Chapter 4

Multiobjective based Community Detection

4.1 Introduction

Mining similar elements of a network is of great importance for its understanding and analysis. This type of mining is referred to as community detection, which aims to group similar elements of a network. In recent years, researchers have been more attracted to identifying communities in dynamic networks. Numerous approaches are explored by the authors to address the problem. Due to the dynamic nature of networks, not only finding communities but also tracking the evolution of those communities is an important challenge in real-world scenarios. Techniques used in static networks are adapted to dynamic networks with certain adjustments. Many studies have used a two-stage approach for it. In this approach, the entire timeline of network evolution is examined in several steps. Communities are identified at each step, and then they are matched for subsequent steps in order to maintain the smoothness of community evolution [28][95][101]. Lin et al. in [95] exploit block models for identifying communities, and then smoothing is ensured by a probabilistic model, which is an example of this approach.

Optimization techniques are also explored by the authors to identify communities. Various heuristic and probabilistic algorithms are used to optimize objective functions, which are mainly focused on community quality. Newman et al. [115] have proposed a measure,

modularity, to evaluate the quality of a community. A large group of researchers has proposed an optimization-based solution by maximizing modularity. Optimizing one objective function proves to be efficient both for time and accuracy, whereas it limits the focus on a single function. A network may have multiple conflicting interests that require optimization considerations. This causes a shift in the attention of researchers to the multi-objective optimization paradigm.

Several algorithms are found in the literature that are formulated around the multi-objective optimization paradigm. Many methods require prior knowledge regarding the number of communities, which makes them parameter-dependent, whereas most multi-objective optimization-based algorithms are independent of initial parameter tuning. A multi-objective optimization problem can be solved by decomposing it into multiple single-objective problems and solving them at the same time [190][49]. Authors have also used some nature-inspired evolutionary algorithms to attain multi-objective optimization.

From the literature, it is evident that the selection of conflicting objective functions is one of the most crucial tasks in multi-objective optimization. The time-varying component of dynamic networks imposes a great challenge to the complexity of the algorithms. The complexity of the algorithm should be low in order to perform efficiently. An optimization problem attempts to find a solution from the available search space. Global optimization at each time stamp increases the complexity of the problem, and only considering local optimization will risk the overall performance. When a new node is introduced in the graph, it is likely to join the community of its neighbor, i.e., it is desirable to have at least one one-hop neighbor belonging to the same community as its own. This reduces the optimization space significantly, as the search will be limited to its immediate neighbors. The decision to assign a node's community label can be influenced by a number of factors. Inspired by our social surroundings, three network properties are considered by the proposed algorithm in the decision process. They are: similarity of node to its neighbor; goodness of community; and attraction of community towards node. A number of metrics are available in the literature for these network properties. An empirical analysis was performed for the selection of objective functions best suited for the problem. The proposed work exploits these three network properties to identify the community of a node in a selected search space without prior knowledge of the number of communities in a network.

To the best of our knowledge, no researchers have applied multi-objective optimization to a temporal network model, nor have they used these metrics as objective functions. For validation, suitability and correctness of the proposed work are compared with state-of-the-art algorithms against different quality and accuracy metrics. The experiments are performed on various real as well as synthetic datasets.

Modularity and Normalized Mutual Index (NMI) are two widely used objective functions in literature for community detection in dynamic networks. Modularity is a metric to assess community quality while NMI is to check temporal cost. Majority of work consider graph as a series of snapshots. Communities in consecutive snapshots are needed to be mapped and their temporal shift is required to be minimised. Whereas if we consider network as a stream of events, this mapping and temporal shift minimization can be removed. As the graph will be an ever-growing entity, operations will be performed on each event and nodes involved in that event only. Algorithm aims to search for community in neighborhood of node which in turn reduces the complexity to great extent. Speed of algorithm is an important factor in dynamic networks because it is continuously changing and the operations are performed on each event. Hence three objective functions are chosen which are inspired from real social behaviour of an entity. Previously, Chapter 3 address the problem using a tree based technique. This chapter exploits optimization framework to identify communities. The principal contribution of work is listed below:

- The proposed work uses three objective functions which are inspired from network properties. These functions are optimized using pareto front in a relatively smaller search space than the entire network.
- The algorithm runs in temporal network model giving it advantage to perform community detection at every event without waiting for a time stamp to complete.
- The performance of the algorithm is evaluated on connected and disconnected datasets. It includes real as well as synthetic datasets to support the analysis of results on different types of networks.
- A comparative analysis of the algorithm is presented against various state-of-the-art algorithms from the literature.

- Assessment of the result shows that the algorithm is able to perform better than most of the state-of-the-art algorithms. Accuracy metrics dominate quality metrics results, which are slightly behind in synthetic datasets for quality metrics.

4.2 Community Life Cycle

Communities are collections of nodes having greater similarities in certain aspects. The community life cycle is a set of four stages along with two operations. Each stage is associated with an operation that is responsible for the current community stage. A community cannot change its stage without any operation. Each community tends to change with respect to events occurring over time. The event is nothing but one of the two operations involved in the cycle, and these operations are responsible for the resultant stage of a community.

Two operations are present in the dynamics of community structure: addition and diffuse operation. There are two possibilities for a node. It is either a new node or an existing node. A new node will join a community by addition. Whereas for an existing node, it will either stay in the present community or join another one via diffuse operation.

The community life cycle consists of four stages:

- *Birth(B):* When a node cannot join any of the existing communities, it creates a new one, which is said to be the birth stage of the community.
- *Expansion(E):* Any addition of non-existing nodes to a community increases its size and is known as the expansion stage.
- *Contraction(Ct):* A change in the node's links with other members may cause it to diffuse to another community where it is more connected. This stage is called the contraction of community.
- *Death(D):* With continuous diffusion operations, a community may lose all its nodes, called the death stage of the community.

The community life cycle (CLC) is a set of ordered pairs and can be written as:

$$CLC = \{< St, Op >: St \in \{B, E, Ct, D\} \& Op \in \{add, diff\}\} \tag{4.1}$$

where,

- B : Birth stage
- E : Expansion stage
- Ct: Contraction stage
- D : Death stage
- Add: Addition operation
- Diff: Diffuse operation

Figure 4.1 represents four stages of the life cycle where edges are annotated with the operation causing the change in stage. Consider a scenario where network is empty with no element i.e. no node or edge. As soon as an element is introduced in the network, it gives *birth* to a new community consisting of respective element(s) of the entity. This introduction is a result of an *addition* operation. Over time more elements evolve in network with *addition* operations, resulting in *expansion* stage and increase in the community size. Community is associated with some criteria that defines the membership of any node to it. As the network grows, that criteria may not be fulfilled by all nodes and they $drift$ to join another community by leaving earlier one in *contraction* stage and *expansion* stage for new one. Multiple $drift$ operation in same community may left it with no node resulting in *death* stage for that community.

4.3 Multi-objective Optimization

An optimization problem aims to find the best solution among all candidate solutions. A multi-objective optimization problem deals with two or more conflicting objectives and attempts to find a trade-off among them. For solving a multi-objective optimization

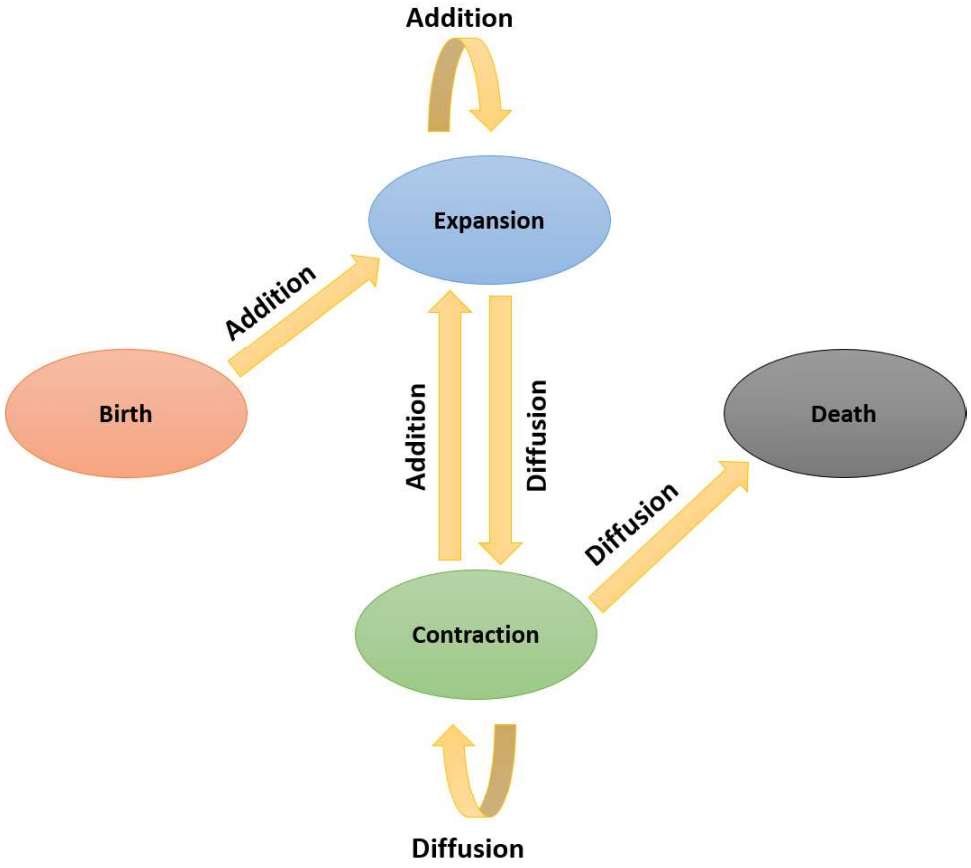

FIGURE 4.1: Community life cycle

problem, it is first formulated as a maximization or minimization problem. The work considers the multi-objective maximization problem. For community set $\mathbf{C_t}$ at time t, it can be expressed as:

$$\max_{\forall C_t^i \in C_t^{can}} (f_1(C_t^i), f_2(C_t^i), ..., f_h(C_t^i)) \tag{4.2}$$

where $C_t^{can} \subseteq \mathbf{C_t}$ is a candidate community set for finding optimal community, f_i is the i^{th} objective function, and h is the total number of objective functions. Candidate community sets reduce the global search space and, hence, facilitate the algorithm's ability to work faster and more efficiently.

Multi-objective optimization theory does not provide the single best solution for a problem. Solution space is a h-dimensional space consisting of a large number of solution points.

Thus, the pareto optimal solution is taken into consideration, which can be defined as follows:

Definition 4.1 (Pareto optimal solution)**.** A solution is said to be pareto optimal if there is no other solution where any one of the objectives can be further refined without degrading others.

At time t a community $C_t^i \in C_t$ is said to dominate $C_t^j \in C_t$ if the following conditions are satisfied:

1. $\{\forall l \in \{1,2,...,h\} | f_l(C_t^i) \geq f_l(C_t^j)\}$ and
2. $\{\exists l \in \{1,2,...,h\} | f_l(C_t^i) > f_l(C_t^j)\}$

A set of non-dominated points, α, is obtained, where for each point in α, there does not exist a point that dominates it.

4.4 Proposed Work

This section specifies the work proposed by the authors of the paper. The theory of pareto optimal solution is applied in the work. Pareto optimality aims to find a non-dominated solution, also known as a pareto solution. Since a problem can have a number of probable solutions, each solution may dominate the other in some aspect. Finding a solution in such a scenario is a tricky task. Hence, the theory of pareto optimality has proven to be a good approach for handling such situations. In pareto optimality, a set of points is selected from the solution space known as non-dominated points (mentioned in Section 4.3) and these points represent the final solution.

Multi-objective optimization helps to find a trade-off between multiple objectives rather than focusing on a single one. A problem is composed of various aspects with different significance. These aspects are either desired to be minimized or maximized, depending on their nature. We need to choose a set of aspects among them and formulate all of them in either a maximizing or minimizing nature. The selection of aspects or objectives requires one more consideration, i.e., they need to conflict with each other. Optimizing

two co-related objectives is not advisable, as it will not contribute to the accuracy of the solution space and the computational overhead will be increased too.

The work optimizes three conflicting objective functions. The selection of them is based on the properties of networks and communities. The three properties considered in the work are as follows:

- In a network, nodes are surrounded by similar nodes, and they are likely to belong to the same community. If a node u joins a community C_i, u tends to possess greater similarity to node $v \in N(u)$ than $w \notin N(u)$ where $N(u)$ is the neighbor set of u.
- The quality of a community is also an important aspect of its selection. A node u tends to prefer C_i over C_j if the former possesses higher quality than the latter.
- Attraction of a community towards a node is the third criteria addressed in the work.

The probable solution space consists of all points that act as candidates to be selected as the final solution. Computational complexity is an important concern in dynamic networks. Though the optimization approach offers a better solution, it also imposes the challenge of greater complexity. To deal with this challenge, we have reduced the global solution space to the proximity space. The decision of a node joining a community will be based on its proximity, i.e., one-hop neighbors. Thus, the solution space will only consist of candidates belonging to the node's proximity.

The formulation of the problem into a multi-objective optimization approach is discussed in detail in the next section. Mathematical formulas for objective functions are also described in the same section.

4.4.1 Mathematical Formulation

The concepts of objective functions and solution space are discussed in this section. The three objective functions used here are inspired by the three important network properties mentioned earlier. Mutual attraction among nodes is an influential factor in community identification. Communities are formed with the basic idea of grouping similar nodes, as they tend to bond stronger. The Adamic-Adar index is used here to compute mutual

attraction among nodes. To assess the quality of a community, we have used a metric proposed by Pizzuti named, community score. Attraction between community and node is measured using the ratio of edges shared between them.

Adamic and Adar have presented a similarity coefficient [4] between two nodes based on the number of edges shared among their common neighbors. AA-similarity between u and v is shown in equation 4.3, where $CN(u,v)$ represent common neighbors of the respective nodes.

$$AA(u,v) = \sum_{i \in CN(u,v)} \frac{1}{\log_2(degree(i))} \tag{4.3}$$

The community score [129] of a community structure $C = \{C_1, C_2, ..., C_k\}$ is the summation of the score of all C_i, can be written as:

$$CS(C) = \sum_{i=1}^{k} Score(C_i) \tag{4.4}$$

Let the volume V_{C_i} of C_i be the sum of adjacency matrix, A, values for all i and j such that $i, j \in A_{ij}$, as written in equation 4.5.

$$V_{C_i} = \sum_{i,j \in C_i} A_{ij} \tag{4.5}$$

For C_i, power mean, $M(C_i)$ of r^{th} order, can be written as:

$$M(C_i) = \frac{\sum_{i \in C_i} (\mu_i)^r}{|C_i|} \tag{4.6}$$

where μ_i is the ratio of the internal degree of i^{th} node to the total number of nodes in C_i. If $K_i^{in}(C_i)$ is the internal degree of node i in C_i, then:

$$\mu_i = \frac{K_i^{in}(C_i)}{|C_i|} \tag{4.7}$$

$Score(C_i)$ is product of power mean and volume given in equation 4.8 as:

$$Score(C_i) = M(C_i) * V_{C_i} \tag{4.8}$$

The work uses *Score* to assess the quality of a community; the higher the value, the better the quality of the community. The third objective function used here measures the attraction of a community towards a node. It is the ratio of intra- and inter-community connections weighted by the node's internal degree to community. For community C_i and node u, $Pull_{C_i \leftarrow u}$ can be written as:

$$Pull_{C_i \leftarrow u} = K_i^{in}(C_i) * \frac{C_i^{intra}}{C_i^{inter}}. \tag{4.9}$$

Equations 4.3, 4.8 and 4.9 are taken into consideration to calculate the three objective functions used in the work. Let $M1$, $M2$ and $M3$ be the three functions, respectively. For every node u, a 3-dimensional solution space will be created where each point is a triplet $< M1, M2, M3 >$. One non-dominated point is being selected using pareto optimal theory, and u will be assigned to the respective community.

4.4.2 Dynamic Settings

The paper presents an algorithm that identifies communities in dynamic networks. Section 2.2 presents a list of datasets used in the paper. The algorithm uses these datasets in a dynamic setting. This setting is applied by casting existing datasets into the mathematical model presented in Section **??**. Each dataset represents a social network graph as a collection of its edges. These graphs are fed as a stream of edges to incorporate dynamic behavior, where each input edge (u, v) at time t represents a change in the graph at that time. Since we are considering the case of an ever-growing graph, we assume that no edge can disappear over time.

The proposed algorithm uses a temporal network model, which means its input will be a series of time-stamped edges. We assume that the network grows with every single edge, and initially, graph G is empty with no nodes or edges. The algorithm 5 inputs a dynamic graph G in the form of a series of edges and identifies its community structure. As G is an evolving graph, every single edge of G will be processed by algorithm 5 at a time, and the loop continues till no new edge is generated in the graph. The algorithm is capable of presenting the existing community structure C of G at time t.

Algorithm 3: Main

Input: Dynamic Graph: $G(V,E)$
Output: $G.communities$: Community Structure

```
1  foreach new edge (u,v) do
2      if (u ∉ G) then
3          u.new ← 1
4          u.community ← ∞
5      if (v ∉ G) then
6          v.new ← 1
7          v.community ← ∞
8      if (u.new ∧ v.new) then
9          Add u, v and edge (u,v) to G
10         G.V ← G.V ∪ {u,v}
11         G.E ← G.E ∪ (u,v)
12         id ← newCommunityID
13         u.community ← id
14         v.community ← id
15         u.new ← 0
16         v.new ← 0
17     else if (!u.new∧!u.new) then
18         if (u.community == v.community) then
19             continue
20         else
21             if (u.degree < v.degree) then
22                 RearrangeCheck(u)
23             else
24                 RearrangeCheck(v)
25     else
26         G.E ← G.E ∪ (u,v)
27         if (u.new == 1) then
28             u.community ← v.community
29             u.new ← 0
30         else
31             v.community ← v.community
32             u.new ← 0
33 Return G.communities
```

4.4.3 Algorithm

Algorithm 5 is the main algorithm of the proposed work, which takes graph G as input. G is a dynamic graph as per the settings mentioned in the section 4.4.2. Nodes in a graph

Algorithm 4: RearrangeCheck

Input: Node: u
Output: Updated Graph: $G(V,E)$

1 $N \leftarrow u.neighbors$
2 $C_{candidate} \leftarrow \{\}$
3 **forall** $(i \in N)$ **do**
4 $\quad C_{candidate} \leftarrow C_{candidate} \cup i.community$
5 $m1 \leftarrow \{\}$
6 **forall** $(j \in N)$ **do**
7 $\quad m1.j \leftarrow similarity(u,j)$ ▷ Refer to equation 4.3
8 $m2 \leftarrow \{\}$
9 **forall** $(k \in C_{candidate})$ **do**
10 $\quad m2.k \leftarrow communityScore(k)$ ▷ Refer to equation 4.8
11 $m3 \leftarrow \{\}$
12 **forall** $(l \in C_{candidate})$ **do**
13 $\quad m3.l \leftarrow communityPull(u,l)$ ▷ Refer to equation 4.9
14 $ss \leftarrow solutionSpace(m1,m2,m3)$
15 $pp \leftarrow paretoPoints(ss)$
16 $final \leftarrow rand(pp)$
17 **if** $(u.community != final)$ **then**
18 $\quad u.community \leftarrow final$
19 $\quad$ update community structure
20 **Return** $G(V,E)$

are associated with a parameter named *new* which is used by the algorithm to check if a node exists in G or not. Initially, *new* parameter of nodes is examined, and it is set to 1 for non-existing nodes (Line 2–7). After that, if the input edge consists of only new nodes, a new community is created for those nodes (lines 8–16 of algorithm 5). Whereas if one node is new and another one is an old node, i.e., it already belongs to some community, then the new one will simply join the community of its neighbor (lines 25–32 of algorithm 5).

For an edge between old nodes of the graph two conditions might be possible, either it is an intra community edge or inter community one. Intra community edges needs no operations as they will improve the strength of community. While Algorithm 4 will be used for every edge which contains nodes of different communities. Node having less degree will call algorithm 4 to check if its community label needed to be updated or not. The complexity of updating process is directly proportional to its degree, hence we choose the node with

lesser degree. The algorithm 4 comprises of three main phases: candidate identification, solution space formation and pareto solution.

Candidate identification Lines 1–4 in the pseudocode represent this section. In this phase, a candidate set of possible community labels is generated by the algorithm. Since a node can join the community of any of its neighbors, the candidate set will consist of all neighbors' community labels.

Solution space formation In this section, objective functions are calculated and the solution space is formulated, which is represented by lines 5–14. The similarity of node u is calculated from all its neighbors and stored in $m1$. The community score of all candidate communities is then calculated in $m2$. Also, the community pull of the candidate community towards u is calculated in $m3$. For each neighbor i of u, a triplet $< m1_i, m2_i, m3_i >$ is formulated, which represents a probable solution point in solution space.

Pareto solution Among all points in solution space, a set of non-dominated or pareto points is calculated. All points that follow the required conditions for a pareto optimal solution will be added to this set; these conditions are already discussed in Section 2.4. If the set consists of more than one point, then the algorithm randomly selects one point as the solution. The community associated with the final pareto point is then matched with the existing community of u and updated if required.

Figure 4.2 visualizes the proposed work, where blocks in the upper rectangle represent steps involved in algorithm 5 and the lower rectangle covers algorithm 4.

The workings of the proposed algorithm are presented with the help of an exemplary graph in figure 4.3 over a period of four time stamps. The evolution of community structure can be visualized through it. Table 4.1 shows operations performed and changes in stages of communities in the graph. Initially graph was empty hence when the edge AB is added to the network, node A creates a new community C_1 and B joins it. C_1 goes from *birth* to *expansion* stage. Three more edges are added at T_0 resulting in *expansion* of C_1. At T_1, two new communities C_2 and C_3 emerges and expanded. We observe $diffuse$ operation at

FIGURE 4.2: The working of proposed algorithm

T_2 and *contraction* stage for C_2 and C_3. At T_3, C_2 goes into *death* stage when $F\ diffuse$ to C_1. Definitions of operations and stages are discussed in Section 4.2.

4.4.4 Complexity Analysis

This section presents time complexity of the proposed work. For algorithm 5, line 1-32 runs $O(E)$. Line 2-7 are node initialization steps which requires $O(1)$ time. New community is formed in line 8-16 which again requires linear time. In line 17-24, algorithm 4 will be called ones. Consider the average degree of nodes be K for given network. Worst case time complexity of for loops 1-4 in algorithm 4 will be of $O(K)$. Best and worst case complexity for finding non-dominated points are $O(Nlogd)$ and $O(N^2)$, where N is size of space and d

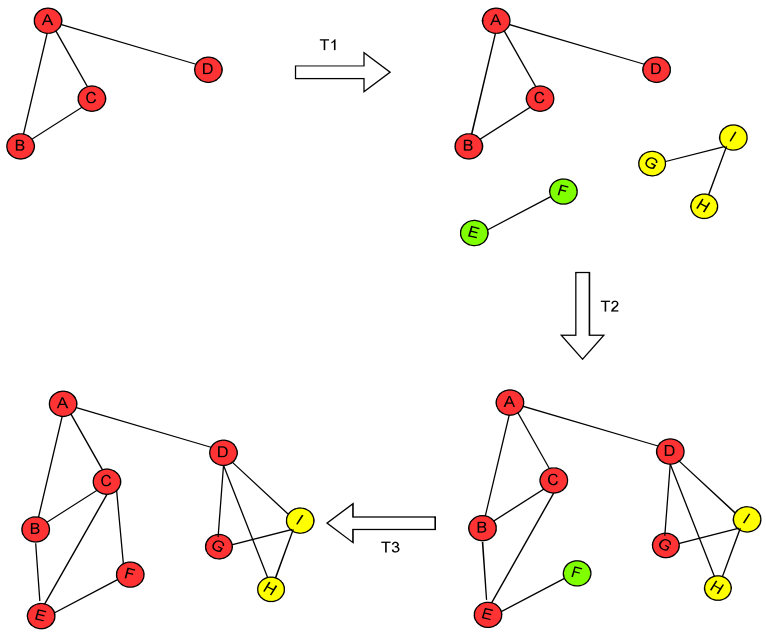

FIGURE 4.3: The evolution of community structure in an example graph at different time stamps

is the number of dimensions. Considering the worst-case scenario, algorithm 4 will runs in $O(K^2)$ time. Hence the overall time complexity of algorithm 5 will be $O(EK^2)$.

4.5 Performance Assessment

4.5.1 State-of-the-art Algorithms

- LICOD - The author in [70] proposed the concept of a leader in communities. The algorithm aims to identify leaders using the concepts of betweenness and degree centrality, and each leader presents a community. After that, all remaining nodes compute their membership for each leader to choose a community.

- RandW - The notion of a random walk being associated with community is exploited by the Steinhaeuser et al. in this work [152]. They use agglomerative clustering along

Time stamp	Input edge	Operation	Stage	Community
T_0	AB	Addition	B(C_1), E(C_1)	$C_1 = \{A,B,C,D\}$
	AC	Addition	E(C_1)	
	BC	Addition	E(C_1)	
	AD	Addition	E(C_1)	
T_1	EF	Addition	B(C_2), E(C_2)	C_1 = {A,B,C,D}, C_2 = {E,F}, C_3 = {G,I,H}
	GI	Addition	B(C_3), E(C_3)	
	HI	Addition	E(C_3)	
T_2	BE	-	-	$C_1 = \{A,B,C,D,E,G\}$, $C_2 = \{F\}$, C_3 = {I, H}
	CE	Diffuse	E(C_1), Ct(C_2)	
	DI	-	-	
	DH	-	-	
	DG	Diffuse	E(C_1), Ct(C_3)	
T_3	CF	Diffuse	E(C_1), D(C_2)	$C_1 = \{A,B,C,D,E,G,F\}$, $C_3 = \{I,H\}$

TABLE 4.1: Various stages of community life cycle in evolution of community structure in the exemplary graph

with the random walk technique to propose a less complex algorithm for identifying communities.

- LeadF - In spite of focusing on external links, Shah et al. [146] utilize the internal connectivity of communities. Nodes are divided into leaders and followers on the basis of their centrality value. Each leader represents a community.

- Tiles - Rossetti et al. in [138] also divide nodes as core and peripheral on the basis of connections among them. For every change in the network, the label propagation algorithm is initiated only among core nodes.

- DANMF - It is a non-negative matrix factorization (NMF)-based approach fused with an auto-encoder model [172]. Hierarchical mapping of network details can be better mimicked via encoder-decoder, as simple NMF models fail to learn complex network structures.

- MNMF - Network embedding aims to capture network properties in a lower-dimensional representation. This downscaling often compromises mesoscopic network structures. Wang et al. in [165] coined a modularity-based NMF approach to deal with it.
- NNSED - Sun et al. in [156] also use the NMF concept for community detection. They use a symmetric encoder-decoder model, which helps to achieve node representation implicitly.
- SymmNMF - NMF's widespread use in network clustering proves its efficiency in capturing communities. Kuang et al. [75] also use it for the same with some improvements. They compute a similarity matrix for all network nodes, which is further used in NMF.

4.5.2 Datasets

This section briefly introduces the datasets used in the work to present the performance of the algorithm. The work uses open-access datasets available on various online platforms. Social networks comprise a wide variety of networks involving friendship, collaboration, communication, and others. Assessment is performed on real as well as synthetic datasets. Nine real and three synthetic datasets are considered here. Since these networks can be connected or disconnected, among the twelve, five are connected and the rest are disconnected. Table 4.2 summarizes the characteristics of these datasets. Three synthetic datasets used in the work are generated on the basis of the LFR (Lancichinetti-Fortunato-Radicchi) benchmark. Edge lists representing datasets are randomized before they are inputted to the algorithm. This randomization of edge order will help to avoid algorithm bias towards specific sequences.

4.5.3 Metrics

The literature on community detection problems incorporates various performance metrics. A set of metrics is selected from them to evaluate the quality as well as the accuracy of the results obtained from the algorithm. The section 2.4 presents a description of the evaluation metrics used in the paper.

Name	No. of nodes	No. of edges	Ground-truth	Network type	Connected	Description
Karate	34	78	✓	Real	✓	Interaction network of Zachary karate club [48]
Football	115	613	✓	Real	✓	Football match of American Division IA
Dolphin	62	159	✓	Real	✓	Interaction network of dolphins [99]
emailEU	1005	25571	✓	Real	✗	Email communication of a research institute of Europe [86]
AstroPh	18772	198110	✗	Real	✗	Collaboration network of arXiv in Astro Physics [86]
CondMat	23133	93497	✗	Real	✗	Collaboration network of arXiv in Condensed Matter [86]
GrQc	5242	14496	✗	Real	✗	Collaboration network of arXiv in General Relativity [86]
HepPh	12008	118521	✗	Real	✗	Collaboration network of arXiv in High Energy Physics [86]
HepTH	9877	25998	✗	Real	✗	Collaboration network of arXiv in High Energy Physics Theory [86]
LFR-1	1000	5166	✓	Synthetic	✓	LFR benchmark [81]
LFR-2	999	5335	✓	Synthetic	✓	LFR benchmark [81]
LFR-3	991	5013	✓	Synthetic	✗	LFR benchmark [81]

TABLE 4.2: Datasets used in performance evaluation

4.6 Results

This segment gives an empirical investigation of the proposed algorithm with well-established state-of-the-art algorithms over numerous connected and disconnected datasets. The result of these empirical investigations is shown and discussed in the different subsections, i.e., results based on quality metrics and accuracy.

4.6.1 Quality

A two-part experimental investigation has been conducted to assess the relationship between quality estimates. The first part compares a quality measure-based comparison of proposed algorithms among nine well-established state-of-the-art algorithms on five connected datasets, while the second part compares four algorithms with proposed algorithms on seven disconnected datasets to evaluate the performance over various quality matrices. A comparison of five state-of-the-art algorithms (BigClam, DANMF, MNMF, NNSED, and SymmNMF) is not performed for disconnected graphs as they only work with connected graphs. The results of this detailed examination have been displayed in Table 4.3 (for connected datasets) and Table 4.4 (for disconnected datasets). From the results, it is clear that our proposed algorithm performs better for real connected datasets as compared to synthetic connected datasets. From the results, it is clear that our proposed algorithm provides better quality in most of the quality assessment metrics for real connected datasets (i.e., in the top three) and disconnected datasets (i.e., modularity and coverage), but not in synthetic datasets. The overall performance of our proposed algorithm for real connected datasets achieves better (i.e., in the top three) in modularity, average isolability, coverage, and external density over the ten state-of-the-art methods, but lags in synthetic connected datasets (i.e., LFR1 and LFR2) in all the quality measures excluding external density. For disconnected datasets, the proposed algorithm performs better than other algorithms in quality measures such as modularity (excluding the CondMat dataset) and coverage (excluding EmailEU and LR3), whereas it lags in quality measures such as average isolability and external density. From this empirical investigation, it is also clear that the performance of the proposed algorithm is superior for the real datasets over the synthetic datasets. Overall, our proposed algorithm attains better quality in real connected datasets (i.e., in the top three) and disconnected datasets (i.e., modularity and coverage) but not in synthetic datasets over the well-established state-of-the-art algorithms. It is also suitable for connected datasets as well as disconnected datasets.

TABLE 4.3: Comparison of quality metrics value of state-of-the-art methods on connected datasets

Dataset	Algorithm	Quality Metrics Results				
		Modularity	Average Isolability	Coverage	External Density	NOC
Karate	LICOD	0.6053	0.1865	0.8462	0.0738	6
	RandW	0.3809	0.5638	0.5641	0.059	2
	LeadF	0.3008	0.3875	0.3846	0.053	6
	Tiles	0.1289	0.1818	0.2949	0.0557	11
	BigClam	0.6226	0.8717	0.8718	0.0174	2
	DANMF	0.0653	0.0516	0.0897	0.0695	14
	MNMF	0.2486	0.3092	0.2949	0.053	10
	NNSED	0.164	0.1436	0.2436	0.0636	7
	SymmNMF	0.0653	0.0516	0.0897	0.0695	14
	Proposed	0.5152	0.3896	0.7308	0.0315	4
Football	LICOD	0.0798	0.0227	0.0881	0.0891	58
	RandW	0.133	0.1	0.1468	0.0464	10
	LeadF	0.687	0.182	0.7586	0.0436	9
	Tiles	0.4772	0.3299	0.5041	0.0246	22
	BigClam	0.5741	0.5885	0.6346	0.0186	12
	DANMF	0.1935	0.1301	0.2137	0.0385	28
	MNMF	0.6382	0.704	0.7047	0.0152	10
	NNSED	0.3474	0.2252	0.3834	0.032	16
	SymmNMF	0.1935	0.1301	0.2137	0.0385	28
	Proposed	0.4522	0.4148	0.4992	0.0249	18
Dolphin	LICOD	0.3015	0.1362	0.3396	0.0643	11
	RandW	0.0778	0.0559	0.0881	0.0451	9
	LeadF	0.3637	0.27	0.4088	0.0321	16
	Tiles	0.2969	0.1147	0.3522	0.0283	32
	BigClam	0.8215	0.9547	0.9623	0.0035	2
	DANMF	0.0993	0.072	0.1132	0.0395	18
	MNMF	0.446	0.4336	0.5283	0.0218	10
	NNSED	0.1955	0.0853	0.239	0.0387	13

Table 4.3 – continued from previous page

Dataset	Algorithm	Quality Metrics Results				
		Modularity	Average Isolability	Coverage	External Density	NOC
	SymmNMF	0.0993	0.072	0.1132	0.0395	18
	Proposed	0.6577	0.4554	0.7736	0.0136	6
LFR1	LICOD	0.9589	0.9664	0.988	0	2
	RandW	0.0393	0.0405	0.0736	0.005	41
	LeadF	0.5889	0.1233	0.5968	0.0048	96
	Tiles	0.2542	0.1304	0.2819	0.0041	284
	BigClam	0.7809	0.5116	0.825	0.0093	2
	DANMF	0.6969	0.6932	0.7079	0.0015	42
	MNMF	0.7107	0.719	0.722	0.0015	10
	NNSED	0.5009	0.1714	0.5088	0.0036	12
	SymmNMF	0.3307	0.2189	0.3363	0.0035	32
	Proposed	0.5221	0.3925	0.531	0.0025	49
LFR2	LICOD	0.8275	0.9695	0.9843	0.0004	3
	RandW	0.0378	0.0413	0.0649	0.0052	46
	LeadF	0.5846	0.1125	0.5928	0.0053	93
	Tiles	0.1973	0.1197	0.2024	0.0046	275
	BigClam	0.7407	0.5016	0.7773	0.0105	2
	DANMF	0.5718	0.5558	0.582	0.0022	47
	MNMF	0.5936	0.5998	0.6042	0.0023	10
	NNSED	0.4181	0.2465	0.4255	0.0042	8
	SymmNMF	0.1602	0.1279	0.1635	0.0045	32
	Proposed	0.4031	0.271	0.4107	0.0033	45

TABLE 4.4: Comparison of quality metrics value of state-of-the-art methods on disconnected datasets

Dataset	Algorithm	Quality Metrics Results				
		Modularity	Average Isolability	Coverage	External Density	NOC
AstroPh	LICOD	0.5311	0.1659	0.5875	0.0002	3420
	RandW	0.5181	0.1636	0.2599	0.0005	1800
	LeadF	0.6341	0.1902	0.3141	0.0004	2953
	Tiles	0.5441	0.1681	0.1174	0	4035
	Proposed	0.7101	0.1354	0.7196	0.0002	382
CondMat	LICOD	0.6752	0.2822	0.1145	0.0001	5027
	RandW	0.7363	0.2903	0.3689	0.0001	2300
	LeadF	0.6803	0.3203	0.3274	0.0001	4030
	Tiles	0.6702	0.2442	0.2287	0	6352
	Proposed	0.7004	0.1876	0.7021	0.0001	808
GrQc	LICOD	0.6503	0.7509	0.7654	0.0002	615
	RandW	1.1125	0.3472	0.5617	0.0003	500
	LeadF	0.9461	0.3956	0.4409	0.0003	1032
	Tiles	1.0124	0.2916	0.1147	0.0003	1359
	Proposed	0.8631	0.2022	0.8787	0.0001	475
HepPh	LICOD	0.5725	0.2343	0.2917	0.0006	1598
	RandW	0.6173	0.203	0.3165	0.0007	1200
	LeadF	0.5278	0.2656	0.267	0.0006	2194
	Tiles	0.8967	0.2234	537.445	0	2796
	Proposed	0.8015	0.1890	0.9141	0.0001	394
HepTH	LICOD	0.6889	0.6584	0.6918	0.0001	570
	RandW	0.8652	0.3181	0.434	0.0002	900
	LeadF	0.7162	0.3519	0.3352	0.0002	2238
	Tiles	0.7206	0.2635	0.1315	0.0015	2607
	Proposed	0.7695	0.2148	0.7722	0.0001	612
EmailEU	LICOD	0.5452	0.0435	0.4351	0.0071	96
	RandW	0.071	0.0409	0.0865	0.0161	38

Table 4.4 – continued from previous page

Dataset	Algorithm	Quality Metrics Results				
		Modularity	Average Isolability	Coverage	External Density	NOC
	LeadF	0.4644	0.0635	0.3544	0.0141	151
	Tiles	0.626	0.0235	0.8668	0.0002	104
	Proposed	0.6915	0.0942	0.8274	0.008	21
	LICOD	0.2249	0.1159	0.2363	0.0041	164
	RandW	0.0351	0.045	0.086	0.0049	44
LFR3	LeadF	0.4667	0.1744	0.4734	0.0042	99
	Tiles	0.4148	0.1869	0.4640	0.0032	235
	Proposed	0.4031	0.271	0.4107	0.0033	45

4.6.2 Accuracy

To assess the accuracy of our proposed algorithms, an empirical comparison of the connected and disconnected datasets has been conducted. For the analysis of connected datasets, an empirical analysis of our proposed algorithm has been carried out along with eight state-of-the-art algorithms on three real connected and two synthetic connected datasets. On the other hand, for disconnected datasets, an empirical analysis of our proposed algorithm along with three state-of-the-art algorithms on two datasets (i.e., one real disconnect and one synthetic disconnect) has been carried out. The result of the accuracy analysis is shown in Table 4.5 (for connected datasets) and Table 4.6 (for disconnected datasets). From the results, it is clear that our proposed algorithm offers better accuracy as compared to the state-of-the-art methods. From the results, it is clear that our proposed algorithm is superior in almost all accuracy measures for real connected datasets such as Karate and Dolphin (excluding the entropy measure) and lags only with the MNMF algorithm for the football dataset. Whereas for synthetic datasets (LFR1 to LFR2), our proposed algorithm archives better accuracy (i.e., in the top three) in almost all accuracy measures for synthetic connected datasets except MNMF (in measures NMI, ARI, and F-score), LICOD (in measures NMI, ARI, and Entropy), and BigClam (in measure Entropy). On the other hand, it is clear that our proposed algorithm also performs

tremendously well in all accuracy measures for real disconnected datasets (except LeadF in the NMI Measure) and synthetic disconnected datasets. Overall, our proposed algorithm attains higher accuracy not only with connected datasets (in both real connected and synthetic connected datasets) but also with disconnected datasets (in both real connected and synthetic connected datasets) over the well-established state-of-the-art algorithms. It is also suitable for connected datasets as well as disconnected datasets.

TABLE 4.5: Comparison of accuracy metrics value of state-of-the-art methods on connected datasets

Dataset	Algorithm	Accuracy Metrics Results				
		NMI	ARI	F-score	Entropy	NOC
Karate	LICOD	0.1604	0.0514	0.0801	0.32	6
	RandW	0.0656	0.0582	0.5448	0.9321	2
	LeadF	0.3707	0.1952	0.4723	0.3751	6
	BigClam	0.5801	0.5722	0.7959	0.4046	2
	DANMF	0.4492	0.176	0.3754	0	14
	MNMF	0.3644	0.0916	0.298	0.2131	10
	NNSED	0.5046	0.3287	0.5446	0.1062	7
	SymmNMF	0.4492	0.176	0.3754	0	14
	Proposed	0.6996	0.742	0.8673	0.1529	4
Football	LICOD	0.5088	0.0924	0.0655	0.2846	58
	RandW	0.2149	0.029	0.171	0.7934	10
	LeadF	0.1881	0.0025	0.1656	0.8686	9
	BigClam	0.26	0.032	0.1958	0.4007	12
	DANMF	0.6842	0.322	0.4516	0.2255	28
	MNMF	0.8923	0.8127	0.8477	0.1358	10
	NNSED	0.616	0.3444	0.4618	0.3834	16
	SymmNMF	0.6842	0.322	0.4516	0.2255	28
	Proposed	0.8178	0.5798	0.6593	0.129	18
Dolphin	LICOD	0.3517	0.1232	0.1534	0.5013	11
	RandW	0.0669	0.0073	0.2645	0.8005	9
	LeadF	0.2452	-0.0021	0.3529	0.4347	16
	BigClam	0.2759	0.2566	0.6287	0.543	2

Table 4.5 – continued from previous page

Dataset	Algorithm	Accuracy Metrics Results				
		NMI	ARI	F-score	Entropy	NOC
	DANMF	0.3334	0.0782	0.2185	0.1106	18
	MNMF	0.2072	0.0616	0.2559	0.4883	10
	NNSED	0.3473	0.1537	0.4053	0.2666	13
	SymmNMF	0.3334	0.0782	0.2185	0.1106	18
	Proposed	0.5417	0.4856	0.6961	0.1333	6
LFR1	LICOD	0.7393	0.3471	0.0504	0.3995	2
	RandW	0.1842	0.0096	0.0601	0.7979	41
	LeadF	0.317	0.0022	0.0601	0.7463	96
	BigClam	0.0142	0.0011	0.0566	0.2516	2
	DANMF	0.3884	0.1728	0.2222	0.441	42
	MNMF	0.7537	0.3936	0.4296	0.3724	10
	NNSED	0.1875	0.0277	0.0843	0.8456	12
	SymmNMF	0.5862	0.3028	0.3472	0.4358	32
	Proposed	0.7292	0.3565	0.3962	0.2875	49
	LICOD	0.5772	0.328	0.125	0.3002	3
	RandW	0.2176	0.0082	0.0623	0.7612	46
	LeadF	0.2867	0.0003	0.0556	0.7675	93
	BigClam	0.0192	0.0011	0.0538	0.2731	2
LFR2	DANMF	0.4281	0.1868	0.234	0.3628	47
	MNMF	0.6766	0.3534	0.3909	0.4354	10
	NNSED	0.1812	0.0351	0.0891	0.8424	8
	SymmNMF	0.4624	0.1752	0.23	0.5382	32
	Proposed	0.6142	0.2505	0.2958	0.4143	45

TABLE 4.6: Comparison of accuracy metrics value of state-of-the-art methods on disconnected datasets

Dataset	Algorithm	Accuracy Metrics Results				
		NMI	ARI	F-score	Entropy	NOC
	LICOD	0.1936	0.0120	0.0909	0.7537	96
EmailEU						

Table 4.6 – continued from previous page

Dataset	Algorithm	Accuracy Metrics Results				
		NMI	ARI	F-score	Entropy	NOC
	RandW	0.1543	0.0023	0.0718	0.7577	38
	LeadF	0.4112	0.0087	0.0938	0.5226	151
	Proposed	0.233	0.0264	0.11	0.7498	21
LFR3	LICOD	0.3410	0.0011	0.0648	0.6802	164
	RandW	0.2149	0.0082	0.0627	0.7638	44
	LeadF	0.4671	0.0105	0.0669	0.5966	99
	Proposed	0.9739	0.9278	0.9324	0	45

4.7 Discussion

The empirical investigation is performed on nine real and three synthetic datasets. The proposed algorithm uses a temporal network model, which means its input will be a series of time-stamped edges. The result of these empirical investigations is based on quality and accuracy metrics.

To assess the relationship between quality estimates, a two-part experimental investigation has been conducted. The first part compares a quality measure-based comparison of proposed algorithm with nine well-established state-of-the-art algorithms on five connected datasets. While the second part compares four algorithms with proposed algorithm on seven disconnected datasets to evaluate the performance over various quality metrics. The search space of the algorithm is not global, as it attempts to find suitable solution around node's neighborhood which does not limit the algorithm to only connected datasets. The quality evaluation results show that our proposed algorithm is suitable not only for connected graphs but also for disconnected graphs while most of the state-of-the-art algorithms (BIGCLAM, DANMF, MNMF, NNSED, and SIMNMF) are not suitable for disconnected graphs because they only work with connected graphs. Overall, our proposed algorithm attains better quality in real connected datasets (i.e., in the top three) and disconnected datasets (i.e., modularity and coverage) but not in synthetic datasets over well-established state-of-the-art algorithms.

To assess the accuracy of our proposed algorithms, an empirical comparison of the connected and disconnected datasets has been conducted. For the analysis of connected datasets, an empirical analysis of our proposed algorithm has been carried out along with eight state-of-the-art algorithms on three real connected and two synthetic connected datasets. On the other hand, for disconnected datasets, an empirical analysis of our proposed algorithm along with three state-of-the-art algorithms on two datasets (i.e., one real disconnected and one synthetic disconnected) has been carried out. The accuracy evaluation results also show that our proposed algorithm is suitable not only for connected graphs but also for disconnected graphs. Overall, our proposed algorithm attains higher accuracy not only with connected datasets (in both real connected and synthetic connected datasets) but also with disconnected datasets (in both real connected and synthetic connected datasets) over the well-established state-of-the-art algorithms.

4.8 Conclusion and Future Work

The paper presents a new algorithm for identifying non-overlapping communities in dynamic networks. The community detection problem is being formulated as a multi-objective optimization problem. The algorithm uses three conflicting objective functions, which are optimized using pareto front optimality. Objective functions represent three important properties that are believed to play crucial roles in community structure. As dynamic networks require fast and simple calculations, the search space for optimization is limited to one-hop neighbors. A node will join only that community for which it has at least one direct neighbor who also belongs to the same community. This fact also supports the use of a one-hop neighbor search space. The performance of the algorithm is supported by a detailed assessment against four quality and four accuracy metrics. Twelve real and synthetic datasets are used, of which five are connected and the rest are disconnected. The result shows that the proposed algorithm performs better for real datasets than synthetic ones. In some datasets, the results of quality metrics are dominated by accuracy metrics. The algorithm performs well in most of the accuracy metrics for real as well as synthetic datasets, whereas the quality metrics results of synthetic datasets are lacking in connected datasets. This detailed empirical analysis presents a visualization of the algorithm's performance over different aspects of datasets. Objective functions used in the work can be explored more, and the search space can be further expanded in search of

better results. In the future, we also intend to improve the performance of synthetic datasets. Experiments on larger datasets are also intended by us for future work.