  : A set $\mathcal{V}$ containing one or more nodes,
         neighbor list $N$ and a $scoreList$ both
         containing $\langle keyList, value\rangle$ pairs
Output: Best scoring neighbor set $\mathcal{N}_{best}(\mathcal{V})$ of $\mathcal{V}$ and
         its $weight$
1  $maxScore \leftarrow 0,\ maxWeight \leftarrow 0$
2  $\mathcal{V}_i \leftarrow \emptyset$
3  foreach $v \in N[\mathcal{V}]$ do
4      foreach $key\ in\ scoreList.keys$ do
           // $key$ is a set of one or more nodes
5          if $v\ is\ a\ part\ of\ key$ then
6              $\mathcal{V}_i \leftarrow key$
7              go to 10
8          end
9      end
10     $weight[\mathcal{V}_i] \leftarrow \frac{\left|N[\mathcal{V}]\cap N[\mathcal{V}_i]\right|}{min\left\{\left|N[\mathcal{V}]\right|,\left|N[\mathcal{V}_i]\right|\right\}}$
11     if $scoreList[\mathcal{V}_i] > maxScore\ and$
        $weight[\mathcal{V}_i] > maxWeight$ then
12         $maxScore \leftarrow scoreList[\mathcal{V}_i]$
13         $maxWeight \leftarrow weight[\mathcal{V}_i]$
14         $\mathcal{N}_{best}(\mathcal{V}) \leftarrow \mathcal{V}_i$
15     end
16 end
17 return $\mathcal{N}_{best}(\mathcal{V}), maxWeight$
```

Algorithm 2: $SbChain(G, \lambda)$

```
Input  : A graph $G(V, E)$ and overlapping parameter
         $\lambda$
Output: Community set $C$ and its modularity $Q$
1  foreach $v_i \in V$ do
2      $\mathcal{N}(v_i) \leftarrow Neighbor(G, v_i)$
3      $LCC(v_i) \leftarrow LocalCC(G, v_i)$
4      Append $\langle\{v_i\}, \mathcal{N}(v_i)\rangle$ into $N^{(0)}$
5      Append $\langle\{v_i\}, LCC(v_i)\rangle$ into $scoreList^{(0)}$
6  end
7  $maxQ \leftarrow 0$
8  $m \leftarrow |E|$
```

Algorithm 2: $SbChain(G, \lambda)(Contd.)$

```
9  for $i \leftarrow 1\ to\ |V|$ do
10     $scoreList^{(i)} \leftarrow \emptyset$
11     Arrange $scoreList^{(i-1)}$ in non-increasing order
12     foreach $\mathcal{V}_j \in scoreList^{(i-1)}.keys$ do
           // $\mathcal{V}_j$ is a set of one or more nodes
              (keys)
13         $\langle\mathcal{V}_k, weight\rangle \leftarrow BestNeighbor(\mathcal{V}_j, N^{(i-1)},$
              $scoreList^{(i-1)})$ // $\mathcal{V}_k$ is the best
              neighbor of $\mathcal{V}_j$
14         if $\mathcal{V}_j \in scoreList^{(i)}.keys$ OR
            $\mathcal{V}_k \in scoreList^{(i)}.keys$ then
15             go to 12
16         if $|\mathcal{V}_j| > 1$ AND $|\mathcal{V}_k| > 1$ then
17             if $weight \geq \lambda$ then
18                 go to 21
19             else
20                 go to 12
21         $n \leftarrow |\mathcal{V}_j \cup \mathcal{V}_k|$
22         $s^{(n)} \leftarrow \mathcal{V}_j \cup \mathcal{V}_k$
23         $\mathcal{N}(s^{(n)}) \leftarrow N^{(i-1)}[\mathcal{V}_j] \cup N^{(i-1)}[\mathcal{V}_k]$
           // updated neighbors and scores of a
              snowball
24         Append $\langle s^{(n)}, \mathcal{N}(s^{(n)})\rangle$ into $N^{(i)}$
25         Append $\langle s^{(n)}, GlobalCC(G, \mathcal{N}(s^{(n)}))\rangle$ into
            $scoreList^{(i)}$
26     foreach
        $\mathcal{V}_j \in scoreList^{(i-1)}.keys - scoreList^{(i)}.keys$ do
27         Append $\langle\mathcal{V}_j, N^{(i-1)}[\mathcal{V}_j]\rangle$ into $N^{(i)}$
28         Append $\langle\mathcal{V}_j, scoreList^{(i-1)}[\mathcal{V}_j]\rangle$ into
            $scoreList^{(i)}$
29     $comm\_list \leftarrow scoreList^{(i)}.keys$
30     $Q \leftarrow Modularity(m, comm\_list, E)$
31     if $maxQ < Q$ then
32         $maxQ \leftarrow Q$
33     if $scoreList^{(i)}.keys = scoreList^{(i-1)}.keys$ then
34         go to 35
35 return $comm\_list, maxQ$
```

represented by $\mathcal{V}_j$ are sorted in non-increasing order of their $scoreList^{(i-1)}$ values. For each set $\mathcal{V}_j$ in the current iteration, its best neighbor is calculated by Algorithm 1. It is pertinent to note that best neighbor $\mathcal{N}_{best}(\mathcal{V}_j)$ can be a single node or a snowball formed by a set of nodes, hence it is represented as a set $\mathcal{V}_k$. Next, the current node set $\mathcal{V}_j$ and its best neighbor $\mathcal{V}_k$ are checked for non-redundant node strategy, i.e., if they are already a part of a snowball formed in the current iteration, then their further processing is stopped.

It is important to note that if a node, $\mathcal{V}_k$ or its best neighbor, $\mathcal{V}_j$, have cardinality as one, they are merged without any *overlapping parameter (λ)* consideration. For the generated snowball, $s^{(n)}$, its neighbor set and score values are updated as given by step 24 and 25, respectively. The score for $s^{(n)}$ is calculated by the global clustering coefficient of the subnetwork comprising $s^{(n)}$ and $\mathcal{N}(s^{(n)})$, given by equation (5). However, if the current node set and its best neighbor, both are snowballs then the *overlapping parameter* (λ) is used as shown in step 17. This is an external parameter which decides the minimum percentage overlap that should exist between $\mathcal{N}(s^{(n)})$ and $\mathcal{N}(\mathcal{N}_{best}(s^{(n)}))$ (here $\mathcal{V}_j = s^{(n)}$ and $\mathcal{V}_k = \mathcal{N}_{best}(s^{(n)})$), so as to combine them in the current iteration. The *weight* from step 13 is the ratio of number of common neighbors to minimum number of neighbors of the two sets. If this *weight* is greater than the provided threshold

TABLE II: Statistics of datasets

Dataset	#Nodes	#Edges	#Communities
Karate	34	78	2
Dolphin	62	159	2
Polbooks	105	441	3
Football	115	613	12

of λ, then the nodes are combined to grow the community. Else, the iteration continues with other nodes or snowballs to find their best scoring neighbor.

All the values of scores and neighbor set of the combined snowball are updated for the next iteration. For the node set that remained unchanged, their *scoreList* and *N* are copied from the previous iteration to the current iteration as shown in step 26. The algorithm converges when the community structure remains unchanged for two consecutive iterations and returns the set of communities with the highest modularity value.

V. Experimental Setup and Results

In this section, the experimental setup and evaluation results are presented to establish the efficacy of SbChain approach. A comparative analysis of SbChain with some of existing state-of-the-art methods in terms of the modularity is also presented. Modularity (Q), as defined by equation (6), is used as a measure to decide the final cut of the community in a bottom-up approach being followed by the SbChain. The total number of edges are m, a_{ij} is the value of adjacency matrix entry for nodes v_i and v_j, k_i and k_j represent the degree of nodes v_i and v_j, respectively. *Kronecker delta function* $\delta(c_i, c_j)$ equals 1, if both the nodes v_i and v_j lie in the same community, otherwise it is set to 0. The range of values of Q is $[-1, 1]$. A positive value of Q indicates higher number of observed edges than the expected number of edges considering the random connections.

$$Q = \frac{1}{2m} \sum_{ij} \Big[a_{ij} - \frac{k_i k_j}{2m}\Big] \delta(c_i, c_j) \tag{6}$$

The proposed approach is evaluated over four real-world datasets described in table II and over computer-generated datasets using *Lancichinetti-Fortunato-Radicchi (LFR)* benchmark.

A. Real-World Networks

Table II briefly describes the datasets[2] used in the experiment. The results of modularity identified by the proposed approach are mentioned in table III and compared with Stepping LPA-S and LPA [7].

1) Zachary's karate club: This dataset is described in [30], built around a split of the instructor and administrator of the club, forming two disjoint communities. The results from SbChain approach identifies two communities, with three nodes misrepresented at a allowed threshold of 60% for two snowballs to combine.

TABLE III: Performance evaluation results of SbChain, LPA, and Stepping LPA-S over real-world networks

Dataset	Approach	Modularity (Q)	λ^*
Karate	LPA	0.3573	-
	Stepping LPA-S	0.3715	-
	SbChain	0.3523	0.6
Dolphins	LPA	0.4868	-
	Stepping LPA-S	0.3787	-
	SbChain	0.4347	0.66
Polbooks	LPA	0.5117	-
	Stepping LPA-S	0.4967	-
	SbChain	0.4978	0.63
Football	LPA	0.5897	-
	Stepping LPA-S	0.5754	-
	SbChain	0.4791	0.52

*Overlapping parameter used in SbChain

2) Dolphin social network: An undirected social network of bottlenose dolphins, having links depicting frequent associations between them [31]. There are three communities identified by the proposed approach as compared to two in the original dataset. But the modularity value is strikingly high as compared to Stepping LPA-S. Also, the modularity value is comparable to that of LPA with allowed overlap of 66%.

3) Books about US politics: The nodes represent books about US politics available on Amazon.com. The edges exist between the books that are frequently co-purchased by the same customers [32]. There are three communities in the original network, and the proposed algorithm also identifies three communities with a higher value of modularity than Stepping LPA-S and an allowed overlap of 63%.

4) American college football: A network of football games amongst American colleges during regular season Fall 2000 [33]. The original communities were twelve in number, and SbChain identifies thirteen communities with a moderate value of modularity with 52% of granted overlap.

It can be seen that the devised technique performs fairly well on *Dolphin* and *Polbooks* dataset as compared to Stepping LPA-S, though its performance is not comparable to LPA. However, it should be noted that high modularity values do not necessarily ascertain results closer to ground-truth communities. As an example, in *Karate* dataset, Stepping LPA-S predicts three communities with *Q* as 0.3715, whereas SbChain predicts two communities (which is the ground-truth reality) with *Q* as 0.3523. This is because modularity favors dense groups, which may lead to splitting of networks and formation of a large number of communities.

Figures 2-5 represent the visualizations produced by application of SbChain on four real-world datasets. The *Karate* dataset is shown to produce two communities, *Dolphin* dataset is identified to have three communities. While *Polbooks* and *Football* have three and thirteen communities, respectively, detected by SbChain approach. Also, fig. 6 shows the comparison between the modularity values generated by LPA, Stepping LPA-S and SbChain in real-world datasets.

[2] http://www-personal.umich.edu/~mejn/netdata/

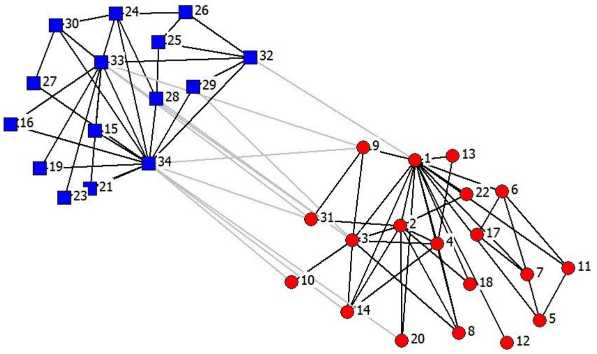


Fig. 2: Visualization of identified communities from Karate dataset using SbChain approach

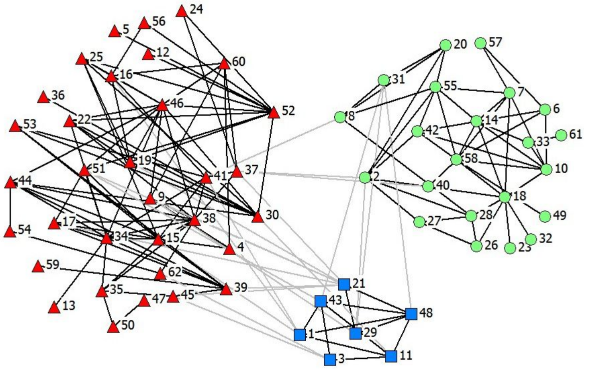


Fig. 3: Visualization of identified communities from Dolphin dataset using SbChain approach

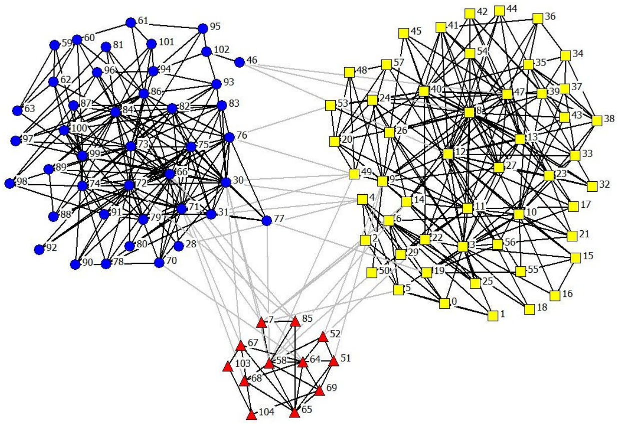


Fig. 4: Visualization of identified communities from Polbooks dataset using SbChain approach

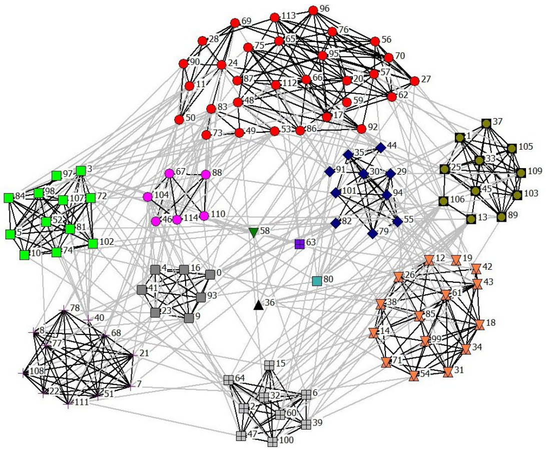


Fig. 5: Visualization of identified communities from Football dataset using SbChain approach

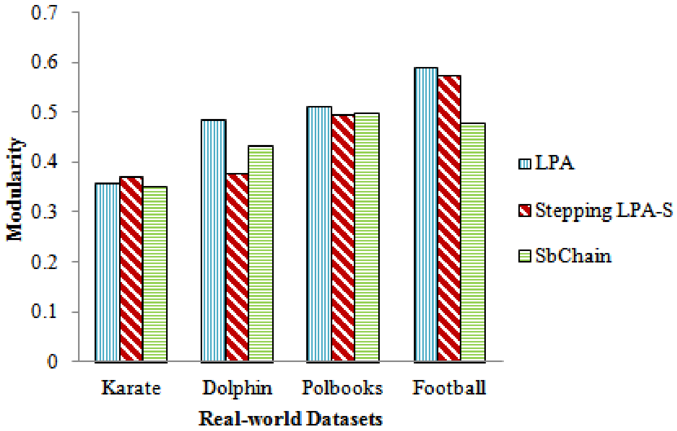


Fig. 6: Visualization of the evaluation results of SbChain, LPA, and Stepping LPA-S over real-world datasets based on modularity values

B. LFR Benchmark Networks

As discussed in [34], Lancichinetti-Fortunato-Radicchi (LFR) benchmark networks are used to generate synthetic datasets. Various parameters used for generation of LFR datasets are mentioned in table IV. μ represents the connections with neighbors in other communities and is set within the range of $[0.1, 0.4]$, varied at a step size of 0.05. The modular structure of a community becomes fuzzy when $\mu > 0.5$, hence we consider values of μ till 0.4. The modularity (Q), identified and actual communities are shown in table V.

VI. Complexity Analysis

This section presents the best-case and worst-case time complexity analysis of SbChain approach. In the best-case, all nodes combine with their best neighbor to form snowballs in each iteration. Hence, each iteration is left with half the number of nodes from the previous iteration. Therefore, the number of iterations is $log_2 n$. The number of nodes that are processed in these $log_2 n$ iterations are $n + \frac{n}{2} + \frac{n}{4} + ... + 1$,

TABLE IV: Parameters used in LFR dataset

Parameter/Representation	Value
Number of nodes/N	1000
Average degree/$\langle k \rangle$	20
Minimum community size/c_{min}	20
Maximum community size/c_{max}	100
Maximum degree/k_{max}	50
Community size distribution exponent/β	1
Degree distribution exponent/γ	2
Mixing parameter/μ	[0.1,0.4]

TABLE V: Performance evaluations results on LFR benchmark networks

μ	Q	#Identified	#Actual	λ
0.1	0.7115	21	21	0.51
0.15	0.7115	21	21	0.51
0.2	0.4963	17	20	0.5
0.25	0.4259	17	18	0.495
0.3	0.7272	20	20	0.5
0.35	0.2835	19	19	0.47
0.4	0.2952	18	19	0.45

forming a geometric progression with sum as n. Hence, the best-case time complexity of `SbChain` is $O(n)$. In the worst-case, only a single pair of nodes merge in each iteration. Therefore, the total number of iterations required to process n nodes is n. And, the total number of nodes processed in n iterations would be $n + n - 1 + ... + 1$, resulting is worst-case time complexity as $O(n^2)$.

VII. Conclusion and Future Work

Inspired from the snowball sampling technique, this paper proposes a simple snowball-chain approach (`SbChain`) for detecting community structures in social networks. The novelty of the proposed approach lies in the requirement of a single overlapping hyperparameter (λ), which is used to merge two snowballs to grow community structures. λ controls the coarseness of the communities to be identified from the social graph, i.e., a higher λ value will split the communities faster and result in small dense communities. Similarly, a low λ value would produce larger communities with low cohesion. This parameter is determined empirically, as the number of nodes and edges in a network play an important role in selecting the value of λ.

The evaluation results of `SbChain` on real-world networks are comparable (and even better in some cases) with two existing state-of-the-art community detection methods. It also works fairly fine over the LFR-benchmark networks. The proposed approach can be extended to work equally well over the directed and weighted networks. Also, the approach can be extended to find communities in multi-attributed graphs that generally model both structural and textual information available in online social media. In line to [18], the `SbChain` can also be extended to detect overlapping communities by allowing a node to join multiple communities.

References

[1] M. E. Newman, "The Structure and Function of Complex Networks," *SIAM Review*, vol. 45, no. 2, pp. 167–256, 2003.

[2] S. Y. Bhat and M. Abulaish, "Analysis and mining of online social networks - emerging trends and challenges," *WIREs Data Mining and Knowledge Discovery*, vol. 3, no. 6, pp. 408–444, 2013.

[3] R. D. Luce and A. D. Perry, "A method of matrix analysis of group structure," *Psychometrika*, vol. 14, no. 2, pp. 95–116, 1949.

[4] A. Lancichinetti and S. Fortunato, "Community Detection Algorithms: A Comparative Analysis," *Physical Review E*, vol. 80, no. 5, p. 056117, 2009.

[5] U. N. Raghavan, Réka, and S. A. Kumara, "Near linear time algorithm to detect community structures in large-scale networks," *Physical Review E*, vol. 76, no. 3, p. 036106, 2007.

[6] S. Y. Bhat and M. Abulaish, "Hoctracker: Tracking the evolution of hierarchical and overlapping communities in dynamic social networks," *IEEE Transactions on Knowledge and Data Engineering*, vol. 27, no. 4, pp. 1019–1032, 2014.

[7] W. Li, C. Huang, M. Wang, and X. Chen, "Stepping community detection algorithm based on label propagation and similarity," *Physica A: Statistical Mechanics and its Applications*, vol. 472, pp. 145–155, 2017.

[8] L. A. Goodman, "Snowball sampling," *The annals of mathematical statistics*, pp. 148–170, 1961.

[9] V. Dongen and S. Marinus, *Graph clustering by flow simulation*. PhD thesis, 2000.

[10] P. Pons and M. Latapy, "Computing communities in large networks using random walks," in *International symposium on computer and information sciences*, pp. 284–293, Springer, 2005.

[11] J. J. Whang, D. F. Gleich, and I. S. Dhillon, "Overlapping community detection using seed set expansion," in *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*, pp. 2099–2108, ACM, 2013.

[12] B. Cai, H. Wang, H. Zheng, and H. Wang, "An improved random walk based clustering algorithm for community detection in complex networks," in *2011 IEEE International Conference on Systems, Man, and Cybernetics*, pp. 2162–2167, IEEE, 2011.

[13] J.-Y. Pan, H.-J. Yang, C. Faloutsos, and P. Duygulu, "Automatic Multimedia Cross-modal Correlation Discovery," in *Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 653–658, ACM, 2004.

[14] T. S. Evans, "Clique graphs and overlapping communities," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2010, no. 12, p. P12037, 2010.

[15] G. Palla, I. Derényi, I. Farkas, and T. Vicsek, "Uncovering the overlapping community structure of complex networks in nature and society," *nature*, vol. 435, no. 7043, p. 814, 2005.

[16] H. Shen, X. Cheng, K. Cai, and M.-B. Hu, "Detect overlapping and hierarchical community structure in networks," *Physica A: Statistical Mechanics and its Applications*, vol. 388, no. 8, pp. 1706 – 1712, 2009.

[17] A. Lancichinetti, S. Fortunato, and J. Kertész, "Detecting the overlapping and hierarchical community structure in complex networks," *New Journal of Physics*, vol. 11, no. 3, p. 033015, 2009.

[18] S. Y. Bhat and M. Abulaish, "Ocminer: A density-based overlapping community detection method for social networks," *Intelligent Data Analysis*, vol. 19, no. 4, pp. 1–31, 2015.

[19] L. Huang, G. Wang, Y. Wang, E. Blanzieri, and C. Su, "Link Clustering with Extended Link Similarity and EQ Evaluation Division," *PLOS ONE*, vol. 8, pp. 1–18, 2013.

[20] T. S. Evans and R. Lambiotte, "Line graphs, link partitions, and overlapping communities," *Physical Review E*, vol. 80, p. 016105, 2009.

[21] M. E. Newman and M. Girvan, "Finding and evaluating community structure in networks," *Physical Review E*, vol. 69, no. 2, p. 026113, 2004.

[22] M. E. Newman, "Fast algorithm for detecting community structure in networks," *Physical Review E*, vol. 69, no. 6, p. 066133, 2004.

[23] S. White and P. Smyth, "A spectral clustering approach to finding communities in graphs," in *Proceedings of the 2005 SIAM international conference on data mining*, pp. 274–285, SIAM, 2005.

[24] Q. Chen, T.-T. Wu, and M. Fang, "Detecting local community structures in complex networks based on local degree central nodes," *Physica A: Statistical Mechanics and its Applications*, vol. 392, no. 3, pp. 529–537, 2013.

[25] Y. Pan, D.-H. Li, J.-G. Liu, and J.-Z. Liang, "Detecting community structure in complex networks via node similarity," *Physica A: Statistical Mechanics and its Applications*, vol. 389, no. 14, pp. 2849–2857, 2010.

[26] M. Tasgin and H. O. Bingol, “Community detection using boundary nodes in complex networks,” *Physica A: Statistical Mechanics and its Applications*, vol. 513, pp. 315–324, 2019.

[27] M. Tasgin and H. O. Bingol, “Community detection using preference networks,” *Physica A: Statistical Mechanics and its Applications*, vol. 495, pp. 126–136, 2018.

[28] P. G. Lind, L. R. da Silva, J. S. A. Jr, and H. J. Herrmann, “Spreading gossip in social networks,” *Physical Review E*, vol. 76, no. 3, p. 036117, 2007.

[29] F. D. Zarandi and M. K. Rafsanjani, “Community detection in complex networks using structural similarity,” *Physica A: Statistical Mechanics and its Applications*, vol. 503, pp. 882–891, 2018.

[30] W. W. Zachary, “An information flow model for conflict and fission in small groups,” *Journal of anthropological research*, vol. 33, no. 4, pp. 452–473, 1977.

[31] D. Lusseau, K. Schneider, O. Boisseau, P. Haase, E. Slooten, and S. Dawson, “The bottlenose dolphin community of Doubtful Sound features a large proportion of long-lasting associations,” *Behavioral Ecology and Sociobiology*, vol. 54, pp. 396–405, 01 2003.

[32] M. E. J. Newman, “Modularity and community structure in networks,” *Proceedings of the National Academy of Sciences*, vol. 103, no. 23, pp. 8577–8582, 2006.

[33] M. Girvan and M. E. J. Newman, “Community structure in social and biological networks,” *Proceedings of the National Academy of Sciences*, vol. 99, no. 12, pp. 7821–7826, 2002.

[34] A. Lancichinetti, S. Fortunato, and F. Radicchi, “Benchmark graphs for testing community detection algorithms,” *Physical Review E*, vol. 78, no. 4, p. 046110, 2008.