Republic of Iraq
Ministry of Higher Education and Scientific Research
University of Babylon
College of Information Technology
Department of Software

IDENTIFYING THE INFLUENCER NODES BASED ON TRACKING THE EVOLUTION OF COMMUNITIES IN DYNAMIC SOCIAL NETWORKS

A Dissertation
Submitted to the Council of the College of Information Technology, University of Babylon in Partial Fulfillment of the Requirements for the Degree of Doctorate of Philosophy in Information Technology-Software

Elaf Adil Abbas Essa

Supervised by
Prof. Dr. Huda Naji Nawaf Omrain

2022 A.D. **1443 A.H.**

Supervisor Certification

I certify that the dissertation entitled (**Identifying the Influencer Nodes Based on Tracking the Evolution of Communities in Dynamic Social Networks**) was prepared under my supervision at the department of Software/ College of Information Technology/ University of Babylon as partial fulfillment of the requirements of the degree of Doctor of Philosophy in Information Technology-Software.

Signature:

Supervisor Name: Prof. Dr. Huda Naji Nawaf

Date: / /2022

The Head of the Department Certification

In view of the available recommendations, I forward the dissertation entitled "Identifying the Influencer Nodes Based on Tracking the Evolution of Communities in Dynamic Social Networks" for debate by the examination committee.

Signature:

Assist. Prof. Dr. Ahmed Saleem Abbas

Head of Software Department

Date: / /2022

بسم الله الرحمن الرحيم

﴿أَنزَلَ مِنَ السَّمَاءِ مَاءً فَسَالَتْ أَوْدِيَةٌ بِقَدَرِهَا فَاحْتَمَلَ السَّيْلُ زَبَدًا رَّابِيًا ۚ وَمِمَّا يُوقِدُونَ عَلَيْهِ فِي النَّارِ ابْتِغَاءَ حِلْيَةٍ أَوْ مَتَاعٍ زَبَدٌ مِّثْلُهُ ۚ كَذَٰلِكَ يَضْرِبُ اللَّهُ الْحَقَّ وَالْبَاطِلَ ۚ فَأَمَّا الزَّبَدُ فَيَذْهَبُ جُفَاءً ۖ وَأَمَّا مَا يَنفَعُ النَّاسَ فَيَمْكُثُ فِي الْأَرْضِ ۚ كَذَٰلِكَ يَضْرِبُ اللَّهُ الْأَمْثَالَ﴾

صدق الله العظيم

/ سورة الرعد 17/

Dedications

My father, who is no longer in this world, Adel Abbas, your constant support since my childhood and the emphasis on the importance of education has been the reason I am today.

My mother, who takes me through the valley of darkness with the light of hope and support. Without her prayers and wishes for me, I would not have been able to reach my academic goal.

I also dedicate this work of my husband, Adnan, who encouraged me all along, making me do everything in my power to finish what I started. Adnan has been a constant source of love, support, and strength all these years. I wish to express my love and gratitude to him. None of this would have been possible without the love and patience of my husband.

To my children, Hussien, Ahmed, and Durr for your patience and endurance during the hardships of the road to success.

My beloved brothers and sisters, to all my family, are a symbol of love.

It is also dedicated to all the people in my life who touch my heart.

Elaf Adel Abbbas

Acknowledgement

I am dedicating this dissertation to Allah, my Creator, and my Master. Without divine care, I could not even have contemplated all the work involved in this study. My great teacher and messenger, Mohammed (May Allah bless and grant him), who taught us the purpose of life.

My heartfelt thanks to my supervisor, Prof. Dr. **Huda Naji Nawaf** for her help and support during the period of my study. She is the pillar of my work. Dr. Huda showed me that she was so much more than just my supervisor. Her openness to new ideas, extensive knowledge, and willingness to help me at every turn has made her the best advisor and guide I could hope for .

Finally, but most importantly, I would like to thank all the lecturers in the College of Information Technology who have imparted immense knowledge to me.

Abstract

The social network structure evolves constantly. This is certainly reflected in the formation of communities over time. Naturally, each community goes through a life cycle that is characterized by a series of significant events such as continuing, birth, death, splitting, and merging. Much of the literature in this area focuses on tracking the influential nodes in a dynamic social network as a whole. However, these nodes may have a specific impact on their communities. As such, the influential nodes should be tracked in dynamic communities as well. In fact, all of the events that the community is subject to have an impact on the influencer nodes, at least on one side. On the other hand, events may play a role in the selection of influencer nodes in a specific community and time. Therefore, it is possible to benefit from the influencers who appeared in the community for the previous snapshot, provided that one of the events shows an expansion, contraction, split, or merger in the current snapshot.

This research focuses on tracking the influencer nodes across dynamic communities with two main stages: Dynamic community detection has used an improving Louvain algorithm, which exploits the clique concept in this improvement as a first stage. The second stage has identified the influencers across dynamic communities by using the Non-dominated Sorting Genetic Algorithm II (NSGA-II), adopted in influence maximization to produce the so-called NSGAII-based IM (NSGAII-IM).

Principally, the population is represented by individuals of variable lengths as the seed group, and the diffusion model should be designed named (Weighted Integration Cascade) WIC to model the influence between each pair of nodes so as to formulate a multi-objective function. For the multi-objective function, increasing the coverage size of influence and decreasing the number of seed nodes as far as possible have been set as the conflicting objectives.

In the context of the community, several experimental results have been performed on different data sets. The traditional and improved Louvain algorithms have been applied to record the results and then analyze them. Experimentally, the results prove that execution time has been reduced by around 24.25% of Facebook dataset if it is compared with the traditional algorithm while preserving the quality of partitions at the same time somewhat. Empirical experiments on static social networks show the proposed NSGAII-IM algorithm has achieves better performance by 14% of Digg dataset in terms of both influence spread and faster convergence on superior Pareto Fronts. Implementing the proposed NSGAII-IM algorithm with dynamic networks and taking advantage of community events, on the other hand, improves performance by 11% in the merging event when compared to performance without events. This ratio reflects the highest rate of improvement when compared to the other events.

Table of Contents

Table of Figures

List of Tables

Table of Algorithms

List of Abbreviations

Abbreviation	Meaning
BC	Betweenness Centrality
CC	Closeness Centrality
DC	Degree Centrality
DDS	Degree-Descending Search
EC	Eigenvector Centrality
GA	Genetic Algorithm
GN	Girvan and Newman
HIGHDEG	High Degree
IC	Independent Cascade
ID	Identifier of User
IM	Influence Maximization
INT	Influential Node Tracking
LFR	Lancichinetti–Fortunato–Radicchi benchmark
LPA	Label Propagation Algorithm
LT	Linear Threshold
MOEA	Multi-Objective Evolutionary Algorithm
NSGA-II	Non-dominated Sorting Genetic Algorithm II
NSGAII-IM	Non-dominated Sorting Genetic Algorithm II based Influence Maximization
OSLOM	Order Statistics Local Optimization Method
ScoDA	Streaming Community Detection Algorithm
SDISC	Single Discount
SND	Semantic Nodes Detection
UBI	Upper Bound Interchange Greedy
WC	Weighted Cascade
WIC	Weighted Integration Cascade

List of Symbols

Symbols	Description
K	Size of seed
G^T	Graph in the snapshot T
V^T	Set of vertices in the snapshot T
E^T	Set of edges between vertices in the snapshot T
$C_T = \{C_T^1, C_T^2, \ldots\ldots, C_T^M\}$	Set of communities discovered in G_T.
F_1	First fitness function
δ	Denotes the chosen seed set
F_2	Second fitness function
le	length of the initial set of seeds
P_{ij}	Probability of node i influencing node j
d_i	Degree of node i
Q	Modularity
$\sigma(C_u, C_v)$	Function is 1 if nodes u and v are in the same community , else it has a value of 0
C_q	Cliques of size
μ	Mixing parameter
S^T	Snapshot in time T
$Sim(C_T^i, C_{T+1}^j)$	Similarity between the pair of communities
α and β	Constant values between (0, 1).
Ğ	Sub-graph

Chapter One:

General Introduction

1.1 Introduction

Recently, people have used social networks not only to keep in touch with friends but also to publish their opinions, ideas, even their diaries, which contributes to the spread of information in the network. Researchers are interested in understanding and analyzing the method of information dissemination and the influence of users in these networks (Burbach et al. 2020).

Online social networks are used in a variety of disciplines, including marketing, healthcare, and political science, because of their remarkable ability to reach large audiences in a relatively short period of time (Alasadi and Almamory 2019; Mohapatra et al. 2020). For example, companies strive to select a small group of the most influential users on social networks who can influence the largest number of users. This is an essential aspect of e-marketing and recommendation systems, as well as influencing voter opinions in elections, which can also play a crucial role in political situations and decisions (Y. Shang 2019). In other words, it is possible to influence the outcome of democratic elections by propagating false information or fake news through a network of influencers on social media, thus influencing the voter's decision (Wilder and Vorobeychik 2018).

Influence Maximization (IM) describes the final issue, which can be characterized as a subset of users of size ***K*** that can be determined in advance and is commonly referred to as seed or initial adopters. The initial adopter has the ability to influence his friends, who in turn can influence others, and so on. In this way, the largest number of users is reached (Cui et al. 2018).

Social networks can be divided into static and dynamic social networks. Recently, a growing amount of attention has been paid recently to the identification of influential nodes in dynamic social networks (Hafiene, Karoui, and Ben Romdhane 2020). Since the topology of static networks is defined by friendship ties that evolve slowly over time, while many social networks are defined by ephemeral interactions between entities, adding and removing edges in a network significantly changes the structure of the network over time (Aggarwal, Lin, and Yu 2012). Each user in the network represents a node, and the edge represents the interaction from one user to another. These edges change over time as the interactions between users evolve. Social networks such as (Facebook, Twitter, LinkedIn, Tencent WeChat, and Sina Weibo) are networks that evolve dynamically over time. The process of selecting and attracting the most influential users is of utmost importance (Y. Yang et al. 2018).

For large dynamic networks, it is difficult to find a group of influencers called a seed set. As nodes join and leave, connections form and disappear, and the intensity of these interactions among nodes varies over time. As a result, the topology of the network changes over time (Ding et al. 2020).

Classical diffusion models, like Independent Cascade (IC) and the Linear Threshold (LT) models, as well as more modern seeding techniques (Ju et al. 2020; F. Wang et al. 2018), they are inapplicable due to their reliance on a static network structure. Traditional diffusion models assume a global perspective of the entire social network in order to begin the propagation of influence. Changes in the network necessitate new seeding algorithms. Using a large number of static snapshots is a workaround. Moreover, such a method is inefficient because the heuristics identified in previous time steps are not used, and the solutions have to be explored from scratch (W. Li et al. 2021). This is the focus of the current work since it is possible to use the existing solutions from the previous snapshots instead of starting from scratch.

IM can be thought of as an optimization problem in a wide distributed dynamic space as an alternative approach. The Genetic Algorithm (GA) is recognized as the most important method of evolutionary computation. Evolutionary computation has been widely used to solve a wide range of optimization problems. In the problem of influence maximization, GA supports efficient research in a well-defined problem space that contains a large number of coded proposed solutions, called seed sets (Panizo-LLedot, Bello-Orgaz, and Camacho 2020).

Virtual social networks, particularly large-scale social networks, often exhibit a distinct community structure with highly interconnected nodes. However, it is worth noting that nodes in different communities are often loosely connected. Finding seed nodes in each community can effectively minimize the computational overhead since the communities are usually much smaller than the entire network (Z. Zhang, Li, and Gan 2021).

Dynamic networks are characterized by a variety of modifications that can co-evolve with the behavior of the nodes. In social networks, for example, such changes occur as new members join the network, old members leave it, and existing members establish or terminate relationships over time. These changes can lead to major shifts in the structure of the community, resulting in death, birth, growth, contraction, division, and merger of communities throughout the history of the network (Xu et al. 2020). Detecting and tracking the evolution of new communities in social networks can provide useful insights into the internal structure of the networks and the basic organizing principles, as well as important applications such as analyzing the evolution of research communities within and across academic disciplines and mobile participant networks, in addition to its role in this work (Xu et al. 2020).

Nevertheless, tracking the formation of dynamic communities in complex networks has proven to be a difficult and challenging endeavor (Dakiche et al. 2019), as these networks change rapidly and often unpredictably. For this reason, dynamic networks have mostly been treated as a collection of static graphs created from data collected over long periods of time (Han et al. 2017).

1.2 Research Aim

This study focuses on identifying influential nodes in dynamic social networks from two perspectives: tracking the evolution of communities over serial snapshots as well as detecting influencer nodes in these communities.

The proposed system seeks to use the influencers discovered for communities in previous snapshots, depending on the event the community is going through, to take advantage of them to nominate influencer nodes in the current snapshot instead of starting from scratch.

1.3 Research Problem

In reality, the dynamic behavior of the social network over time affects the identification of influencer nodes to be strongly influenced by the topology of the network. The problem is to track influencer nodes at different intervals so that each period depends on the previous one, which is a challenge. The problem is formulated as follows:

1. Given G_T= (V_T, E_T), where V_T denotes a set of vertices, and E_T denotes a set of edges between vertices in the snapshot T in an undirected graph G_T. Let C_T be a set of communities C_T = {C_T^1, C_T^2, ..., C_T^M} discovered in G_T.
2. In each community the NSGAII-IM algorithm was used to increase the influence spread while decreasing influence cost.
3. Let Pareto Front be the set of the influencer nodes selected from C_{T-1}^M according to the optimization model. Pareto Front be a group of influential individuals selected from C_{T-1}^M after applying an algorithm NSGAII-IM.

4. The selection of the set of influencers for the community C_T^M is related to the community C_{T-1}^M and the event is experienced when the events are Split, Merge, Contraction, Extension. While the set of influencers of the community C_{T-1}^M remains the same as that of C_T^M or is completely different (starts from the beginning) when the event is continuous or birth, respectively.

1.4 Related Works

Over the last few years, the popularity of social networks has skyrocketed. Users of social networks can use them to make new contacts, share their thoughts with like-minded people, and keep in touch with old friends and colleagues; not only that but they are also used in commerce to try to influence the opinions and desires of the users of these networks. The greater a person's influence, the more attractive they are to other companies or individuals looking to promote an idea or sell a product.

This section reviews community detection algorithms, methods of IM, and methods of tracking the influencer nodes in dynamic social networks. Table 1-1 summarizes all the related works that are listed below with additional information.

1.4.1 Community Detection

The earliest community detection algorithm developed by Girvan and Newman (GN) is known as the GN algorithm. Principally the betweenness value is computed for all edges of the graph. Then, the edges with the highest betweenness are removed. These steps are repeated until no edges are left. Then, the level of communities with higher modularity is determined (Newman and Girvan 2004).

The Louvain algorithm is a method developed by (Blondel et al. 2008) from the University of Louvain for extracting communities from large networks. It has gained wide use for the optimization of modularity to be the most effective algorithm in terms of quality and speed. As a matter of fact, it uses hierarchical division and improved greed in two primary stages: (i) the local movement of the vertices and; (ii) grouping several vertices of the original graph into super vertices to reduce the size of a network at each level. Some algorithms have been presented to improve the Louvain method.

The authors (Hu et al. 2016) proposed a hybrid algorithm based on the Louvain algorithm and Label Propagation Algorithm (LPA). This algorithm divided the network into two sub-graph: the first one includes the key nodes with a high degree which predetermined, and the second is the edge nodes which include the other nodes. Afterward, the Louvain algorithm and LPA have been used to detect the communities of the first and second sub-graphs, respectively. That is, the authors intend to improve the running time in the large network using both algorithms.

The work that has been presented by (Traag, Waltman, and van Eck 2019) is to overcome the problem of poorly connected inside communities, thus addressing the weakness at the Louvain algorithm when it's implemented iteratively from their view. Consequently, the authors proposed a new algorithm called the Leiden algorithm. In this algorithm, they validate the real relations inside the community before resuming their procedure. Practically, their experimental results have shown that the running time and the community structure have been improved.

The improved Fast Louvain method is suggested to improve the detection efficiency of large-scale networks. The approach proposed by (J. Zhang et al. 2021) optimizes iterative logic from cyclic to dynamic iteration, which increases convergence speed and separates the network's local tree structure. This makes the algorithm more efficient. The network is split up iteratively, the tree structure is added to the partition results, and the results are optimized to speed up the computation. It has more community aggregation, and the effect of community detection has been made better. The Fast Louvain algorithm is better than the traditional Louvain algorithm at partitioning and speeding up the process.

1.4.2 The Influence Maximization

The influence maximization problem was initially introduced by (Domingos and Richardson 2001), suggesting a probabilistic solution. They proposed a greedy algorithm that considered the maximum effect as an algorithmic problem and a stochastic Markov approach to obtain a probabilistic solution to the effect propagation process. Their greedy algorithm is very inefficient, especially in large social networks with a lot of nodes. From another view, several heuristics have been offered as potential IM time-saving measures.

Centrality measures like Degree Centrality (DC), Betweenness Centrality (BC), Closeness Centrality (CC), and Eigenvector Centrality (EC) have been used to identify influences in the network. The Degree centrality might return suboptimal solutions because it does not take into account the global characteristics of networks but rather depends on the attributes of the nodes (Bao et al. 2017). Since BC and CC are global measures and are based on the structural centralities of the network, these methods have certain computational complexity when used with large-scale networks (Lü et al. 2016). Furthermore, EC is based on iterative refinement centralities, in which the influential node is defined according to the importance and number of adjacent nodes directly related to this node, in addition to the importance of the node itself (Behera et al. 2019).

Evolutionary algorithms are a potential candidate to solve the problem of influence maximization within a social network. For instance, the work presented by (Bucur and Iacca 2016) is considered one of the first works to use the genetic algorithm as a viable tool to solve the influence maximization problem, particularly when there is no prior knowledge about the characteristics of the social network topology. The state-of-the-art evolutionary algorithm Degree-Descending Search (DDS) has been introduced by (Cui et al. 2018) to improve the efficiency of greedy algorithms by evading the time-consuming repeated simulation. The Degree Descending Search strategy generated a seed set that affects the spread according to the degree centrality.

In the study by (Silva et al. 2018), the genetic algorithm has been applied with different strategies to form the initial population from which an individual has been selected. The initial population consists of an individual who has nodes with a high centrality measure, such as Closeness, Betweenness, PageRank, or Degree. In contrast, the rest of the individuals in the population have been randomly selected from the nodes of a network. A linear threshold model has been used as a fitness function to measure the influence propagation of each candidate. The experimental results indicate that the inclusion of an individual consisting of nodes with high central quality yields relatively better results than the case in which no individual is included in the initial population.

Recently, several types of multi-objective optimization algorithms were proposed to solve a number of influence maximization issues within social networks. (Bucur et al. 2017 ; Bucur et al. 2018), proposed a Multi-Objective Evolutionary Algorithm (MOEA). Consisting of conflicting objective functions such that it simultaneously maximizes influence spread and minimizes the size of the individual (number of seeds) by returning the optimal solution. This approach has been shown to be able to overcome two heuristics methods: High Degree (HIGHDEG) and Single Discount (SDISC). The approach has been applied to (ego-Facebook, ca-GrQc, soc-ePinions1) real-world social networks.

The NSGAII algorithm has been suggested by (Mohammadi and Saraee 2018). It is one of the most powerful algorithms for multi-objective optimization to detect different seed sets that have the most influence at different propagation times. The algorithm considers two conflicting objectives, namely increasing the influence coverage ratio and the decreasing propagation time, and thus provides the set of all candidate optimal solutions to support a decision-maker. This straightens out the trade-off between the two conflicting objectives. The results were obtained by applying the NSGAII algorithm on three real-world datasets (WikiVote, NetHEPT, Epinions), which were shown to outperform previous works in terms of influence diffusion.

Finally, the multi-objective optimization method for formulating influence maximization problems has been presented by (Guo, Chen, and Li 2019b). In fact, the spread cost is considered as one of the objectives that have been tackled to improve the evolutionary algorithms. The experimental results indicate the ability to find the optimal set of seeds that have the largest spread of influence with the minimum spread cost and thus are efficient and accurate. However, in their work, seed nodes had selected to provide greater propagation at lower activation cost, and user activation has been adopted as another goal in addition to the volume of spread. The work aimed to select the influencers that have low activation costs and large influence.

1.4.3 Tracking the Influencer Nodes in Dynamic Social Networks

The majority of studies assume that a network is static and neglects the effect of time in information diffusion. Almost all social networks, in actuality, are dynamic and alter over time. When a social network evolves from one time to the next, user relationships may be formed or disbanded. When addressing the influence maximization problem and analyzing influence, taking the temporal aspect would accurately reflect information diffusion in dynamic social networks (Aslay et al. 2018).

In recent works, a time aspect has been taken into account from a dynamic perspective, undoubtedly opening up a new channel. The method by (Hafiene, Karoui, and Ben Romdhane 2020) can be thought of as developing the static algorithm SND (Semantic and structural influential Nodes Detection). SND Update's primary goal is to find the most influential nodes in a dynamic social network. It takes advantage of the network's structural and semantic features. As a result, dynamic social network modeling is a series of snapshot graphs where nodes remain constant, but edges vary over time. SND Update focuses on several steps: First, it identifies communities using the Combo algorithm. Second, it constructs a collection of leading nodes for each snapshot graph. The proposed diffusion model will be applied once the set of initiator leader nodes has been formed. Finally, the proposed diffusion model was applied to identify the influential node.

The Influential Node Tracking (INT) problem(Song et al. 2017) was explored to extend the traditional Influence Maximization (IM) problem under dynamic social networks. The INT problem focuses on tracking a collection of influential nodes that maximize influence. At the same time, the network evolves, whereas the Influence Maximization problem focuses on selecting a group of K nodes to maximize the combined influence under one static network. Upper Bound Interchange Greedy (UBI) and its version, UBI+, are efficient algorithms that take advantage of the network structure's smooth evolution. Instead of creating an initial set from scratch, the algorithms were started from the pre-existing influent seed set, and node replacement was implemented to improve effect coverage. Moreover, the dynamic social networks containing millions of nodes can be scaled using the rapid update method by calculating the marginal gain of the nodes.

According to Li *et al.*, (Li et al. 2021), the Adaptive Agent-Based Evolutionary Model (ABEM) was given as an acceptable research space, and the problem space was optimized by describing the behaviors of each agent. Due to the fact that GA can't promise the best solution, and as the problem space grows, the quality of the solution also gets worse, ABEM helps to alleviate some of the drawbacks of GA by restricting the scope of the search.

Agents (users) engage with neighbors and undertake early evaluations locally before joining the issue space, which greatly minimizes the number of candidate solutions. Then, within a dynamic issue space, ABEM enables GA to design, evaluate, and pick solutions. In this work, the topological structure of the network changes over time.

The agent's candidacy behaviors, i.e. recommending them as influencers and making them members of the influencer pool, were used to create an influencer pool. Members of the influencer pool change as the network evolves, providing an up-to-date problem space for influence maximization. The suggested ABEM can be used in a large-scale scenario because the calculation cost of the influence evaluation is divided into each individual agent.

Table 1-1: Summary of the Related Works

Community Detection				
Reference	**Technique**	**Scope**	**Evaluation**	**Results**
(Hu et al. 2016)	Hybrid Algorithm	Football Collaboration Friendship Amazon	The execution time and the modularity similarity	The execution time compared to the Louvain algorithm respectively decreases by 9.09, 23.07%, 16.35%, and 21.95%. The similarities of the modularity among networks

				are 96%, 99%, 88%, and 95% respectively.	
(Traag, Waltman, and van Eck 2019):	Leiden Algorithm	DBLP Amazon IMDB Live Journal Web of Science	Computational time	Leiden is roughly 2–20 times faster than Louvain in empirical networks. The speed difference is especially noticeable for larger networks.	
(J. Zhang et al. 2021):	The Improved Fast Louvain		The modularity increased or decreased, and the time reduced	The modularity	the time is reduced
		Pokec		decreased 0.04%	45%.
		Facebook		Increased 0.96%	7%
		Google		Increased 0.14%,	84%
		Wikipedia		increased by 0.12%	80%
		Stanford Berkeley		increased by 0.22%	80%
		Stanford		increased by 0.009%	53%
The Influence Maximization					
Reference	**Technique**	**Scope**	**Evaluation**	**Results**	
(Cui et al. 2018)	Evolutionary Algorithms	NetScienc NetGRQC NetHEP	Influence spread and running time	DDSE keeps a close influence spread to CELF algorithm. DDSE is five orders of magnitude faster than CELF algorithm.	
(Silva et al. 2018)	Evolutionary Algorithms	wiki-Vote amazon03	Diffusion of influence of these networks.	The GA improve the results around 50%	
(Bucur et al. 2017):	MOEA	ego-Facebook ca-GrQc	Decrees the number of seed nodes in the set and global influence in	-	

			the graph	
(Mohammadi and Saraee 2018):	NSGAII	WikiVote NetHEPT Epinions	The influence of the seed set and diffusion time.	-
(Guo, Chen, and Li 2019a):	NSGAII	Real networks	Influence spread, cost of spread.	-
Tracking the Influencer Nodes in Dynamic Social Networks				
Reference	**Technique**	**Scope**	**Evaluation**	**Results**
(Hafiene, Karoui, and Romdhane 2019):	Degree Of Nodes	Flixster and NetHept datasets	influence propagation,	UBI is about 30 times faster than IRIE and two times faster than IMM.
(Song et al. 2017):	Greedy	Mobile, HepPh, and HepTh	Running time, Memory Usage,	UBI is about 30 times faster than IRIE and two times faster than IMM. UBI algorithm uses a little more memory than Greedy and IRIE and less memory than IMM
(W. Li et al. 2021):	Evolutionary Algorithms	Facebook, Git Flixster	Influence coverage and running time	-

1.5 The Motivation

Our motivation is divided into two parts: community detection and influencer node tracking in dynamic social networks. The Clique-Louvain algorithm is motivated by the way the traditional Louvain algorithm works to form the community from individual nodes. To reduce the computational cost, the proposed algorithm instead looks for cliques as a starting point for community detection. The second perspective is related to detecting influential nodes in dynamic networks: in our work, we track influential nodes in dynamic communities.

The events that occur in dynamic communities over time, which have not previously played a role in this field, are the main inspiration for this work, where events achieve promising results in the selection of influential nodes. Depending on the nature of the event in a particular community, some influencers of that community in the current interval are selected from the previous interval rather than starting from scratch.

1.6 Contributions

The following are the main contributions of the research:

- ❖ Tracking the influencer nodes with taking into consideration the life-cycle events of the community.
- ❖ The Weighted Integration Cascade (WIC) has been developed as an improved version of the Independent Cascade (IC) diffusion model.
- ❖ The NSGAII-IM algorithm has been proposed to formulate the influence maximization problem using the WIC diffusion model. However, it uses Evolutionary Algorithm (EA) to trade-off between conflicting objective functions and produces a set of optimal solutions.
- ❖ Developing the Louvain algorithm for detecting dynamic communities based on cliques.

1.7 Thesis Outlines

The remainder of this dissertation's chapters are organized as follows:

i. Chapter 2: Presents the general background for the research presented in this dissertation.

ii. Chapter 3: Illustrates the fundamental design phases proposed model of tracking the influencer nodes in dynamic communities. This chapter also covers the datasets utilized in the evaluation task.

iii. Chapter 4: Presents and discusses the findings of the experiments conducted with the datasets.

iv. Chapter 5: The conclusion derived from the findings and discussion in Chapter 4 is presented in this chapter.

Chapter Two:

The Dynamic Social Networks and Influence Maximization

2.1 Introduction

The selection of prominent individuals with whom the dissemination process can begin is one of the most important fields of information diffusion research. The IM problem initially targets a small group of users (referred to as "influencers"), whose influence will spread to a greater number of users in the network. The social network is illustrated using a graph model, where nodes represent individuals and edges reflect interactions between any of these individuals. The IM problem aims to identify a set of influential nodes that will activate the highest number of nodes in the network under a given influence model (Cui et al. 2018).

The use of community structure to address the IM problem is one of the new areas of research being pursued. A network has a community structure if it can be subdivided into a number of nodes, each of which is densely connected to the rest of the network. IM's performance can be improved and computation time reduced by utilizing community structure (Z. Zhang, Li, and Gan 2021).

In this chapter, the fundamental models of information diffusion are first introduced. Next, the dynamic social networks have been analyzed to track the influencers' nodes in the communities. A brief introduction to the evolutionary algorithm in the influence maximization problem is illustrated. It explains the multi-objective evolutionary algorithm used in this study to find the influencer nodes in a dynamic social network and how it worked.

2.2 Information Diffusion

Modeling how information spreads through social networks is very useful in a lot of real-world applications, like finding popular topics, figuring out marketing strategies, identifying opinion leaders, rumors, recommendation systems, and business applications, which require us to be able to observe and regulate the diffusion phenomenon (Kumar et al. 2021).

Diffusion models in social networks have been extensively studied. It is possible to characterize the fundamental theory of the diffusion model as follows: Information propagation begins with the seed node and continues through numerous items, with each activated node attempting to use the activation mechanism to activate the nearby inactive items. Although the processes used in each diffusion model can differ, the activation process is repeated repeatedly until no item scan is activated. This is constant regardless of the model used (Zhou et al. 2021).

Linear Threshold (LT) and Independent Cascade (IC) are the basic diffusion models that have been used to simulate the manner of interactions and information spread among users within the social network. In both models, the node in a social network has either one of two states: active or inactive. The active state of the node is when the node has already been influenced, while the inactive state is when it has not been affected yet. As a matter of fact, if the node's state is inactive and most of its neighboring nodes are active, the chance of it becoming active increases consequently. In general, the contagion proceeds in discrete time steps in both models (Saxena and Kumar 2019).

2.2.1 Linear Threshold Model

This model was proposed by (Y. Li et al. 2018), and it weights each edge between the two nodes. The weight value expresses the level of influence of one node to another. For instance, when $\boldsymbol{W_{ij}}$ is the weight of the edge from $\boldsymbol{i}$ to $\boldsymbol{j}$, the value of $\boldsymbol{W_{ij}}$ shows the level of the effect from i to $\boldsymbol{j}$. Therefore, the sum of the weights of input edges must be equal to or less than one. For the node to be influenced, the sum of the weighted input of the influenced neighbors must exceed a certain threshold (Olivares, Muñoz, and Riquelme 2021). Each node in the network initially has a threshold with a uniformly random value between zero and one. This threshold has to be less than the total summation of weights of input edges required to influence the node. Randomly, a set of active nodes ***S*** has been chosen in time ***T***. At the time ***T+1***, the node will be active when the total weight of the adjacent active nodes is greater than the threshold for this node. When there is no possibility of activation, this process will cease. Several research initiatives have proposed various approaches for specifying the activation threshold to better characterize the diffusion events depending on the desired uses (Alrajebah et al. 2017).

2.2.2 Independent Cascade Model

Concerning the independent cascade, a set of active nodes ***S*** at the time ***T*** is chosen randomly, where each edge in the network holds the probability of diffusion. Whenever the node is active in time ***T+1***, it tries to activate its neighbors with a certain probability propagation ***P***, and for one time only. The probability $\boldsymbol{P_{ij}}$ is the probability of i influencing ***j***, meaning that the edges of the network can diffuse information with the probability ***P***, but not with a probability ***1-P*** (Xin Li et al. 2020); the spreading procedure continues until all nodes have been activated (Huang, Meng, and Shen 2021).

Diffusion in IC model is depicted in Figure 2-1 Node ***B*** is chosen as a seed node and is activated at ***T*** = 0. At ***T*** = 1, only ***A*** is activated by ***B***, and ***B*** no longer can activate any of its neighbors. After that, ***A*** activates ***D*** similarly. The diffusion ends when ***D*** attempts to activate its neighbors and activates ***G*** because there is no active node left to try to activate neighbors (Gursoy 2019).

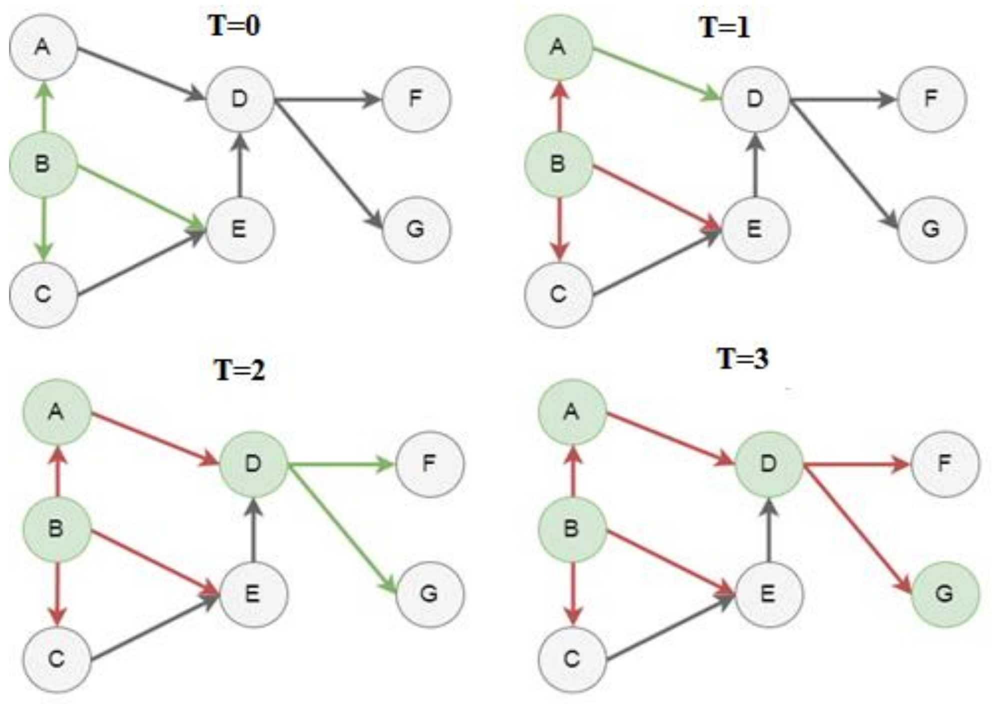


Figure 2-1 IC Model Steps (Gursoy 2019)

Several researchers have tried to develop IC in terms of edge probability in the literature instead of setting the probability with a constant value for all edges, as illustrated in Figure 2-2a. In reality, diffusion probability should be different between any different pair of nodes. Therefore, the Weighted Cascade (WC) model has been suggested to compensate for the (Vardasbi, Faili, and Asadpour 2017; Gokturk and Kaya 2021), according to the degree of the neighbor of an active node. In other words, the same probability value is assigned for all edges of active node j; where the probability of edge $(\boldsymbol{i}, \boldsymbol{j})$ is computed according to Equation (2-1):

$$\boldsymbol{P}_{\boldsymbol{i},\boldsymbol{j}} = \frac{\boldsymbol{i}}{\boldsymbol{in-degree}\,(\boldsymbol{j})}. \qquad (2\text{-}1)$$

Thus, the probability value for any edge between the neighbor j and active node $\boldsymbol{i}$ is the same value as the neighbor, as shown in Figure 2-2b.

The Persuasiveness-weighted cascade model has been proposed to improve the IC models (Hong and Liu 2019). This model depends on the degree of the node in addition to the degree of its neighboring nodes. The propagation probability between the two nodes i and j could be calculated as shown in Equation (2-2):

$$\boldsymbol{P_{i,j}} = \frac{\boldsymbol{d_i}}{\sum_{\boldsymbol{k\epsilon N_j}} \boldsymbol{d_k}} \tag{2-2}$$

Where the $\boldsymbol{d_i}$ represents the degree of node $\boldsymbol{i}$; $\sum_{k\epsilon N_j} d_k$ is the sum of the degree of nodes that are neighbors of node $\boldsymbol{j}$. This model shows that the node with a high degree is of more influence than a node with a lower degree, as shown in Figure 2-2c.

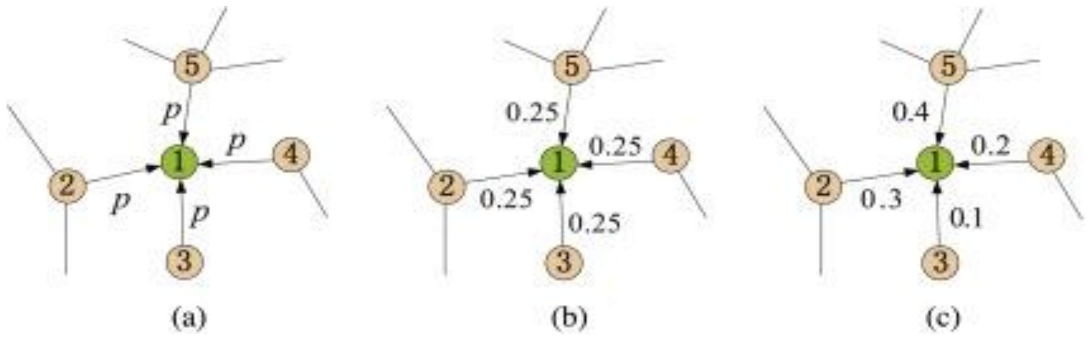


Figure 2-2 Example of Independent Cascade Models: (a) Independent Cascade Model; (b) Weighted Cascade Model; (c) Persuasiveness-Weighted Cascade Model (Hong and Liu 2019)

2.3 Dynamic Social Networks

2.3.1 Community Detection

The community structure of social networks is one of the essential characteristics of these networks because it can give us important information about the behavior of these networks. As a result, according to the research proposed by (Xiao Li et al. 2018), it can be characterized as follows: When the connections between nodes belonging to the same community are dense, the network $\boldsymbol{G}$ has a community structure, whereas nodes belonging to different communities are less connected. Modularity $\boldsymbol{Q}$ is a frequently used metric to describe the robustness of a network community structure (Blondel et al. 2008). It is defined as the fraction of edges, and the Equation (2-3) for determining it is:

$$\boldsymbol{Q} = \frac{1}{2|E|}\sigma(C_u, C_v)\sum_{u,v\in V}\left(e_{uv} - \frac{d_u d_v}{2|E|}\right) \tag{2-3}$$

where $|E|$ signifies the number of edges; $\sigma(C_u, C_v)$ denotes that the value of the function is 1 if nodes u and v are in the same community, else it has a value of 0; d_u, d_v denotes the degree of node u and node v respectively; e_{uv} denotes a direct link between nodes u and v. Furthermore, in a realistic social network, modularity Q is typically between 0.3 and 0.7 (Z. Zhang, Li, and Gan 2021). The social network's community structure grows stronger as modularity value increases.

Palla et al. in (2005) proposed the Clique percolation method for community detection. According to the theory of "clique percolation” edges within the same community can produce a clique, but edges within other communities cannot (Palla et al. 2005). Algorithm 2-1 illustrate how can find clique in graph (Wu and Hao 2015). Using Clique percolation method, community structures may be interpreted effectively and quickly. They can also uncover subgraphs with a high degree of connectivity to estimate the existing communities. As per Clique percolation, a predefined graph G is analyzed, and all cliques of size c_q are detected. After that, a "clique graph" is produced in which nodes represent all the detected cq-cliques, and an edge is inserted between two nodes (c_q-cliques) if they share c_q-1 nodes.

Algorithm 2-1: Clique Algorithm (Wu and Hao 2015)

```
Input:
G: The original graph
Output:
All maximum clique in G
Begin:
1.    Q ← Ø // global value represent clique
2.    CLIQUE(Ø,V)
```

```
3.  Function CLIQUE (C,P)
4.      If ( |C| > |Q| ) then
5.          Q ← C
6.      End_if
7.      if ( |C|+|P| > |Q| ) then
8.          for all p∈ $P$ do
9.              P ← P\ $\{p\}$
10.             C' ← $C \cap \{p\}$
11.             P' ← P ∩ $\{p\}$
12.             CLIQUE(C', P')
13.         End_for
14.     End_if
15. End_Function
16. End_Algorithm
```

The Clique percolation method is used in several algorithms. Two well-known algorithms Order Statistics Local Optimization Method (OSLOM) and Streaming Community Detection Algorithm (SCoDA) could be made better at detecting communities by (Sabour and Moeini 2019) using a preprocessing step called the maximal clique graph before executing the algorithms. When this method was used, the evaluation criteria showed that it was better at detecting communities than other algorithms. The goal of the proposed solution is to add a preprocessing step before feeding the graph to the SCoDA and OSLOM algorithms, illustrated in Figure 2-3.

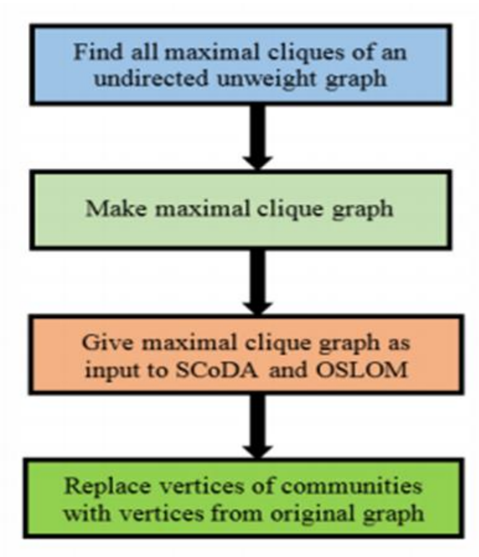

Find all maximal cliques of an undirected unweight graph
Make maximal clique graph
Give maximal clique graph as input to SCoDA and OSLOM
Replace vertices of communities with vertices from original graph

Figure 2-3 The MSCoDA, and MOSLOM Steps (Sabour and Moeini 2019)

In the first step, all maximal cliques of the graph have been found by finding the cliques function. The second step is to make the graph with the maximal cliques, then the edge list of the maximal clique graph is provided as an input to the SCoDA and OSLOM algorithms in the third step. Final communities discovered by this process are displayed as the output of the fourth step.

2.3.1.1 Datasets Utilized By Community Detection Algorithms

The datasets often used in the community detection algorithms can be classified into real graphs and synthetic graphs. There are different kinds of real data that can be represented as a network graph, such as DIGG (Foroozani and Ebrahimi 2021), Dolphin (Bilal and Abdelouahab 2017), Facebook(Weskida and Michalski 2019), and Twitter(Dawar, Bera, and Goyal 2018). The proposed methodology has used different types of real datasets.

Synthetic graphs fast graphic prototypes with varying degrees of community structure are possible thanks to many adjustable parameters that allow fast graphic prototypes (Zhao et al. 2018). The Lancichinetti Fortunato Radicchi (LFR) benchmark creates networks respecting most of the properties of interest of many real-world networks, as shown in Figure 2-4.

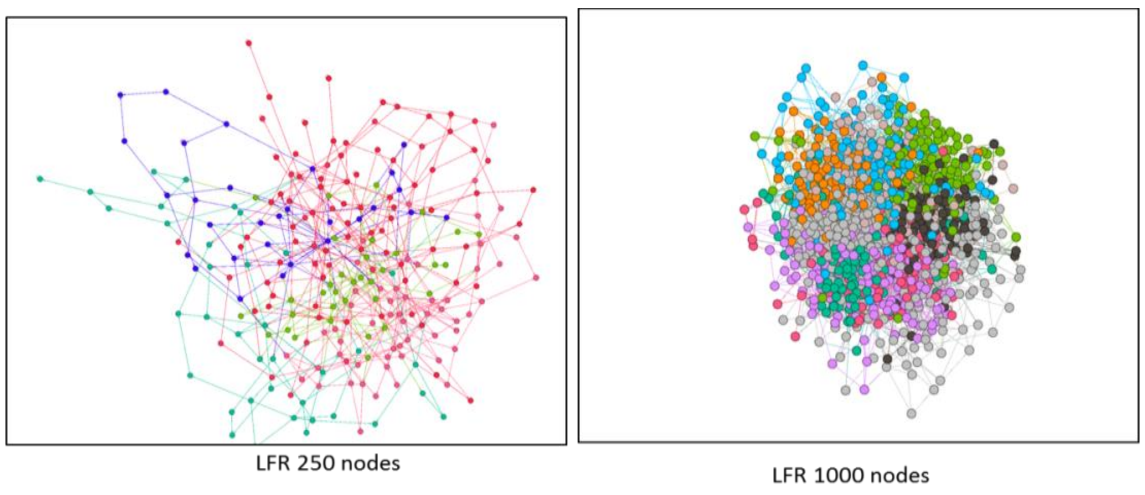


Figure 2-4 LFR with Two Sizes (a) 250 Nodes (b) 1000 Nodes

To create an LFR benchmark, several parameters were needed. In particular, two parameters stand out: the number of nodes $\boldsymbol{n}$ and the mixing parameter $\boldsymbol{\mu}$, the fraction of edges that span across communities, making finding communities harder.

2.3.2 Louvain Algorithm

The Louvain algorithm is one of the most effective algorithms developed to divide the community based on modularity optimization. Initially, the Louvain algorithm considers each vertex as a community (Figure 2-5(a and b)). Mainly, the algorithm involves two steps:

The local movement of the vertices: By randomly choosing one vertex, the modularity $\Delta\boldsymbol{Q}$ gain can be examined to move this vertex to all neighboring communities. Consequently, the vertex would be placed into the neighbor community, which achieves the largest value modularity gain; if the modularity gain is none positive, the vertex stays in its original community. This step will be repeated until all vertices are examined and no longer more modularity gains for any vertex; thus, small communities are formed. The parameter $\Delta\boldsymbol{Q}$ refers to the modularity gain, and it is defined as follows Equation (2-4) (Blondel et al. 2008):

$$\Delta\mathbf{Q} = \left[\frac{\sum_{in} + k_{i,in}}{2m} - \left(\frac{\sum_{tot} + k_i}{2m}\right)^2\right] - \left[\frac{\sum_{in}}{2m} - \left(\frac{\sum_{tot}}{2m}\right)^2 - \left(\frac{\sum k_i}{2m}\right)^2\right] \qquad (2\text{-}4)$$

Where$\sum_{in}$ the summation of the weight of edges inside $\boldsymbol{C}$, $\boldsymbol{k_{i,in}}$ is the summation of the weights of the edges from vertex $\boldsymbol{i}$ and to vertexes in community $\boldsymbol{C}$, $\sum_{\text{tot}}$ the summation of the weight of edges incident to vertices in the community, $\boldsymbol{k_i}$ is the sum of the weights of the edges incident to vertex $\boldsymbol{i}$ and m is the sum of the weights of all the edges in the network . Figure 2-5c states the local movement step.

Grouping several vertices into supervertices: The vertices belonging to the same community are aggregated into a single vertex named *supervertices*, as illustrated in Figure 2-5d.

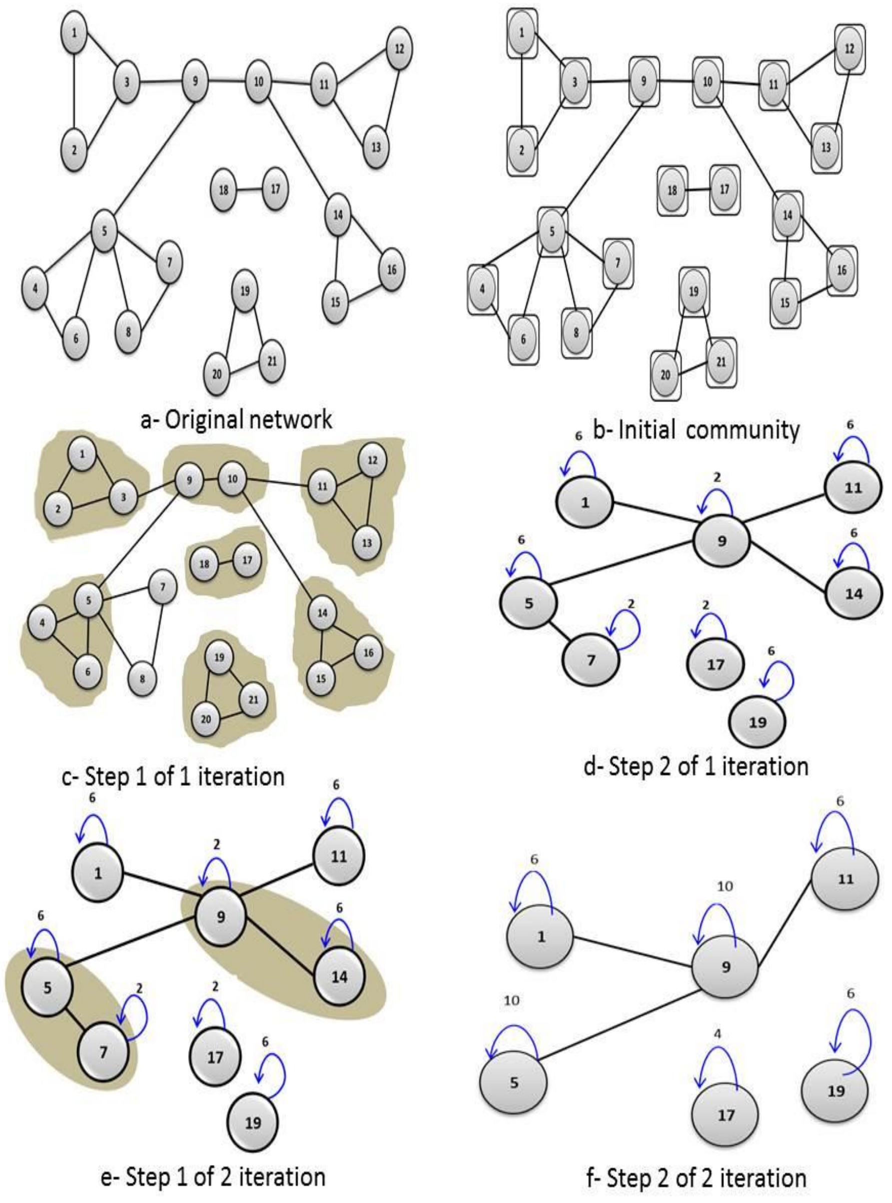


Figure 2-5 Louvain Algorithm Steps (Ozaki, Tezuka, and Inaba 2016)

It was reconstructing meta-graph by computing the weight of edges among all *supervertices*. When there are vertices in different partitions in the graph, and there is at least one edge between them, the *supervertices* must connect with each other. At the lower level of the graph, the weight is calculated, representing the sum of all edges connected between the vertices in the different partitions. At the same time, the self-loop included in *supervertices* with its weight describes the sum of the edges connecting the linked vertices in the same partition of the graph at the lower level.

These two steps of the Louvain algorithm are repeated until the modularity value is no longer improved and hierarchical communities are generated, as shown in Figure 2-5(e and f). The example is illustrated in Figure 2-5 explains the Louvain algorithm (Ozaki, Tezuka, and Inaba 2016) for only two iterations.

2.3.3 Tracking the Evolution of Communities

In a dynamic social network, a community is not only a group of densely interlinked nodes, but its members keep their close interconnection over a prospective time (Javadi, Gharani, and Khadivi 2018). Dynamic graph clustering is another name for this problem. One of the strategies used to discover communities in temporal networks consists of two stages that slice the entire network into a series of snapshots. Then a static community detection approach has been applied to each snapshot separately and compare the clustering of subsequent snapshots to capture the evolution of the communities called independent community detection (Dakiche et al. 2019).

During the span of a community's life cycle, it is confronted with a series of key events that emerge from a shift in the way its members interact with one another from one point in time to another, which may result in a modification in the community's structure. Keeping track of these occurrences helps gain a better understanding of how communities emerge in social networks. For practicality, $\boldsymbol{C_T}$ and $\boldsymbol{C_{T+1}}$ will be referred to as the set of communities in snapshots $\boldsymbol{S_T}$ and $\boldsymbol{S_{T+1}}$, taken at two successive time intervals, respectively.

To pick up and identify the evolutionary events of dynamic networks, actual matching between communities is performed from different snapshots. The Jaccard coefficient (Elhishi, Abu-Elkheir, and Abou Elfetouh 2019; Ziwei He 2018) was used for binary sets of the widely adopted. Given a community $\boldsymbol{C_T^i}$ in snapshot T and $\boldsymbol{C_{T+1}^j}$ is the community in the next snapshot $\boldsymbol{T+1}$, the similarity between the pair of communities has been calculated as follows compute with Equation (2-5):

$$\boldsymbol{Sim}\left(\boldsymbol{C_T^i}, \boldsymbol{C_{T+1}^j}\right) = \frac{\boldsymbol{C_T^i} \cap \boldsymbol{C_{T+1}^j}}{\boldsymbol{C_T^i} \cup \boldsymbol{C_{T+1}^j}} \tag{2-5}$$

$\boldsymbol{C_T^i} \cap \boldsymbol{C_{T+1}^j}$ represents the number of nodes shared between $\boldsymbol{C_T^i}$ and $\boldsymbol{C_{T+1}^j}$, $\boldsymbol{C_T^i} \cup \boldsymbol{C_{T+1}^j}$ represents the number of every node in the two communities together. Whether the similarity exceeds the matching threshold $\boldsymbol{\theta \epsilon [0, 1]}$, $\boldsymbol{Sim}\left(\boldsymbol{C_T^i}, \boldsymbol{C_{T+1}^j}\right) > \boldsymbol{\theta}$, it may be said that the pair of communities are similar (K. Yang et al. 2017).

There are numerous techniques (Yin et al. 2017; Alrajebah et al. 2017) to define possible events in the development of a community. A brief overview of the most prevalent events is provided below, as illustrated in Figure 2-6.

- **Birth**: The creation of a new community occurs when the community that was not observed in the previous snapshot T appears in the next snapshot T+1 (Elhishi, Abu-Elkheir, and Abou Elfetouh 2019).
- **Death**: Dissolution occurs(opposite Birth) when a community is not in the following snapshots, i.e., its members have either disappeared or stopped communicating with each other and scattered among the rest of the communities (Z. Wang et al. 2018).
- **Shrinking**: The community shrinks when some nodes leave it, making it smaller than in the previous snapshot. A community may

contract slightly, that is, by some decade, or greatly lose a large part of its participants (e.g., a reduction of > 10%) (Ziwei He 2018).

- **Growing**: The community grows when some new members join it from other communities, making it bigger than it was in the previous shot. A community may develop such that it doubles in size, triples, or slightly grows (e.g., a growth of > 10%) (Ziwei He 2018).
- **Splitting**: In addition, a community splitting may occur when a community is dividing into more than one community at the following snapshot. In other words, some communities from the following snapshot consist of members of one community from the previous snapshot (Dakiche et al. 2019).
- **Merging**: A community has been formed when two or more communities merge from the previous snapshot into one community in the following snapshot (Dakiche et al. 2019).
- **Continuing**: A community continues its being when two communities in the consecutive snapshots are identical or when a few nodes differ in that community (Xu et al. 2020).

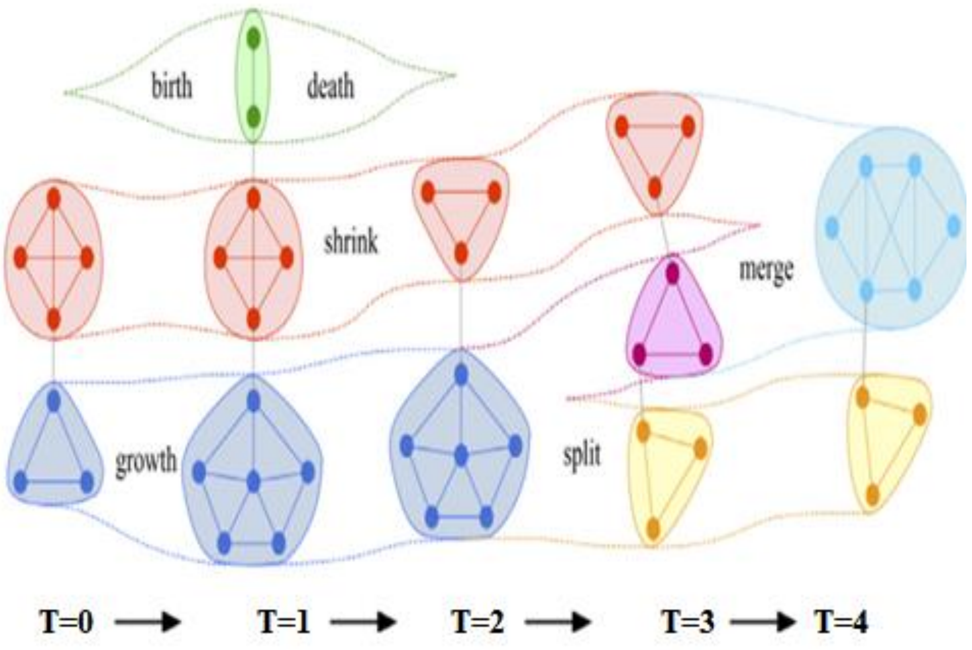


Figure 2-6 Events of Dynamic Communities (J. Shang et al. 2016)

2.4 Influence Maximization

The goal of IM is to select a small set of nodes that can impact as many other nodes in a network as feasible. In this sense, "***i*** influences ***j***" means that ***i*** pass on an opinion or information to ***j*** (potentially indirectly via other nodes). Many algorithms are out there that try to solve this NP-hard combinatorial optimization problem by picking (hopefully) important seed nodes (Can and Alatas 2019).

2.4.1 The Heuristics Algorithms

In recent years, various approximation methods and heuristics have been introduced to achieve maximal influence propagation in the network. Rather than computing the marginal gain of the nodes in each iteration, these heuristic algorithms iteratively choose nodes depending on a specified heuristic, such as degree, closeness. Their disadvantage is that they have a low level of accuracy (Alshahrani et al. 2020). The following has been focused on five different approximation-based algorithms to validate the proposed method in this work.

i. **Degree:** In context, a clear heuristic selects the ***K*** nodes with the greatest degrees, which has a high number of neighbors (Agarwal and Mehta 2018).

ii. **Degree Discount HIGHDEG:** The degree centrality score of the seeds is used to choose the seeds, and the edge that links with the following selected seed is discounted from the node's degree computation (Alshahrani et al. 2020).

iii. **Single Discount SDISC:** Considering the IC model, this method approximately equals the influence spread of the traditional greedy algorithm while improving the degree heuristic. The degree of its one-hop neighbors has been reduced when choosing a max degree node as a seed, which reduces the influence spread of its neighbors. On the other hand, the Degree Discount method estimates discounts on degrees in greater depth. The Single Discount method outperforms the Degree Discount algorithm (Liu et al. 2021).

iv. **Eigenvector Centrality EC:** A node's centrality can be enhanced by its association with other central entities. In some cases, nodes' centrality is not just determined by the number of nearby nodes they have but also by their centrality value. Identifying $\boldsymbol{K}$ of nodes as seeds, with nodes with a large eigenvector centrality value (Dey, Chaterjee, and Roy 2019).

v. **Closeness Centrality CC:** The total of shortest path distances between a node and all other nodes in a network determines its centrality (Salavati, Abdollahpouri, and Manbari 2019), as well as the following definition:

$$\boldsymbol{CC(i)} = \frac{\mathbf{1}}{\sum_{\boldsymbol{i \neq j}} \boldsymbol{d(i,j)}} \qquad (2\text{-}6)$$

Where $\boldsymbol{d(i,j)}$ is the length of the shortest path from node $\boldsymbol{i}$ to node $\boldsymbol{j}$, the set of seed nodes with the greatest closeness centrality are chosen as seeds by a clear heuristic.

2.4.2 Evolutionary Algorithms

Evolutionary algorithms are computer algorithms that tackle complicated problems by mimicking Darwin's method (Maier et al. 2019). Individuals in the population compete against one another in Darwin's method. Individuals are intended to evolve and find better solutions than those previously discovered until the objective is met or some stop criterion is met. As a result, each individual in the population represents a potential solution (set of seeds) to the problem and must be evaluated by a fitness function, then determining the optimal solution that the individual defines as in population.

2.4.2.1 Genetic Algorithm

According to (Krömer and Nowaková 2017), a Genetic Algorithm (GA) is an Evolutionary Algorithm. Even though there are various varieties of a GA, there are four properties shared by all of them: individual populations, fitness function, crossover, and mutation.

In order to solve the IM problem, a genetic algorithm has been proposed in which the seed set is encoded as an individual (i.e., each gene represents a seed) and where the fitness function corresponds to the diffusion model (K. Zhang, Du, and Feldman 2017). Initially, a random selection of individuals is used to create the group. Following that, a binary tournament will be used to determine which individuals will be picked for crossover, and the fitness of each individual will be calculated in order to identify which individual could be a candidate for the next generation of the population (Tsai, Yang, and Chiang 2016).

i. Individuals Populations

One way to think about it is that an individual population collects potential solutions. Based on their numerical sequence, genes are assigned to each candidate solution (individuals or seeds) (Rodrigues et al. 2018).

ii. Fitness Function

The evaluation function value is referred to as (fitness). It is also employed as a selection process to pick individuals from a population for reproduction. Using this function can determine how closely a potential solution matches the problem at hand. In addition, a population's best members can be identified by this method (Krömer and Nowaková 2017). The fitness function can be one of the diffusion models. The fitness function evaluates each individual, i.e., the influence of each individual is calculated according to the diffusion model used (Tsai, Yang, and Chiang 2016).

iii. Crossover Operators

Combining two individuals to create a new individual by using the crossover genetic operator is possible. The concept is that the new individual may be better than the parent individual if it acquires the best features from each of them (Journal, Singh, and Malik 2019). The simple type of crossover operator is one point crossover (Mirjalili 2019), as shown in Figure 2-7.

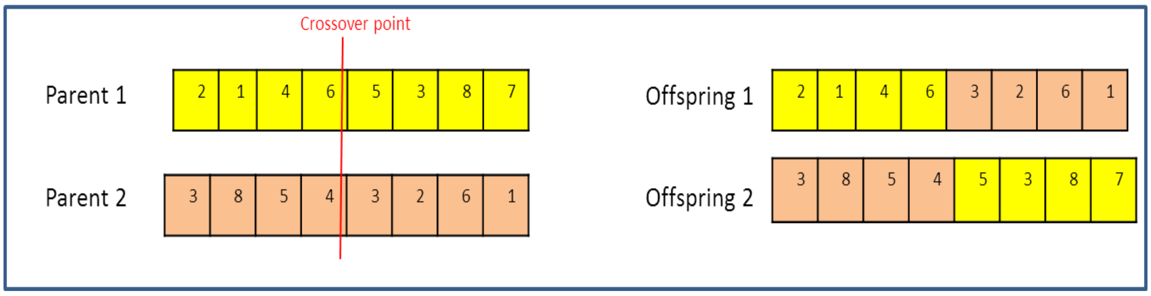


Figure 2-7 One Point Crossover Operator (Abo-Elnaga and Nasr 2020)

iv. Mutation Operators

A mutation happens when one or more of a person's genes are altered. This gene must be picked at random, and the probability of a change must be minimal. The mutation is responsible for the diversification of the population since it arbitrarily alters the individual without considering previous generations' knowledge (Lotf, Azgomi, and Dishabi 2022). This method breaks free from local optimal situations (Katoch, Chauhan, and Kumar 2021).

2.4.2.2 NSGAII Algorithm

Usually, companies choose the influencing people within social networks and pay them money to encourage them to publish targeted information in that social network. Several methods have been investigated to find seed nodes in the influence maximization field in order to achieve the highest prevalence (Abbasi and Fazl-Ersi 2022).

The NSGAII algorithm is the most powerful multi-objective evolutionary algorithm (Zhao et al. 2018). In this algorithm, a random parent population is initially created, after which a quick non-dominated sorting approach is employed to get the Pareto Front which has the sets of solutions that are non-dominant to each other (Peng 2020). Non-domination Pareto front can be classified into multi-levels, where first level is considered the best level, followed by second level, and so on. Thus, each individual in the population has been assigned a fitness respecting its level, as illustrated in Figure 2-8.

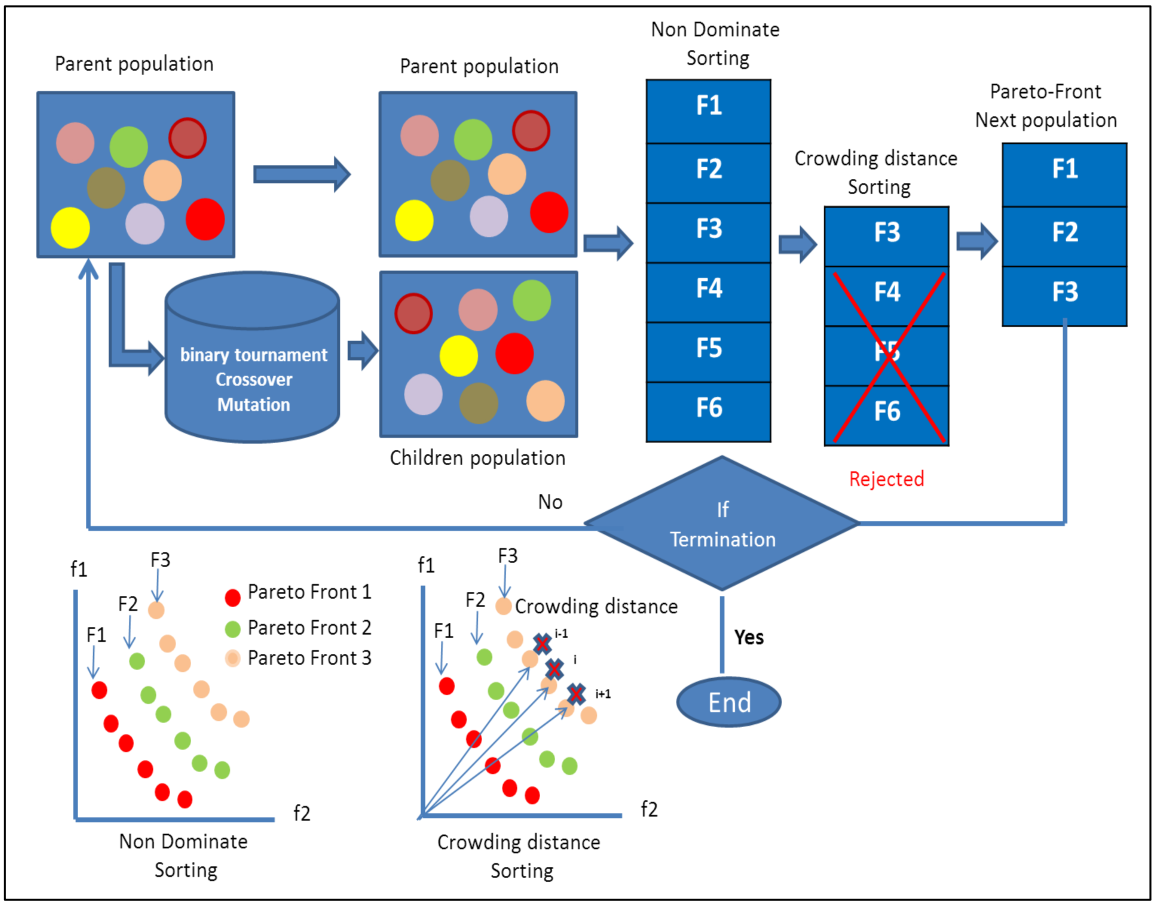


Figure 2-8 Non-Dominate Sorting Genetic Algorithm

Essentially, the NSGAII algorithm uses a binary tournament selection then creates an offspring population by using crossovers at one point and mutation operators. At first, the population combines the offspring and parent populations to create a mating pool. It is followed by sorting the population-based on its non-domination (Deb et al. 2002). Obviously, the procedure is somewhat different after the initial generation, so that the next population is opted from individuals in lower Pareto fronts to create a new generation every time. Suppose the number of individuals on the first level front is less than the size of the population. In that case, the individuals from the next level are added and continue until they reach the population's size, which is explained in Figure 2-8.

The crowded-comparison approach can be used whenever the number of individuals on the next level front is greater than that of need. With respect to the previous approach, it computes the crowding distance between an individual and its neighbors. The crowded-comparison approach can be added as a new parameter to select individuals in the next generation. Indeed, the selection process works by using fitness (ranking the level of Pareto fronts) and the crowding distance of each individual. If both individuals belong to the same level of Pareto fronts, the priority is for the individual that is situated in a lesser crowded region. The selection process can be directed to the optimal Pareto front according to Algorithm 2-2 (Sun, Deng, and Li 2020; X. Wang, Chen, and Xu 2022).

Algorithm 2-2: Crowding-Distance (Sun, Deng, and Li 2020)

```
Input:
P: non-dominated solution set
S: number of solutions

Output:
P[i]_dis: crowding-distance of each solution

Begin:
1.   for each i ∈ S do
2.       set P[i]_dis ← 0.0
3.   End_for
4.   for each objective n do
5.       P←sort(P, n )
6.       P[1]_dis←P[S]_dis← ∞
7.       for i = 2 to S − 1 do
8.           P[i]_dis← P[i]_dis+( P[i+1].n- P[i-1].n)($f_n^{max} - f_n^{min}$)
9.       End_for
```

```
10.    End_for
11.    return P[i]_dis
12     End_ALgorithm
```

For each individual $\boldsymbol{i}$, the crowding distance is represented by ***P[i]***, whereas the value of objective function $\boldsymbol{n}$ is denoted by ***P[i].n***. Both the maximum and minimum objective function values are represented by the notation $\boldsymbol{f_n^{max}}$ and $\boldsymbol{f_n^{min}}$. In the non-dominated solution set $\boldsymbol{P}$.

Chapter Three:

Tracking Influential Nodes in Dynamic Communities

3.1 Introduction

This chapter discusses the significant stages of the proposed model of tracking influencer nodes. Section 3.2 introduces the suggested model architecture in a framework, followed by Section 3.3 gets into the data sets. In section 3.4, the preprocessing task is detailed. Section 3.5 focuses on dynamic communities' detection. Finally, the NSGAII-IM algorithm based on tracking the evolution of communities is described in section 3.6.

3.2 The Proposed System

The scheme in Figure 3 1 clarified the proposed system. It states the preprocessing stage and two key design stages, Tracking Dynamic Community Detection and Tracking Influential Nodes.

Firstly, social relationships are required in addition to the past social activities of the users. The available data is subject to a few prepossessing tasks to extract the necessary information to construct the network of relationships and build a social graph. The network is then divided into a series of snapshots, each snapshot is represented by a graph, and communities are discovered for each snapshot, followed by tracking of community events through these snapshots.

For each community, the influencing nodes will be identified using the Non-dominated Sorting Genetic Algorithm II (NSGA-II), which is adopted in effect maximization to produce the so-called NSGAII-based IM algorithm (NSGAII-IM). The influencers identified for the communities in the previous snapshots can be tapped depending on what event the community is going through and what it represents explore Pareto Front in the schema.

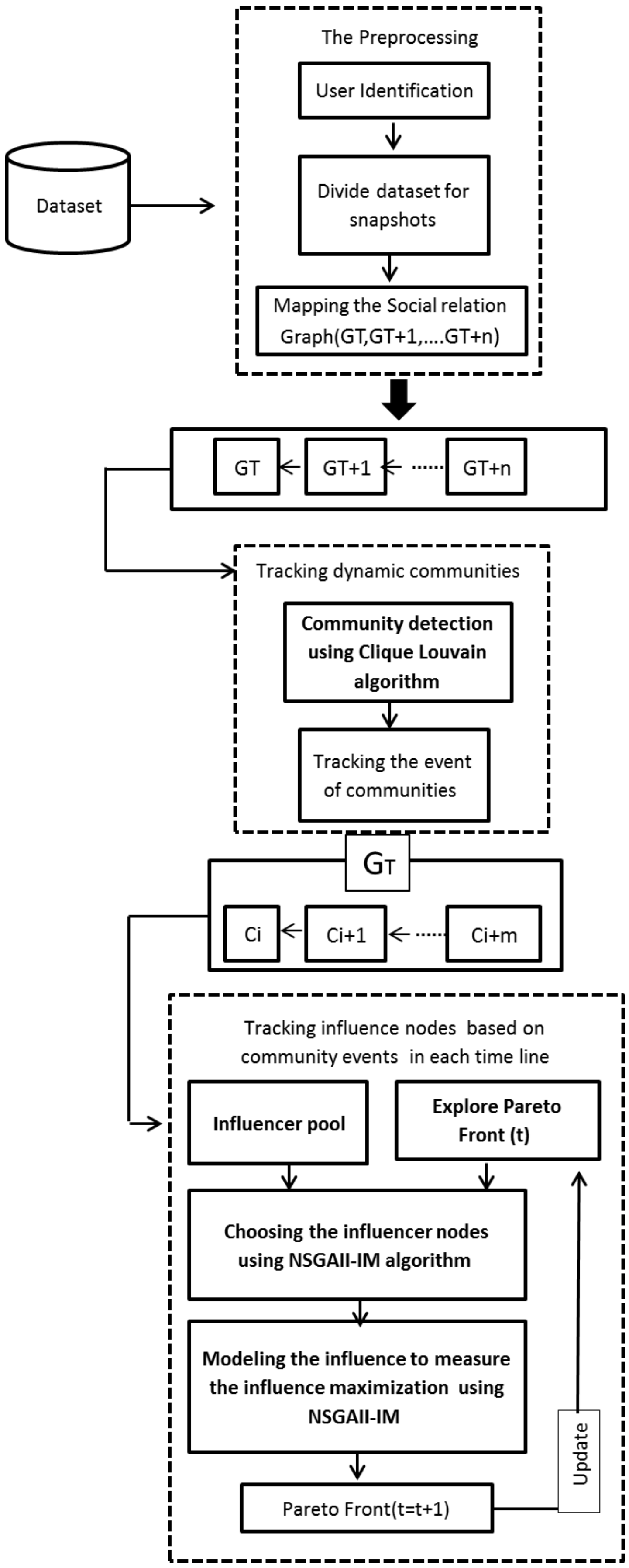

The Preprocessing
User Identification
Dataset
Divide dataset for snapshots
Mapping the Social relation Graph(GT,GT+1,....GT+n)
GT
GT+1
GT+n
Tracking dynamic communities
Community detection using Clique Louvain algorithm
Tracking the event of communities
GT
Ci
Ci+1
Ci+m
Tracking influence nodes based on community events in each time line
Influencer pool
Explore Pareto Front (t)
Choosing the influencer nodes using NSGAII-IM algorithm
Modeling the influence to measure the influence maximization using NSGAII-IM
Update
Pareto Front(t=t+1)

Figure 3-1 The Proposed System

3.3 The Datasets

Two real-world information spreading datasets will evaluate the proposed model; Facebook (Weskida and Michalski 2019), and Twitter (Dawar, Bera, and Goyal 2018).

3.3.1 Facebook Dataset

Facebook gives its users the area commonly referred to as the "wall" to express their ideas and opinions and engage with a virtual social network of friends.

This dataset includes a social network of Facebook friends and the posts made by each of them from September 25, 2006, to January 22, 2009. This dataset has been formed by two separated text files (Friendship links file and wall post file), where it is found around 876,993 wall posts on 46,952 users, and a unique ID identifies each user. A user ID defines each Facebook user, and the friendship between users ***i*** and ***j*** is shown as an undirected link with time stamps. The Facebook Wall Posts dataset is a subset of each user's posts on another user's wall between 09-14-2006 and 01-22-2009. Table 3-1 and Table 3-2 shows the contents of the dataset files, respectively.

The number of posts posted on the user's wall is considered a measure of user influence. Suppose user ***i*** posts to user ***j***'s wall. In that case, we believe this information to be propagated from user ***i*** to user ***j***, and the number of users posting to each user's wall is used to describe the user's influence in a timestamp.

Table 3-1. The Fields of Facebook Relation

No	Fields	Definition
1	Friendship date	Friendship link formed at a specific Unix timestamp.
2	User_ID & Friend_ID	Social relation between User_ID & Friend_ID.

Table 3-2. The Fields of Facebook Wall Post

No.	Fields	Definition
1	Wall post date	Unix timestamp when the user wall post.
2	User_ID$_1$	The user who issued the poster sender.
3	User_ID$_2$	The user who has the wall received.

3.3.2 Twitter Dataset

To start the collection process, "lady gaga" was picked as the most popular user on Twitter and randomly gathered 10,000 followers .These users are considered primary users and utilize a crawler to collect all the followers by traversing the "following" relationships. These followers are shown as a list of users with a total of 53022.

Table 3-3 Twitter Dataset File

No.	Fields	Definition
1	User_Name	The user name
2	Tweet_ID	The ID of the tweet
3	Time	The time of retweet
4	Via	The type of the web app
5	retweet_from	retweet _to_tweet(if not retweet, just "-1")
6	Reply	Reply_to_user reply_to_tweet(if not reply, just "-1")
7	Content	Please notice that we have already transferred the original content into the word index
8	Number_of_link_in_tweet	Here the links include URL, @user.
9	Link1	type_of_link1 link1
10	Link2	type_of_link2 link2
11	Link3	type_of_link3 link3

The crawler observed the change in network structure among 53022 users from October 1, 2010, to January 15, 2011, and got 19497551 dynamic tweets and 2466278 retweets. Consequently, the tweets and retweets among users represent the influence of the diffusion of information through the networks. Table 3-3 states the information about the data.

3.4 The Preprocessing Stage

In the case of the Twitter dataset, this stage aims to extract user activities from the original file and then build the social graph. The Facebook dataset, on the other hand, does not require that procedure because the relationships are explicitly present. The important steps involved in data preprocessing for the Facebook dataset and Twitter dataset as follows:-

- **User Identification**

  Each user has an identifier in Facebook files, while in a Twitter dataset, the user has a unique name and should be replaced by a unique number.

- **Divide dataset into snapshots**

  The network with history time is divided into a series of snapshots corresponding to the accumulation of observable interactions over a period of time (interaction network).

- **Relationship Extraction**

  Define source and target nodes from the relation information text file of the Facebook dataset to simulate the diffusion process. The Twitter dataset file extracts the username who tweeted and the username who retweeted.

- **Build the Social Graph**

3.5 Tracking Dynamic Communities

This stage tracks communities over a series of snapshots of a dynamic network. The clique-Louvain algorithm has been proposed to simulate community detection, improving the Louvain algorithm. Furthermore, the events of the communities are tracked in each snapshot based on its predecessor, except for the first snapshot, which has no precedent and is treated as a static network.

3.5.1 Community Detection Using Clique-Louvain Algorithm

It is possible to represent the network's dynamic behavior across time in the form of a sequence of non-overlapping snapshots. $\boldsymbol{S}=\boldsymbol{S_1}, \boldsymbol{S_2}\ldots\ldots \boldsymbol{S_T}$ represents the collection of all temporal snapshots. For each snapshot, ***M*** communities have been detected, then from each community number of influential candidates will be examined. The interaction among individuals is a good idea to group users in a social graph. For this reason, individuals in the same community are more potential to influence each other, at variance to those across communities.

Essentially, the traditional Louvain's algorithm forms the initial communities by making each node in the network a community; then, the modularity gain is the criterion for moving each community to each other. Intuitively, the operation suffers a time consumption, which is the motivation of this work to alleviate the problem. A clique-Louvain is a proposed algorithm to improve the Louvain algorithm in terms of time, taking into account the burden of the computational time of finding cliques. The traditional algorithm is linear with the number of edges, whereas the improved algorithm is linear with the number of cliques.

Generally, the key point of our proposed algorithm is looking for the cliques that are inherently found in networks to form the communities. Accordingly, given the undirected graph $\boldsymbol{G_T}= (\boldsymbol{V_T}, \boldsymbol{E_T})$, $\boldsymbol{V_T}$ denotes a set of vertices, and $\boldsymbol{E_T}$ means edges between vertices in snapshot ***T***, denoted $\boldsymbol{S_T}$. Furthermore, the number of communities might vary reliant the interactions in that time period.

Figure 3-2a illustrates a simple network of a coherent structure with different sizes as an example. A clique-Louvain algorithm proposes finding the cliques that involve three members at least as the first step of the algorithm (see Figure 3-2b). Principally, each clique or node (not a member in a clique) in the graph is considered a community. It is more likely the members of a single clique belong to one community practically. Under these conditions, the algorithm saves time computations from computing the modularity gain between communities in the first iteration. Calculating the modularity gain is the second step of the algorithm, next moving the vertices or cliques. These two steps are repeated until the modularity value is no longer improved and hierarchical communities are generated.

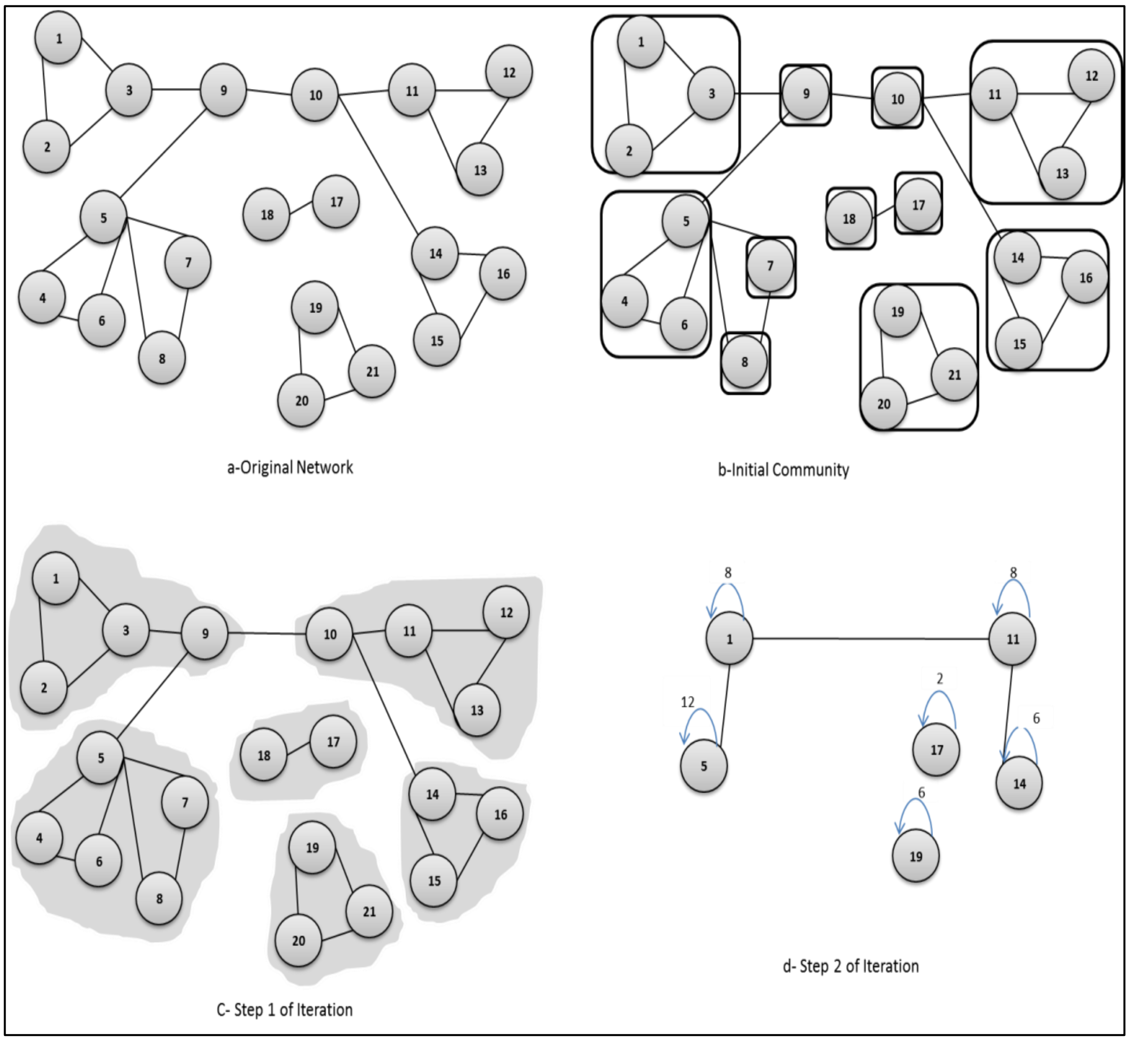


Figure 3-2 Clique-Louvain Algorithm Steps

In practice, the algorithm satisfies the stop criteria (when the value of the modularity is not improved) and has a lower execution time than the traditional algorithm.

Given the initial communities (clique or node), one can be chosen randomly and compute the modularity gain of merging it with a neighbor (clique or node). Next, moving the community chosen into the neighbor achieves a maximum value of modularity gain, as explained in Figure 3-2 (c). Finally, the actual process of the traditional algorithm is resumed, where each resultant community is represented as one node with a self-loop. The self-loop refers to the internal connection of nodes in the same community, and edges refer to the external connection of nodes with other nodes from different communities, as illustrated in Figure 3-2d. Algorithm 3-1 explains the steps in detail.

Algorithm 3-1 : Clique-Louvain Algorithm

```
Input:
G_org: The original graph
Output:
IndexCom: The communities set
G_new: A graph with communities
Begin:
1.   Index_Clique ← findclique(G_org) // call Algorithm 2-1
2.   G_new ← aggregation nodes which are in the same Clique based on Index_
     Clique.
3.   IndexCom ← The index of a clique or node as community// Initialize each
     clique as own community and the vertices that are not in the clique, each one as a
     community
4.   Repeat
5.       Modularity ← Q(G_new, IndexCom) compute with Equation (2-3)
6.       while one or more nodes are moved do
7.           for random i ∈ V(G_new) do
8.               bestGain ← -∞
9.               bestCommunity ← i
10.              for ∀ N_i do // For all nodes N neighboring to the node i
11.                  Calculate gainModularity ← ΔQ between nodes i and n
                     compute with Equation (2-4)
12.                  if bestGain < gainModularity then
13.                      bestGain ← gainModularity
14.                      bestCommunity ← n
15.                  End_if
16.              End_for
17.              IndexCom ← place i in the bestCommunity
18.          End_for
19.      End_while
20.      G_new ← aggregation nodes which are in the same community based on
         IndexCom
21.      IndexCom ← put each node of currentGraph in its own community
22.  until Modularity < Q(G_new, IndexCom)
23   End_Algorithm
```

3.5.2 Tracking the Event of Communities

Most social networks are dynamic; the communities in that network change over time. For each period, exhibit new users are joined or get removed regularly from the network. Based on this concept, analyzed a set of structural events between two successive periods to describe the evolutionary behavior in dynamic networks. Let $\boldsymbol{C_T}$ and $\boldsymbol{C_{T+1}}$ signify the set of communities respectively in snapshots S_T and S_{T+1} at two consecutive intervals.

To capture and identify evolutionary events of dynamic networks to perform actual matching between communities from different snapshots. We use the Jaccard coefficient for widely adopted binary combinations (as illustrated in Section 2.3.3). The events are explained in Figure 3-3.

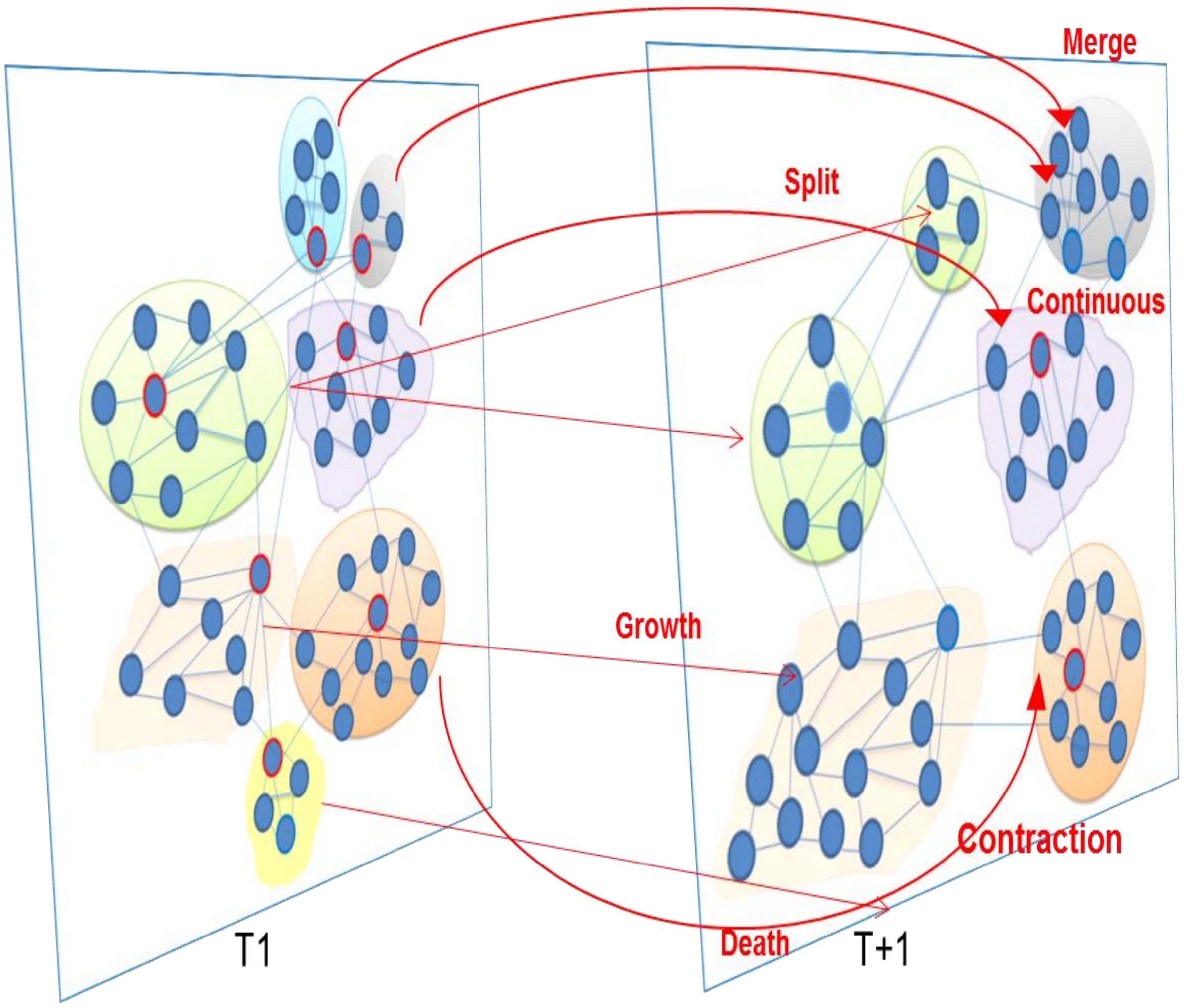


Figure 3-3 Events of Communities

3.6 Tracking Influence Nodes for Each Community

After the network has been partitioned into M communities for each snapshot, the number of influential candidates will be examined. The division of the graph $\boldsymbol{G_T}$ in snapshot $\boldsymbol{S_T}$ into disconnected subgraphs that represent the communities as $\boldsymbol{C_T} = \{\boldsymbol{C_T^1}, \boldsymbol{C_T^2}, \ldots\ldots, \boldsymbol{C_T^M}\}$, the subgraph $\boldsymbol{C_T^M} = (\boldsymbol{V_T^M}, \boldsymbol{E_T^M})$, where $\boldsymbol{V_T^M}$ denotes a set of vertices and, $\boldsymbol{E_T^M}$ denotes a set of edges between vertices in subgraph $\boldsymbol{M}$ of snapshot $\boldsymbol{T}$.

For each community, we propose NSGAII-IM algorithm that provides increased influence maximization and a decrease in the number of seeds. Basically, they are two conflicting objectives, as there should be a minimum number of seed nodes that have the best influence in the whole network. A series of steps must be taken to achieve these two conflicting goals, explained in the subsequent sections.

3.6.1 Explore Pareto Front from Community

The community may go through a series of events in the dynamic networks. The events that happened to this community can be traced from the previous snapshot. For example, the current community $\boldsymbol{C_T^M}$ in the snapshot $\boldsymbol{T}$ may be the result of the original community shrinking in snapshot $\boldsymbol{T\text{-}1}$, where it is possible to benefit from the Pareto Front of this community and include them as ready individuals when creating a population as the first step in the NSGAII-IM algorithm. Hence, the rest of the population members can be completed from the pool, as following below.

3.6.2 Influencer Pool

Let $\boldsymbol{Pool_T} \subseteq \boldsymbol{V_T}$ expresses a collection of potential influencers at $\boldsymbol{T}$, which was created through proactive proposals initiated by nodes $\boldsymbol{V_T}$. The size of pool influencers varies according to the community, in addition to the change in the topological structure of the network, e.g., $\boldsymbol{Pool_T}$ can be different from $\boldsymbol{Pool_{T+1}}$. An element $\boldsymbol{i} \in$ describes an influencer candidate in a pool. Specifically, the nodes whose have high centrality are based on a threshold determined according to experience. This centrality is determined by one of the following: degree, closeness, or eigenvector. Thus, for each network to be analyzed, the three centrality measures are applied, and the pool contains each node with a high centrality, where the same node could have a high degree, closeness, and eigenvector.

3.6.3 Choose the Influencer Nodes Using the NSGAII-IM Algorithm

In this work, the optimization of the influence maximization problem was formulated in the form of a multi-objective evolutionary algorithm, according to an improved model based on two opposing objectives. The first objective is to choose the smallest number of influential individuals, and the second is to achieve the highest spread of information within the network. According to that, the NSGAII algorithm had to be adapted to meet the goal and use the additional information hidden within the network structure to be used when feasible. Some changes in the initial population were performed to impact the quality of the solution. A pool with high centrality nodes was constructed, the individuals in the initial population are randomly selected from that pool. The Algorithm 3-2 has been explained as individual representative evolutionary operators and the objective function.

3.6.3.1 Individuals Representation

In the proposed scenario, the candidate solution individual can be represented as a set of nodes (seeds). The length of each individual is the same as the overall amount of nodes in the network. An individual is represented in binary encoding; if a node is within the seed set, then 1 is assigned to its position in the representation of the individual; otherwise, given to 0. However, this representation may not be implemented easily when the network is extensive in size. Therefore, it is convenient to keep only the seed number rather than its indices, as illustrated in Figure 3-4. In this context, the individuals are characterized by a variable rather than fixed length, as they depend on the number of nodes chosen within the seed nodes.

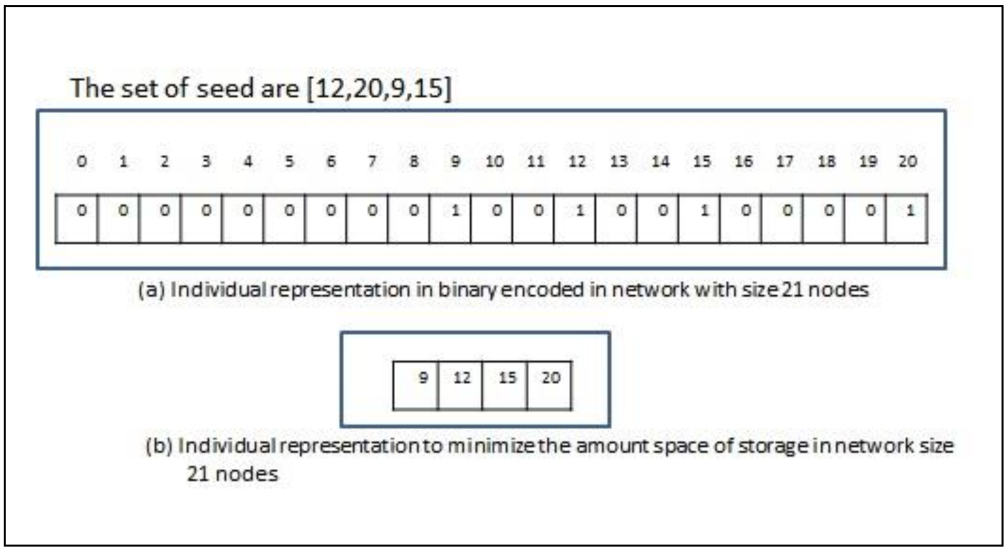

The set of seed are [12,20,9,15]
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
0 0 0 0 0 0 0 0 0 1 0 0 1 0 0 1 0 0 0 0 1
(a) Individual representation in binary encoded in network with size 21 nodes
9 12 15 20
(b) Individual representation to minimize the amount space of storage in network size 21 nodes

Figure 3-4 Individual Representation

Algorithm 3-2: Population Representation Algorithm

Input:
N: Population size
Pool: Pool of community in current snapshot
C_T: community in snapshot (T)

Output: Pop : Population

Begin:
```
1.  Pop ←∅
2.  indv←∅
3.  A=True
4.  while  size(Pop)<= N do // represent each individual in Population
5.      If C_T not has an event from snapshot T-1 then// Initial snapshot when
        all communities appear for the first time or if the birth community event
        is in the snapshots that follow the first snapshot
6.          j←0
7.          rand←Choice random number between 5% & 7% of community
            size
8.          while j<= rand do
```

```
9.                  node← Choose a node randomly from Pool that is not
                    previously shown in indv
10.                 indv←indv ∪ node
11.                 j←j+1
12.           End_ while
13.      Otherwise if A=True then // when the community is formed as a result
         of an event from a previous snapshot and that event is not birth
14.           Par_indv ← Pareto Front from C_{T-1}
15.           for each indv in Par_indv do
16.                 if size(Pop)<N then
17.                      indv ←indv ∩ Pool //delete each node not appear in the
                         current snapshot
18.                      j ← length of indv
19.                      if j< 5% length of community then
20.                           Rand ← Chose random number between 5% & 7%
                              of community size
21.                           while j ≠ rand do
22.                                 Node ← Chose node randomly from Pool ∉
                                    indv
23.                                 Indv ←indv ∪ node
24.                                 J ← j+1
25.                           End_ while
26.                      else j> 7% length of community then
27.                      rand← Chose random number between 5% & 7% of
                         community size
28.                      while j≠ rand do
29.                           delete node randomly from indv
30.                           J ← j-1
31.                      End_ while
32.                      Pop ← Pop ∪ indv
33.                      End_ if
34.                 End_ if
35.           End_ for
36.           A ← False
37.      Pop ← Pop ∪ indv
38.      End_ if
39. End_ for
40. Return Pop
41. End_Algorithm
```

Algorithm 3-2 shows how to represent an initial population. Steps 5 to step 12 relate to communities in an initial snapshot or case of birth of a community. While steps 13 to 38 explain the procedure related to the population's individuals who rely on the Pareto imported from the community in the previous snapshot. In other words, the individuals concerned with steps 13 to 38 whose communities have been formed due to the events. Pareto Front is a group of influential individuals in the community's previous snapshot after applying an NSGAII algorithm used as candidate individuals to the next snapshot to convergence the solution. Each individual representing a set of nodes, to filter out nodes that are not in the community for the current snapshot, or to make their length appropriate to the size of the set of seed, steps 16 through 36 are performed.

3.6.3.2 Evolutionary Operators

Crossover is an evolutionary operator used to produce a new offspring of a population whose features have been inherited from those of parent's specifications. In this work, the simple one-point crossover has been applied with some modification to avoid repeating the node in the identical offspring, as demonstrated in Section (2.4.2). Repeating the node for the same individual is not accepted in this scenario. Figure 3-5a illustrates the basic one-point crossover, where the offspring 1 has node 12, which is repeated twice. At the same time, Figure 3-5b states the modified version of the simple one-point crossover proposed in this work. In brief, the node ID from one parent is chosen randomly to be a crossover point .

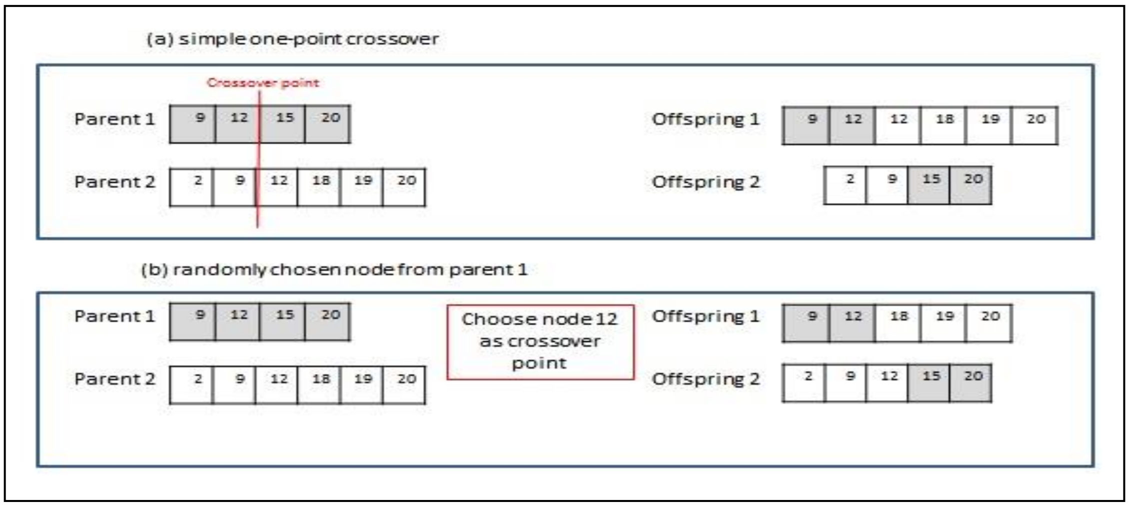


Figure 3-5 Crossover Operation

In this context, offspring 1 can be generated by selecting the node ID equal to or less than the crossover point and larger than the crossover point from ID nodes of parent 1 and parent 2, respectively. On the contrary, offspring 2 consists of the nodes ID larger than that of the crossover point from parent 1 and equal to or less than parent 2. The crossover operation has been explained in Algorithm 3-3.

Algorithm 3-3: Crossover Operation Algorithm

```
Input: :
Pa1: candidate parent1
Pa2: candidate parent2
Cr: crossover rate

Output: Offspring

Begin:
 1. Sorting $Pa_1$, $Pa_2$ descending
 2. if random number < Cr then
 3.         Cp// Cp is the node randomly chosen from $Pa_1$ or $Pa_2$ (crossover point)
 4.         for each node in $Pa_1$ do
 5.            if node < =Cp then
 6.               $Ch_1 \leftarrow Ch_1 \cup \{nodes\}$
 7.            else
 8.               $Ch_2 \leftarrow Ch_2 \cup \{nodes\}$
 9.            End-if
10.         End-for
11.         for each node in $Pa_2$ do
12.            if node < =Cp then
13.               $Ch_2 \leftarrow Ch_2 \cup \{nodes\}$
14.            else
15.               $Ch_1 \leftarrow Ch_1 \cup \{nodes\}$
16.            End-if
17.         End-for
18. else
19.         $Ch_1 \leftarrow$ nodes $Pa_1$
20.         $Ch_2 \leftarrow$ nodes $Pa_2$
21. Offspring $\leftarrow Ch_1 \cup Ch_2$
22. End_if
23. Return Offspring
24. End_Algorithm
```

As for the mutation, which is a second operator, the traditional way has been used, including the addition, replacement, or removal of one node from an individual. In particular, the replaced or addition of a node in an individual is conditional on non-repetition. The individual does not have to contain the same node more than once.

3.6.3.3 Fitness Function

Concerning the multi-objective, the best candidate solution represents a set of seeds that meet two conditions. The first is the best diffusion (i.e., the set of seeds fulfills maximum influence on others). The second is realizing this diffusion with a minimum number of members within that particular set. It is worth mentioning that the WIC model that will be explained in Section (3.6.4) has been implemented in modeling the propagation. The proposed NSGAII-IM algorithm is presented in the Algorithm 3-4, as following detailed steps:

Algorithm 3-4: NSGAII-IM Algorithm

```
Input:
N: population size
No.Gen : max number of generation
Output: best Pareto Front
Begin:
 1.  Pool ← Centrality(node) // create the pool by computing the centrality of each
     node and choosing the nodes with higher centrality
 2.  Pop // call Algorithm 3-2 initialize the population consisting of N random
     individuals with a variable size from pool and Par_indv
 3.  Pop_ch ← Ø // The population of children is empty initially
 4.  for i=1 to No.Gen do
 5.      while s<N/2 do
 6.          Candidat e← selection(Pop) //select two individuals
 7.          Offspring ← Crossover(Candidate) // call Algorithm 3-3.
 8.          Offspring ← Mutate (Offspring)
 9.          Pop_ch ← Offspring
10.      End_while
11.      Pop ← Pop∪ Pop_ch
12.      for each indv in Pop do
13.          Eval(indv) // call Algorithm 3-5
14.      End_ for
15.      Assign rank and CrowdingDistance// accordion to Pareto dominance
         sorting a and CrowdingDistance // call Algorithm 2-2
16.      Pop ← SelectBest(N) // Select the best N individuals based on Rank and
         CrowdingDist
17.  End_ for
18.  return best Pareto Front
```

```
19. End_Algorithm
```

3.6.4 Modeling Information Diffusion Using WIC Model

In this stage, a developed IC model has been proposed to simulate the diffusion tracking is modeled based on the social relations of the graph. The new Weighted Integration Cascade WIC model considers that the interaction of a node is as essential as its relationships. Thus, the user activity and degree can be merged in one measure that can locally define the propagation probability between nodes. According to that, the propagation probability $\boldsymbol{P_{i,j}}$ between nodes, ***i*** and ***j*** could be defined as follows:

$$P_{i,j} = \alpha \frac{d_i}{\sum_{k \in N_j} d_k} + \beta \frac{I_{i,j}}{\sum_{k \in N_j} I_{k,j}} \qquad (31)$$

The first term of Equation (3-1) denotes the degree of the active node ***i*** divided by the sum of degrees of the neighbor nodes that have a direct edge to ***i***. In contrast, the second term represents the past interactions between node i and j divided by the sum of the interactions of node j with its other neighbor nodes. The parameters ***α*** and ***β*** are the constant values between (0, 1).

In the following scenario, consider a sub-graph ***Ğ⊆G*** with inactive node ***j***. Assume that each neighbor's persuasiveness is represented by the degree of the neighbor. Node ***j***'s neighbors have a total degree of 10, and the degrees of neighbors nodes are ***x, y, z***, and ***i*** are 1, 3, 2, 4, respectively; the probabilities set are 0.1, 0.3, 0.2, 0.4, which reflect the degree of each neighbor. Meanwhile, the number of times interaction between j and neighbors node ***x, y, z***, and ***i*** are 3, 5, 2 and 10. The total interaction of node ***j***'s neighbors is 20, as shown in Figure 3-6a.

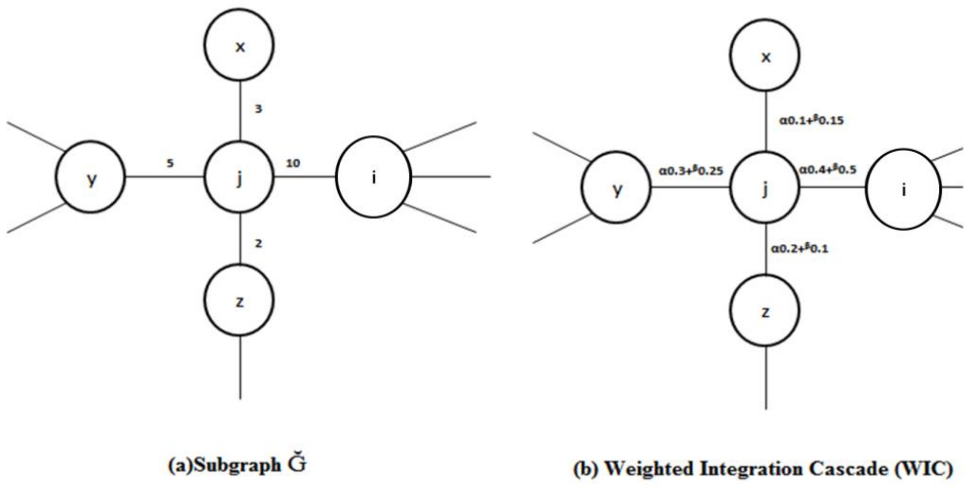


Figure 3-6 (a) Sub-GraphĞ (b) Weighted Integration Cascade

The probabilities as 0.15, 0.25, 0.1, and 0.5 reflect the interaction of each neighbor, which means that the probability of propagating information from $\boldsymbol{i}$ to $\boldsymbol{j}$ is higher than the probability of information propagating from other neighbor nodes as shown in Figure 3-6b. Hence, the fixation of the influence probability over each edge is not feasible in the real-world scenario. Algorithm 3-5 represents the step-by-step procedure of the proposed model.

Algorithm 3-5: WIC Model

```
Input:
$\boldsymbol{G}(\boldsymbol{V}, \boldsymbol{E})$: Social Graph
$\delta$ : individual( Seed nodes)
θ: Activation threshold

Output: Size of active nodes

Begin:
  1.  for each directed edge $\boldsymbol{e}$(i, j) ∈ $\boldsymbol{E}$ do
  2.  Find P(i, j) the propagation probability of nodes i and j  //using Equation(3-1)
  3.  Active ← $\delta$ // Active is a set containing all active nodes after propagation ended;
      initially, it contains all seed nodes
  4.  $New_{Active}$ ← $\delta$ // nodes that were active most recently
  5.  while $New_{Active}$ not empty do
  6.          $New_{one}$ ← Ø
  7.          for each i ∈ $New_{Active}$ do
```

```
8.               for each v (v∈ $N(u)$) do //N is the direct neighbor of u
9.                   if P(i, j)> θ // θ is the activation threshold then
10.                      $New_{one}$ ← $New_{one}$ ∪ {j}// add j to $New_{one}$
11.                  End_if
12.              End_for
13.          End_for
14.          $New_{Active}$ ← $New_{one}$ ∩ Active // So as not to contain previously active
             nodes
15.          Active ← $New_{Active}$
16.  End_ while
17.  return Size of Active
18.  End_Algorithm
```

Chapter Four:

Results and Discussion

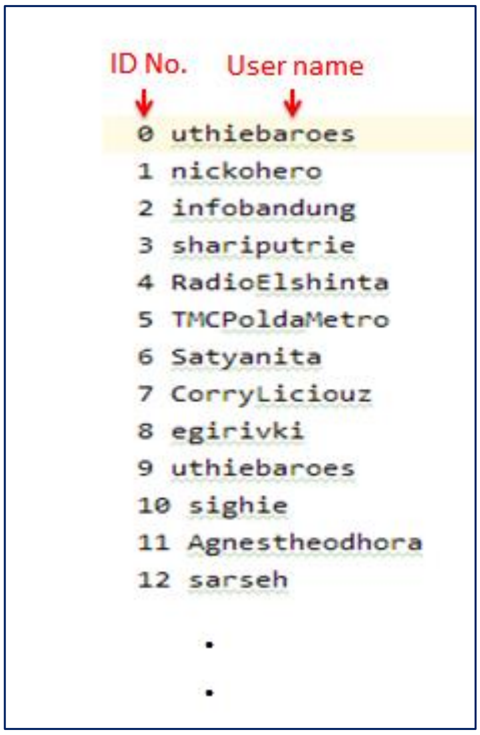
4.1 Introduction

The findings and discussions chapter will showcase the results of the datasets that have been used in this work, which are commonly known in this field, namely the Facebook and Twitter datasets. Additionally, the Facebook and Twitter datasets used in some experiments for various purposes will be discussed during the chapter. Results of the primary research have been separated according to the order of Chapter Three.

4.2 Preprocessing Results

In the present experiment, the files provided by the considered datasets are subject to many processing tasks that form interpretable files required to accomplish the proposed model stages. The User Identification task is performed only for the Twitter dataset since the users have already been anonymized in the Facebook dataset. They are unique in all dataset files, following the extraction of the user's name who has retweeted the post. A user's name tweeted from a dataset file, a sequence of integer values starting from 0, is used as an identifier (ID). The names of users are unified with their corresponding IDs in the entirety of the Twitter dataset files. Figure 4-1 shows a sample of the resulting file.

Concerning the process of dividing the dataset into several snapshots. The approach used to model such a temporal/dynamic social network is to transform a dynamic network into a static network at different snapshots. Each snapshot includes interactions that occurred during its specified time frame, the duration length of which can be picked based on how dynamic the network is.



```
0 uthiebaroes
1 nickohero
2 infobandung
3 shariputrie
4 RadioElshinta
5 TMCPoldaMetro
6 Satyanita
7 CorryLiciouz
8 egirivki
9 uthiebaroes
10 sighie
11 Agnestheodhora
12 sarseh
        .
        .
```

Figure 4-1 User Identification in a Twitter Dataset

The observed structures and the analysis results have been significantly influenced by the duration and number of the snapshots. If the time is divided asymptotically (i.e., snapshots of a short period), the dynamic network will contain a lot of temporal details, which leads to the detection of many unimportant events. On the other hand, the important temporal information may be omitted if large duration time of snapshots in the dynamic network. Hence, there must be a trade-off to distinguish between unnecessary and noise temporal information and between useful and unuseful information. However, several experiments were performed to determine the appropriate temporal resolution for dynamic networks.

The Facebook dataset offers the advantage of allowing users to post to their walls, which is a type of social networking interaction. Friends can leave comments on a user's wall on Facebook, and anyone who visits the user's profile can see them. As a result, wall Posts are a type of broadcast message service available on the site. The information on the wall posts dates from September 25, 2006, to January 22, 2009. We found 876,993 wall posts from 46,952 users in total. For this dataset, we specify the duration of each snapshot to be four months. Consequently, we have six snapshots in this dataset.

Twitter dataset crawled the following links between 53022 users from Twitter time stamps from October 1, 2010, to January 15, 2011. The average time interval between two consecutive timestamps (each snapshot) is about two weeks. There are 2466278 retweets of all the six network snapshots.

To construct the social graphs (set of nodes and edges), the relations information is extracted from specific data fields in the Facebook dataset. While in the Twitter data set, the tweet and retweet can be extracted from the original dataset file, such as the sample in Figure 4-2, and it takes time to build the relationship between the nodes. The source node is the ID of the user who tweeted when the destination nodes are the ID of the retweeting user. The results from the two data sets are stored in text files for each snapshot, where each line consists of a source node, followed by the subsequent nodes (destination nodes), as shown in Figure 4-3.

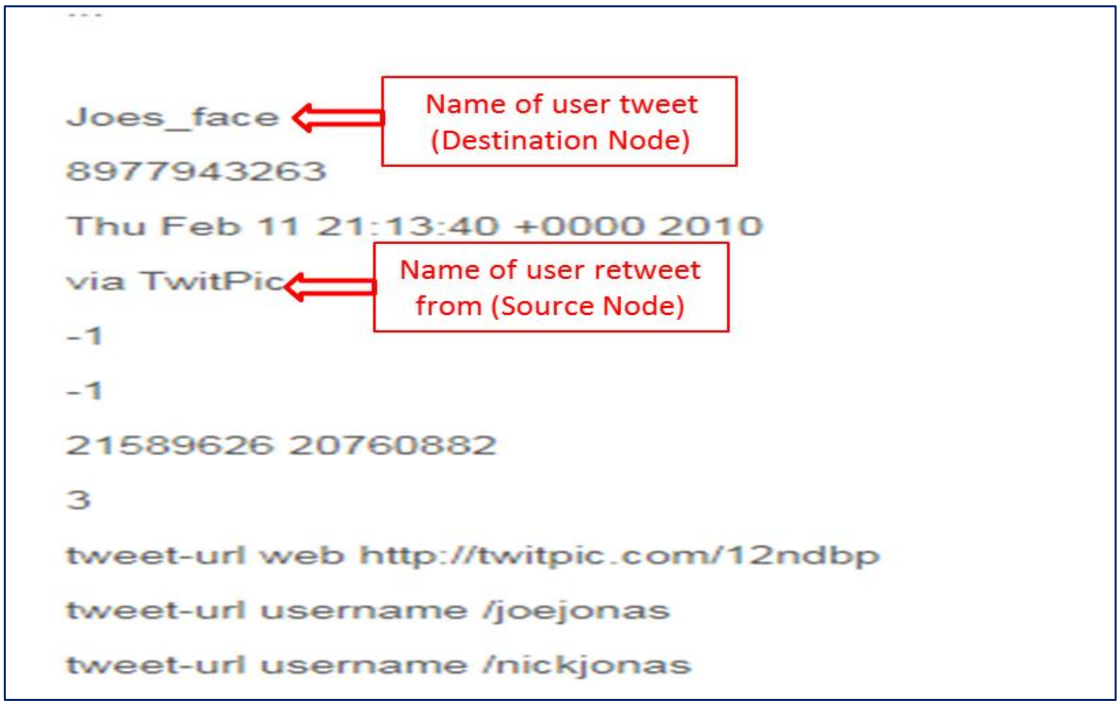
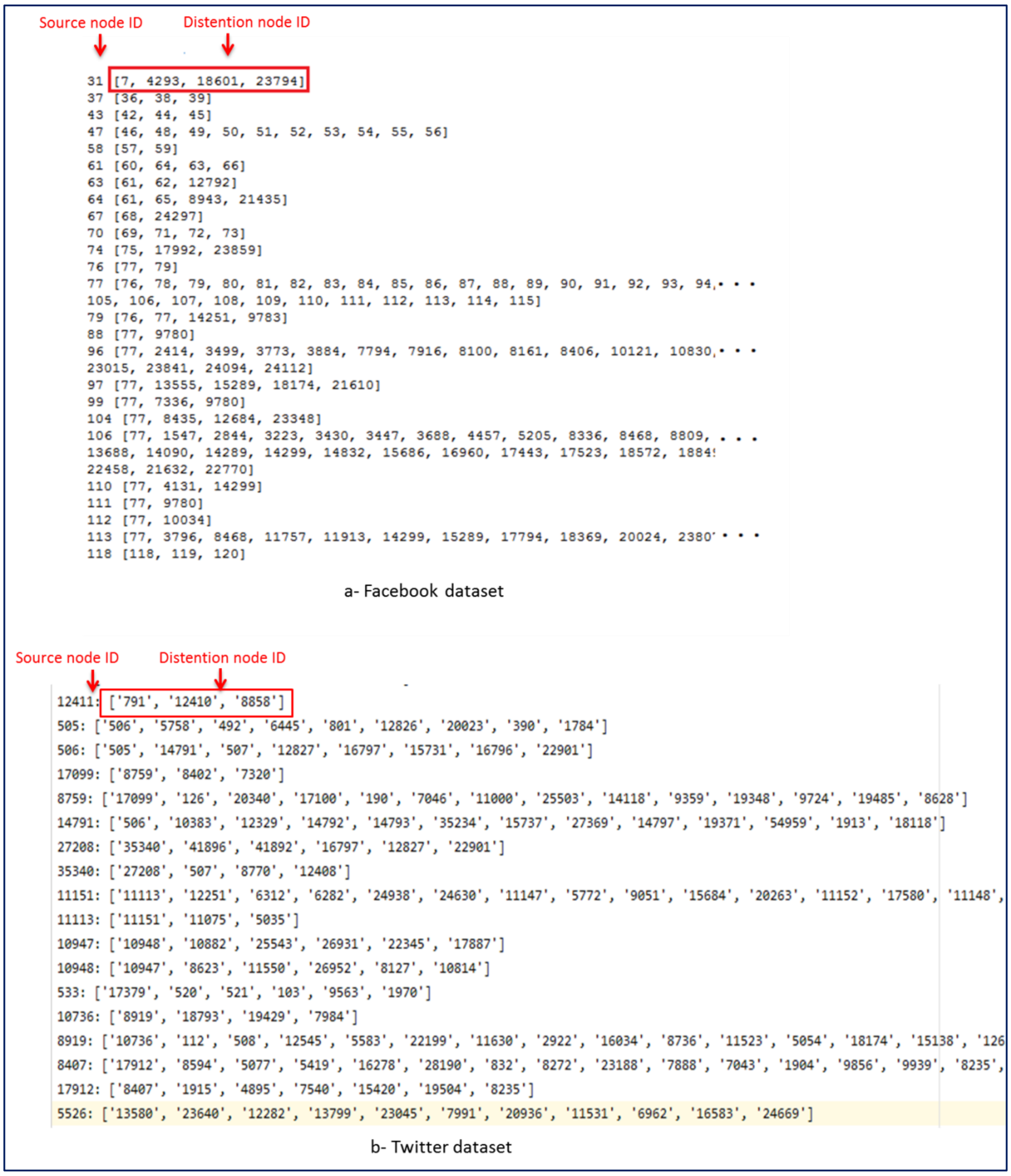

```
...
Joes_face
8977943263
Thu Feb 11 21:13:40 +0000 2010
via TwitPic
-1
-1
21589626 20760882
3
tweet-url web http://twitpic.com/12ndbp
tweet-url username /joejonas
tweet-url username /nickjonas
```
Name of user tweet (Destination Node)
Name of user retweet from (Source Node)

Figure 4-2 Sample of the Twitter Dataset



```
31 [7, 4293, 18601, 23794]
37 [36, 38, 39]
43 [42, 44, 45]
47 [46, 48, 49, 50, 51, 52, 53, 54, 55, 56]
58 [57, 59]
61 [60, 64, 63, 66]
63 [61, 62, 12792]
64 [61, 65, 8943, 21435]
67 [68, 24297]
70 [69, 71, 72, 73]
74 [75, 17992, 23859]
76 [77, 79]
77 [76, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94,• • •
105, 106, 107, 108, 109, 110, 111, 112, 113, 114, 115]
79 [76, 77, 14251, 9783]
88 [77, 9780]
96 [77, 2414, 3499, 3773, 3884, 7794, 7916, 8100, 8161, 8406, 10121, 10830,• • •
23015, 23841, 24094, 24112]
97 [77, 13555, 15289, 18174, 21610]
99 [77, 7336, 9780]
104 [77, 8435, 12684, 23348]
106 [77, 1547, 2844, 3223, 3430, 3447, 3688, 4457, 5205, 8336, 8468, 8809, . . .
13688, 14090, 14289, 14299, 14832, 15686, 16960, 17443, 17523, 18572, 1884
22458, 21632, 22770]
110 [77, 4131, 14299]
111 [77, 9780]
112 [77, 10034]
113 [77, 3796, 8468, 11757, 11913, 14299, 15289, 17794, 18369, 20024, 2380 • • •
118 [118, 119, 120]
```

```
12411: ['791', '12410', '8858']
505: ['506', '5758', '492', '6445', '801', '12826', '20023', '390', '1784']
506: ['505', '14791', '507', '12827', '16797', '15731', '16796', '22901']
17099: ['8759', '8402', '7320']
8759: ['17099', '126', '20340', '17100', '190', '7046', '11000', '25503', '14118', '9359', '19348', '9724', '19485', '8628']
14791: ['506', '10383', '12329', '14792', '14793', '35234', '15737', '27369', '14797', '19371', '54959', '1913', '18118']
27208: ['35340', '41896', '41892', '16797', '12827', '22901']
35340: ['27208', '507', '8770', '12408']
11151: ['11113', '12251', '6312', '6282', '24938', '24630', '11147', '5772', '9051', '15684', '20263', '11152', '17580', '11148',
11113: ['11151', '11075', '5035']
10947: ['10948', '10882', '25543', '26931', '22345', '17887']
10948: ['10947', '8623', '11550', '26952', '8127', '10814']
533: ['17379', '520', '521', '103', '9563', '1970']
10736: ['8919', '18793', '19429', '7984']
8919: ['10736', '112', '508', '12545', '5583', '22199', '11630', '2922', '16034', '8736', '11523', '5054', '18174', '15138', '126
8407: ['17912', '8594', '5077', '5419', '16278', '28190', '832', '8272', '23188', '7888', '7043', '1904', '9856', '9939', '8235',
17912: ['8407', '1915', '4895', '7540', '15420', '19504', '8235']
5526: ['13580', '23640', '12282', '13799', '23045', '7991', '20936', '11531', '6962', '16583', '24669']
```

Figure 4-3 Sample of Relation

In Twitter files, each line represents a set of edges from the node that tweets, followed by the nodes that retweet it in a specific snapshot. To this end, the information needed to create a social graph is available, and for the memory space issue, we chose the largest strongly connected component in the generated graph. Meanwhile, for the Facebook files, any line represents the user's ID followed by the ID of their friends with whom they have interacted during this specific snapshot.

4.3 Discussion of Tracking Dynamic Communities

To simulate tracking of communities over a series of snapshots of a dynamic network. The clique-Louvain algorithm is proposed to detect the communities in each snapshot as the Louvain algorithm is improved. Furthermore, the first snapshot is treated as a static network since it has no precedent. As for the rest of the snapshots, the communities' events are traced back to their predecessors.

4.3.1 Clique-Louvain Algorithm Evaluation

A clique-Louvain in Algorithm 3-1 has been proposed to improve the Louvain algorithm for community detection. Figure 4-4 shows a real sample of the discovered community structure.

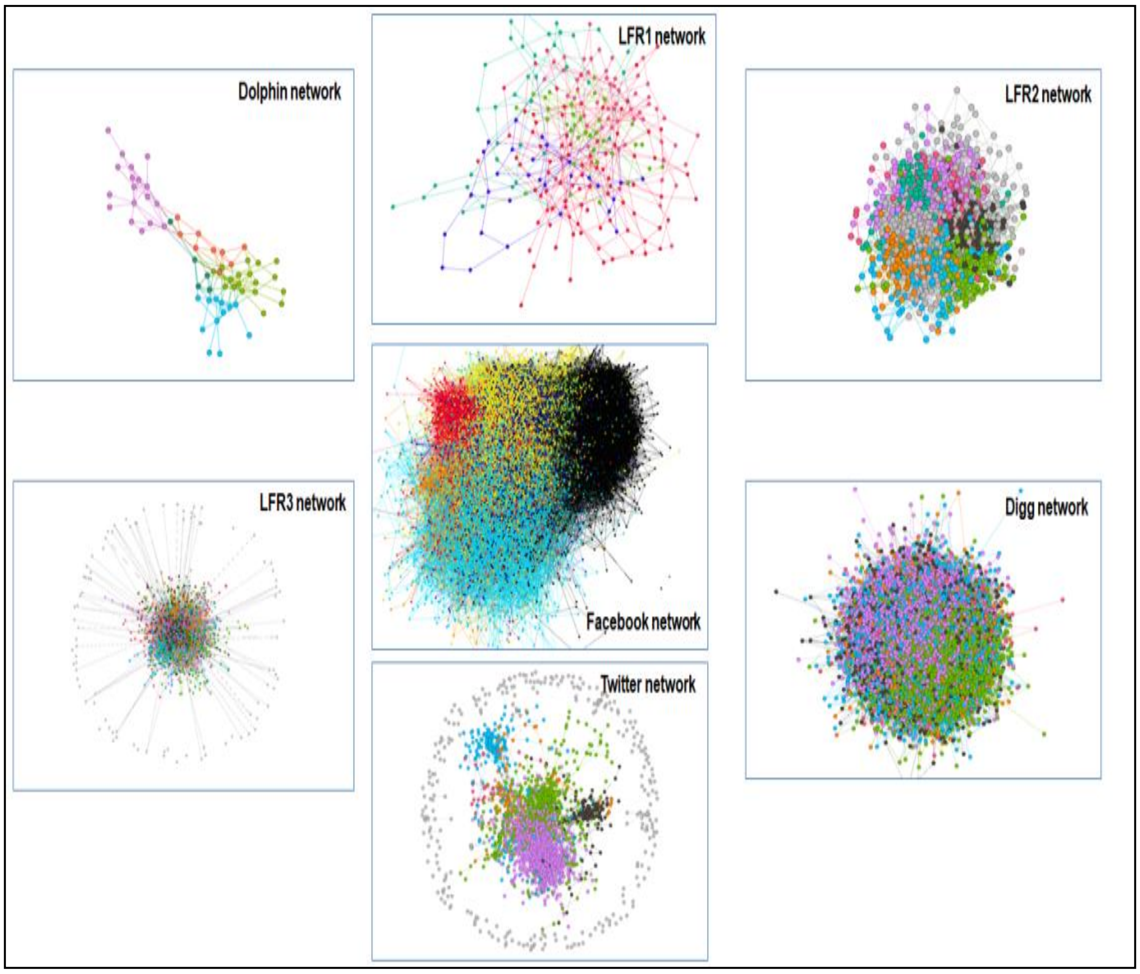
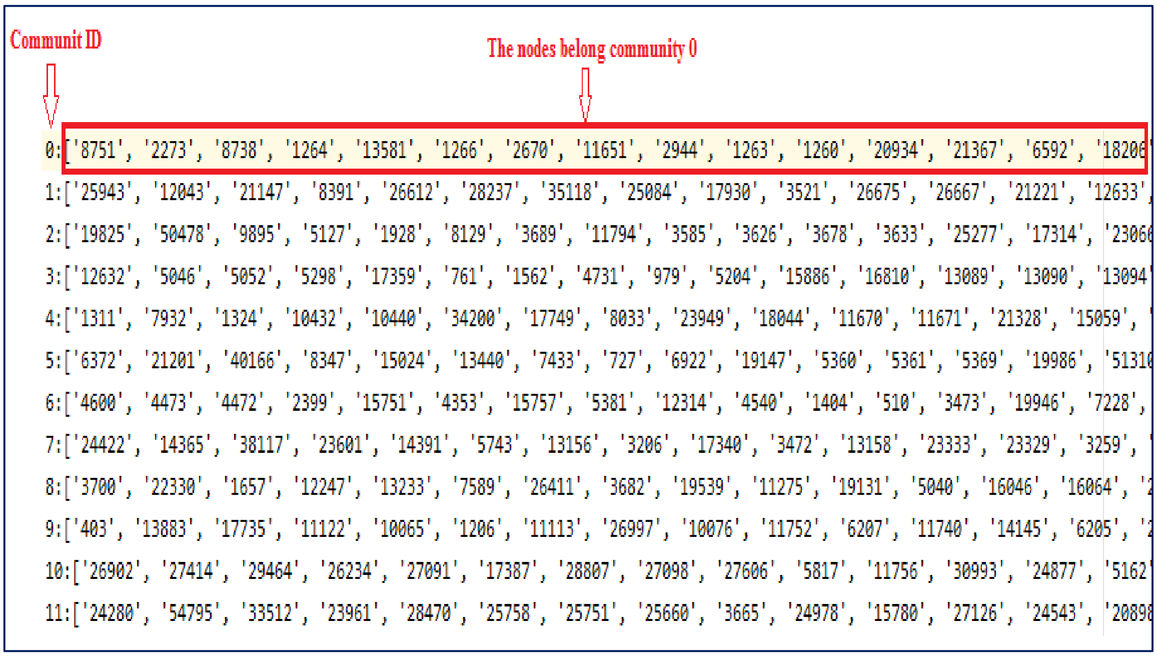


Figure 4-4 Community Detection in Different Networks

The network has different communities, and each community has a unique colour. The community detection results are stored in a file, in which the communities are identified by integer numbers indexed from 0. The file contains the community details, wherein in each row, the community ID is displayed, followed by all the node IDs belonging to the given community, as shown in Figure 4-5.



```
0:['8751', '2273', '8738', '1264', '13581', '1266', '2670', '11651', '2944', '1263', '1260', '20934', '21367', '6592', '18206
1:['25943', '12043', '21147', '8391', '26612', '28237', '35118', '25084', '17930', '3521', '26675', '26667', '21221', '12633',
2:['19825', '50478', '9895', '5127', '1928', '8129', '3689', '11794', '3585', '3626', '3678', '3633', '25277', '17314', '23066
3:['12632', '5046', '5052', '5298', '17359', '761', '1562', '4731', '979', '5204', '15886', '16810', '13089', '13090', '13094'
4:['1311', '7932', '1324', '10432', '10440', '34200', '17749', '8033', '23949', '18044', '11670', '11671', '21328', '15059',
5:['6372', '21201', '40166', '8347', '15024', '13440', '7433', '727', '6922', '19147', '5360', '5361', '5369', '19986', '51310
6:['4600', '4473', '4472', '2399', '15751', '4353', '15757', '5381', '12314', '4540', '1404', '510', '3473', '19946', '7228',
7:['24422', '14365', '38117', '23601', '14391', '5743', '13156', '3206', '17340', '3472', '13158', '23333', '23329', '3259',
8:['3700', '22330', '1657', '12247', '13233', '7589', '26411', '3682', '19539', '11275', '19131', '5040', '16046', '16064',
9:['403', '13883', '17735', '11122', '10065', '1206', '11113', '26997', '10076', '11752', '6207', '11740', '14145', '6205',
10:['26902', '27414', '29464', '26234', '27091', '17387', '28807', '27098', '27606', '5817', '11756', '30993', '24877', '5162'
11:['24280', '54795', '33512', '23961', '28470', '25758', '25751', '25660', '3665', '24978', '15780', '27126', '24543', '20898
```

Figure 4-5 Sample of Facebook Communities

The modularity is used to measure the quality of divided communities, and computation time is utilized to validate the performance of clique-Louvain compared with traditional Louvain's algorithm.

In this work, two types of datasets have been chosen: synthetics such as LRF1, LRF2, LRF3, and real networks such as DIGG, Dolphin, Facebook, and Twitter. It is important to note that some datasets, such as DIGG and Dolphin, have been used in the intermediate stages of the work because their conditions do not correspond to the requirements of the remaining stages. For example, the DIGG data has a two-week interaction time between users, which is insufficient to track the influence over time. The Dolphin dataset, on the other hand, is not considered a dynamic network.

Facebook and Twitter datasets are examples of data that persists across all stages. For evaluation purposes, the performance of the proposed algorithm has to be compared with the original Louvain algorithm.

LFR is a Synthetic benchmark network (see Section 2.3.1), which has many adjustable parameters that allow fast graphic prototypes of varying degrees of community structure. In this step, three scenarios with different configurations have been applied. In the first scenario, the LFR1 network consists of 250 nodes; average node degree 5; node power-law distribution 3; community-scale power-law index 1.5; and community structure definition is equal to 0.1. In the second and third scenarios, the LFR2 and LFR3 networks have 1000, 10000 nodes, respectively, with the same parameters :average node degree 8; the node power-law distribution 2; community-scale power-law index 2; and community structure defined as 0.56.

In order to validate the proposed algorithm, a more realistic network environment has been utilized, such as; Dolphin Social Network, DIGG, Facebook, and Twitter. The Dolphin Social Network is a social network of bottlenose dolphins, where the nodes are the bottlenose dolphins, and an edge indicates a frequent association. The Dolphins were observed between 1994 and 2001. While DIGG is the friendship network among bloggers, each node in the network is a user, and each directed edge denotes that a user replied to another user. The dataset was observed from 2005 to 2009. Finally, both Facebook and Twitter datasets have been explained in Section (4.2). Table **4-1** demonstrates the related information of other datasets.

Table 4-1. Real Dataset Information

Dataset	Dolphin	Digg	LFR1	LFR2	LFR3
Vertex	62	193808	250	1000	10000
Edges	159	886335	491	4268	33794
Average degree	5.1290	9.1465	5	8	8

Time Computation Measurement

Louvain and the clique-Louvain algorithms have been implemented on the synthetic and real networks for comparison purposes in terms of execution time and modularity. Figure 4-6 shows the performance of the algorithms over seven datasets versus time, where the vertical axis is the execution time, and the horizontal axis is the dataset kind.

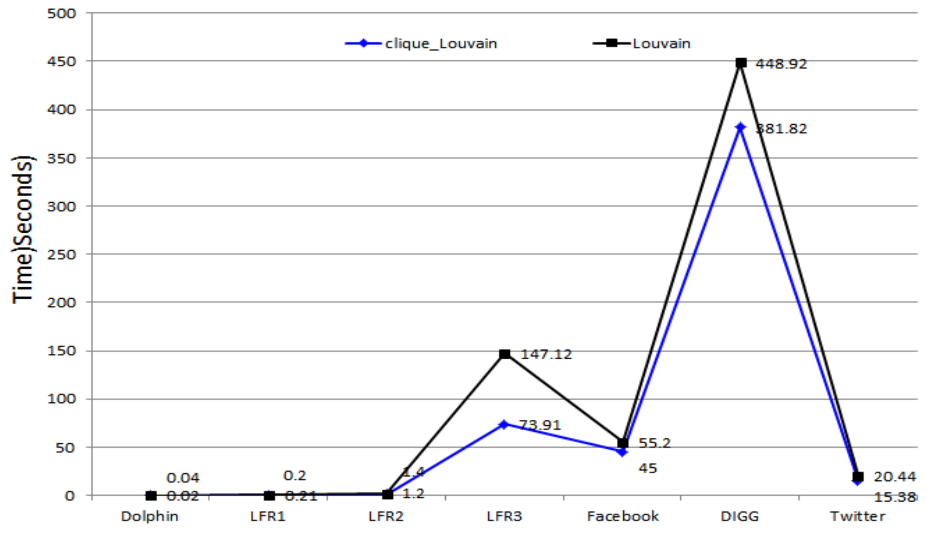


Figure 4-6 Execution Time of Clique-Louvain and Louvain Algorithms

As illustrated, the performance of the clique-Louvain algorithm is a bit better than the traditional algorithm when applied in artificial networks. Whereas the algorithm records a very good performance when applied in the real networks. Generally, the execution time increases as the size of the dataset increases. However, the matter may differ in the case of synthetic networks, which need more running time than real networks despite the small size of their data. The difference in execution time between the original and improved algorithm is clearer as the size of the network increases.

Modularity Measurement

A modularity measure is an objective function to evaluate communities' quality. So modularity is the important standard to measure the quality of the Louvain algorithm, as explained in Equation (2-3). The results obtained in Figure 4-7 illustrated the modularity measure of Louvain and clique-Louvain algorithms under various datasets. It can be said the results are relatively close to each other and can overlook the little difference versus speed up the proposed algorithm.

More precisely, the modularity of the clique-Louvain algorithm is identical, if not better, than that of the traditional algorithm when applied in LRF1, LRF2, LRF3, DIGG, and Twitter. As for the Facebook and Dolphin datasets, the modularity is a bit less than that of the traditional algorithm, which may be the reason is the structure of the dataset. Indeed, the structure of the Facebook dataset is characterized by largely overlapping cliques, which may affect the modularity.

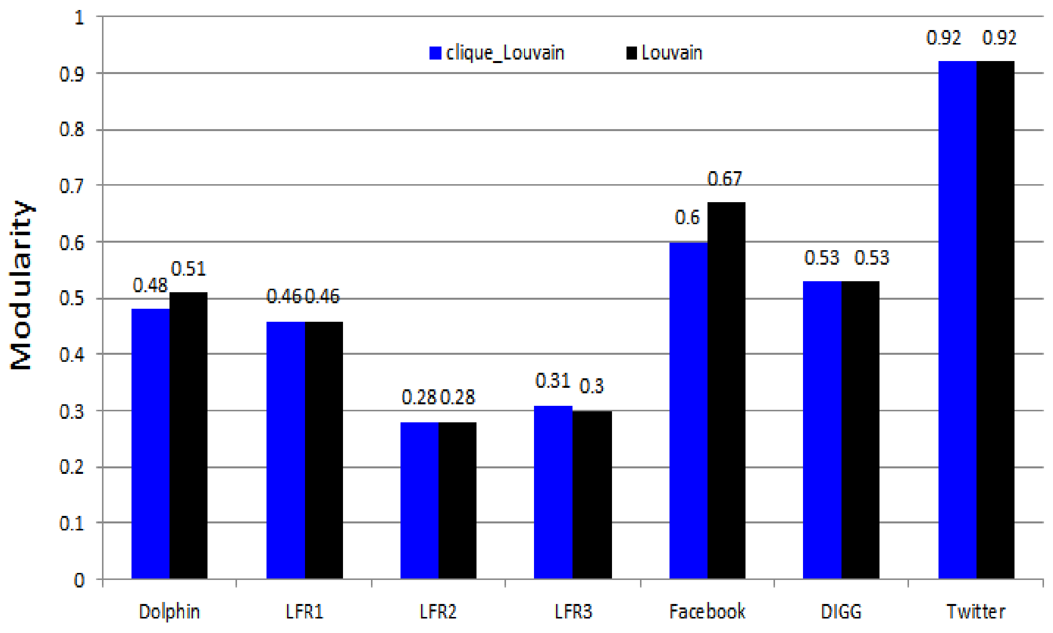


Figure 4-7 Modularity Measure of clique-Louvain and Louvain

In nature, modularity and execution time are very important measures to assess the performance of an algorithm. However, setting a trade-off between execution time and quality may depend on the applications. The two parameters may relate to one another somewhat inversely since the traditional Louvain is a greedy algorithm to approximate the optimal modularity of the graph.

4.3.2 Tracking and Analysis of the Events of Communities

Communities are classified according to the changes they have undergone over time. We consider seven events: Birth, Death, Growth, Shrink, Continue, Split, and Merge. These events are based on the relationship between communities, and their parameters are determined based on the Jaccard parameter (named similarity threshold). The histogram in Figure 4-8 represents the number of events of communities in each snapshot of the Facebook dataset. Generally, we observe an increase in the number of events steadily over time from snapshot 2 to snapshot 5, consistent with the number of registered Facebook users increasing month by month from 2006 to 2008.

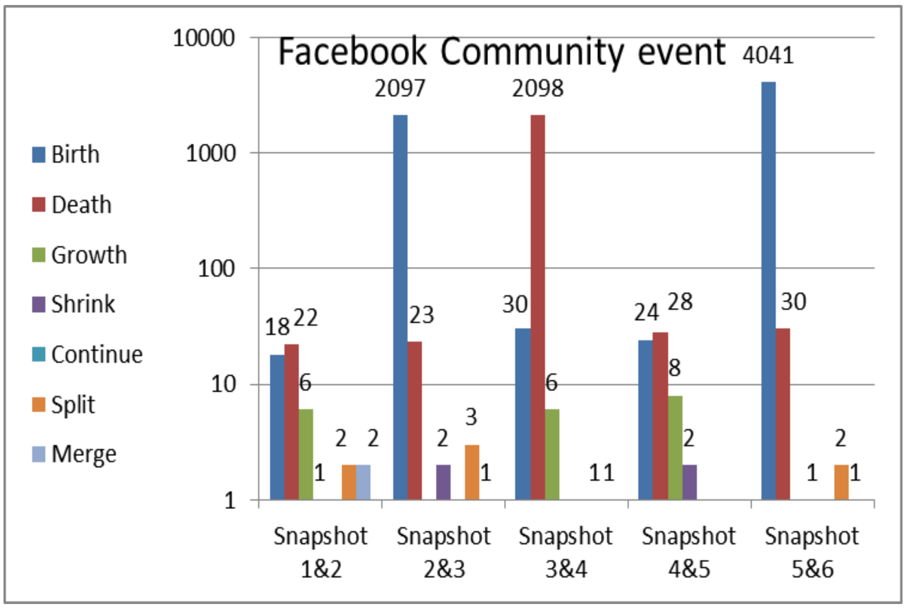


Figure 4-8 The Events Detected in Facebook Networks

Several experiments were performed to choose the appropriate period for each snapshot. Figure 4-9 shows the difference in the shape of the network after discovering the communities when choosing four different periods for each snapshot. We notice that several nodes are not linked to each other to form a community when the duration is only one month, leading to small communities consisting of two or more nodes. This is illustrated by Figure 4-9a. However, when choosing a more extended period, which is two months three months until arrive four months, can see large communities are formed, as shown in Figure 4-9(b-d); which affects the mechanism of discovering influencers in the structure of these communities.

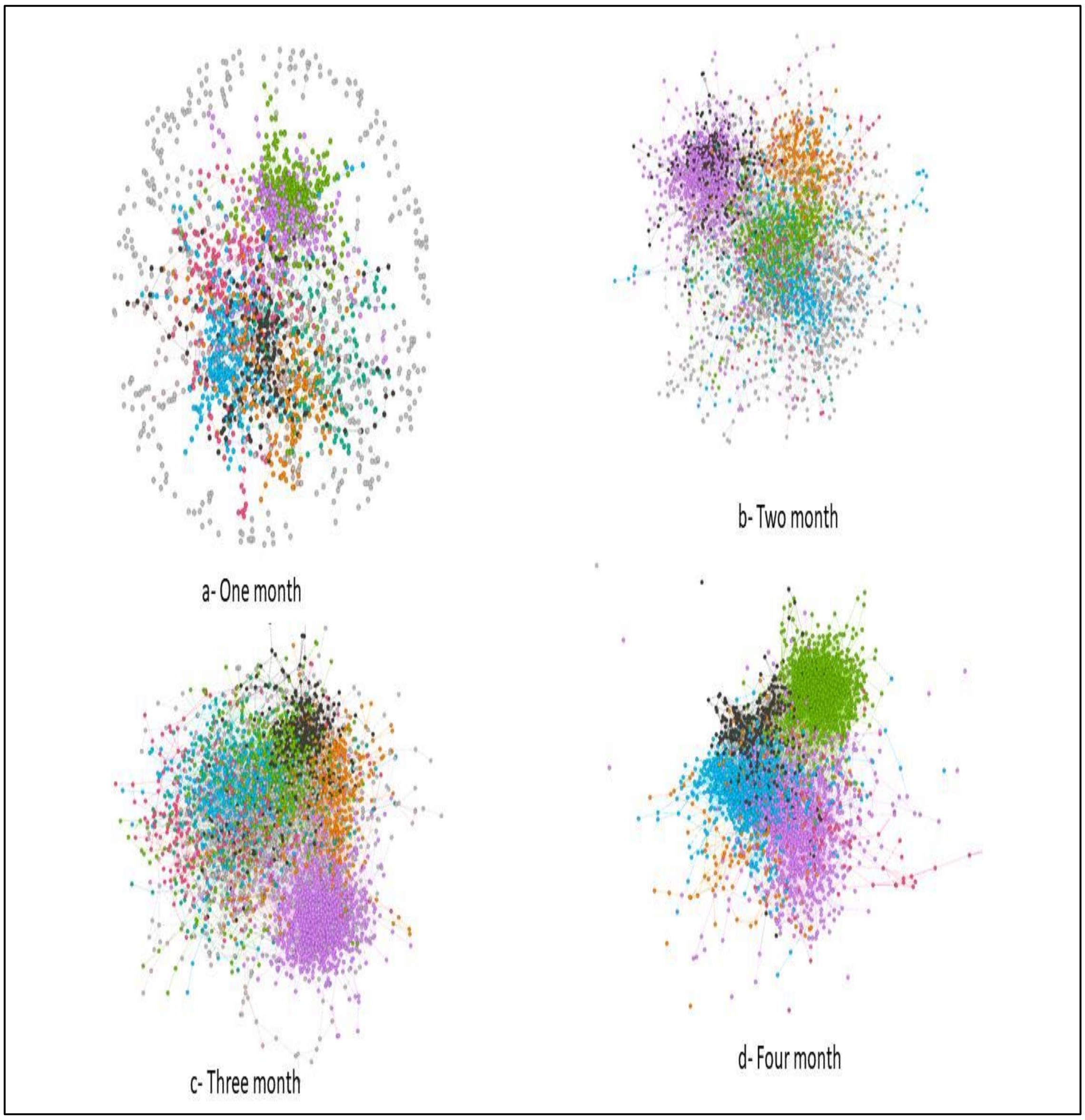


Figure 4-9 Community Detection with Different Periods for Facebook Snapshots

Figure 4-10 offers the stability of the Twitter network over time in the number of events of communities in each snapshot to some extent. The network is relatively stable because the number of events does not frequently vary over time.

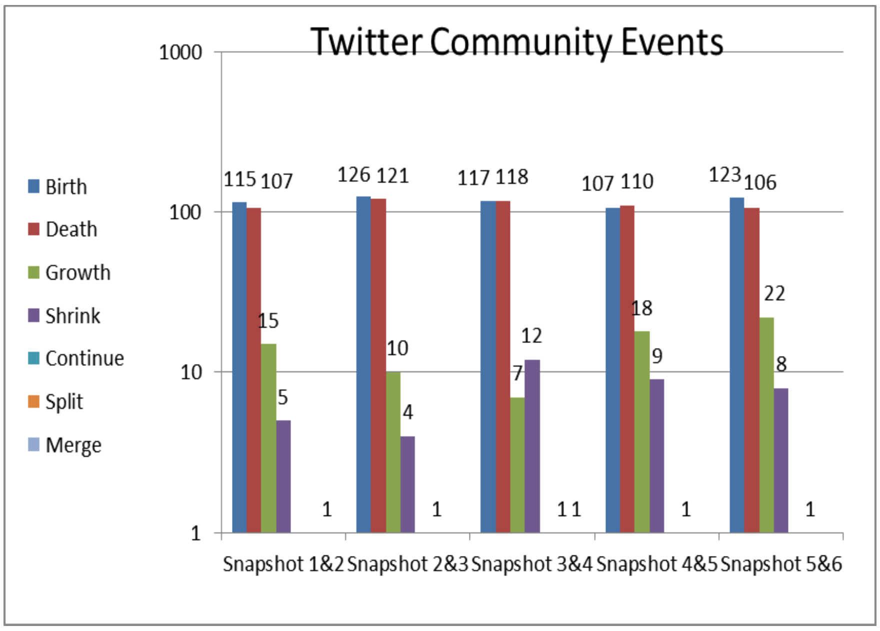


Figure 4-10 The Events Detected in Twitter Networks

Again, the experiments have been conducted to determine the snapshot period for the Twitter network and choose the most appropriate snapshot time among them, as selecting the proper period for the snapshot is one of the most critical challenges. When comparing three different periods, we get the average number of communities shown in Figure 4-11.

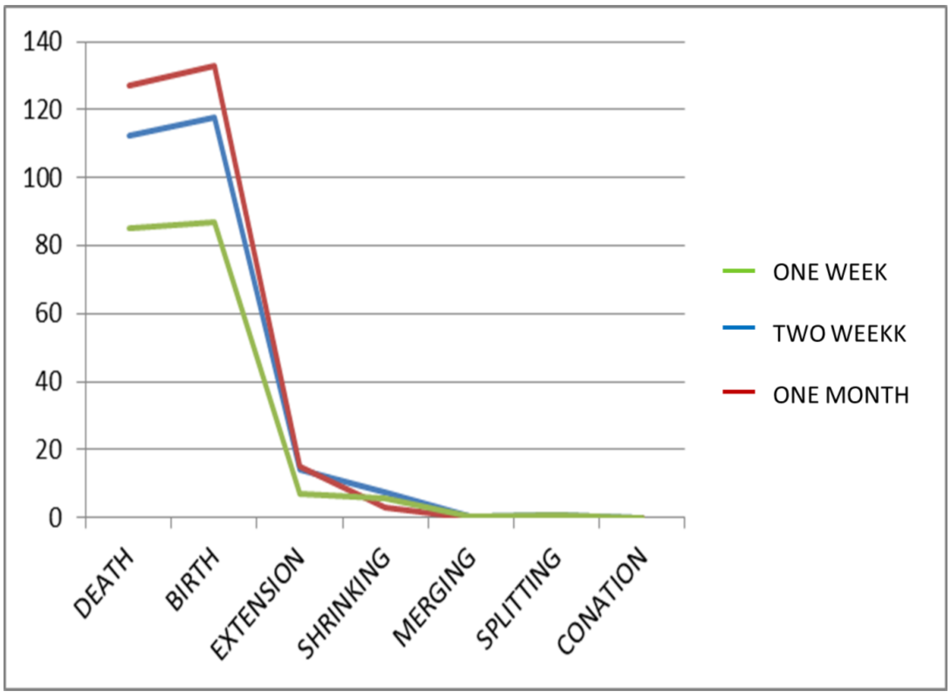


Figure 4-11 The Average Number of Communities in Each Event over Three Periods for Twitter Dataset.

We note that most events are apparent when choosing two weeks, while some events such as the division and merging of communities disappear during one month. In the case of one week, the possibilities of extension and shrinking decrease dramatically, and most of them are in tiny communities.

The similarity threshold is a crucial parameter in events detection of the dynamic community since it limits the algorithm's accuracy by defining the maximum amount of noise that can be tolerated. Figure 4-12 and Figure 4-13 depict the number of Contraction and Splitting events discovered in the Twitter and Facebook datasets at various similarity threshold settings.

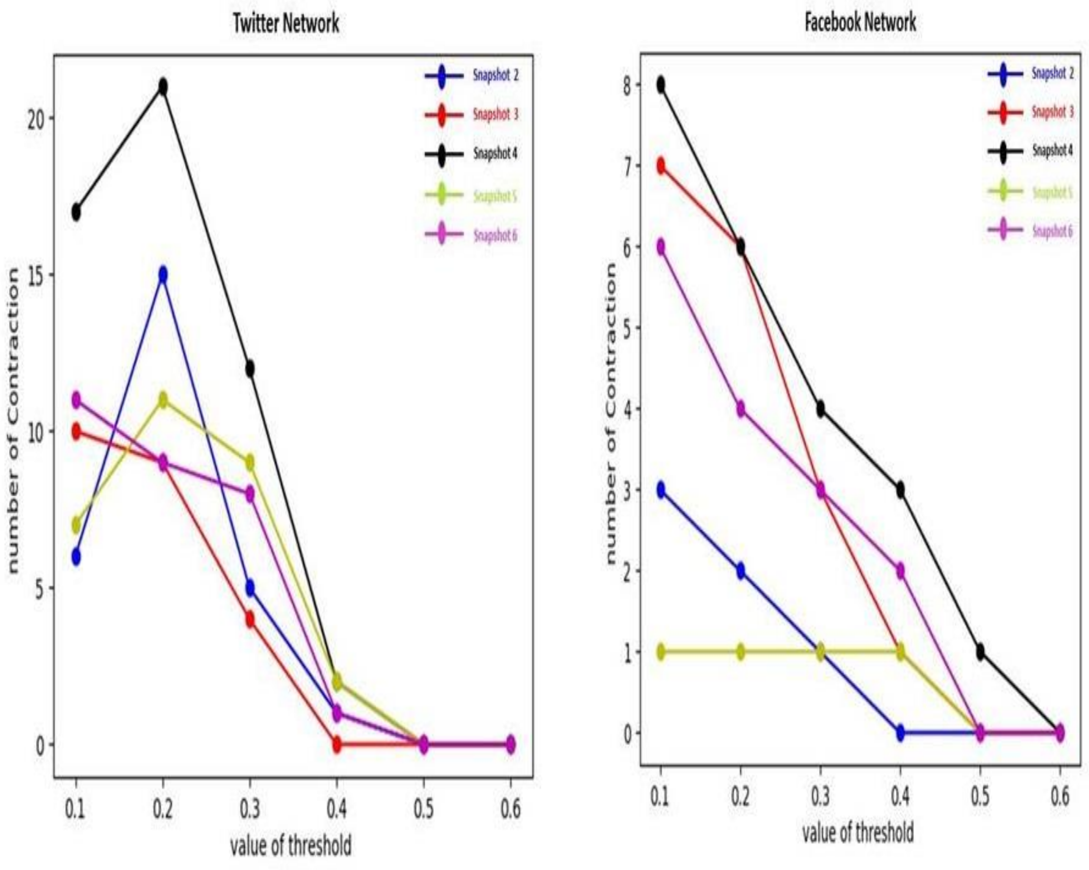


Figure 4-12 Number of Contraction Events with Varied Threshold Values

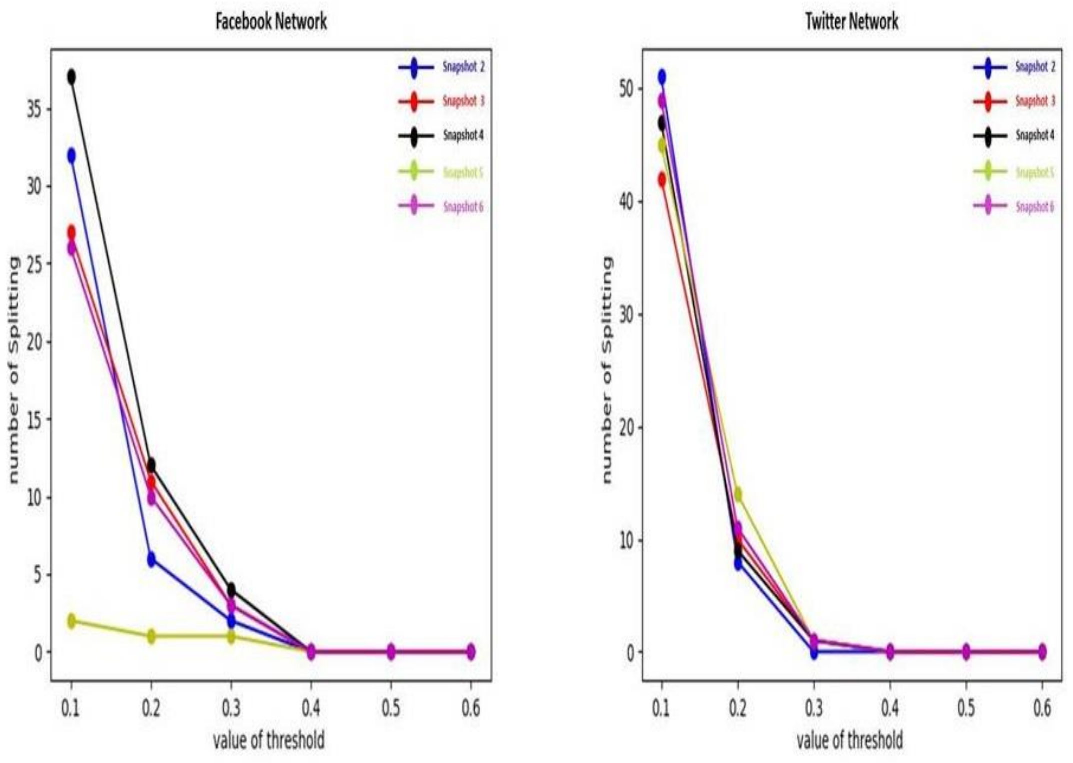


Figure 4-13 Number of Splitting Events with Varied Threshold Values

The other two events extension and merging) follow a similar pattern as Figure 4-14, Figure 4-15.

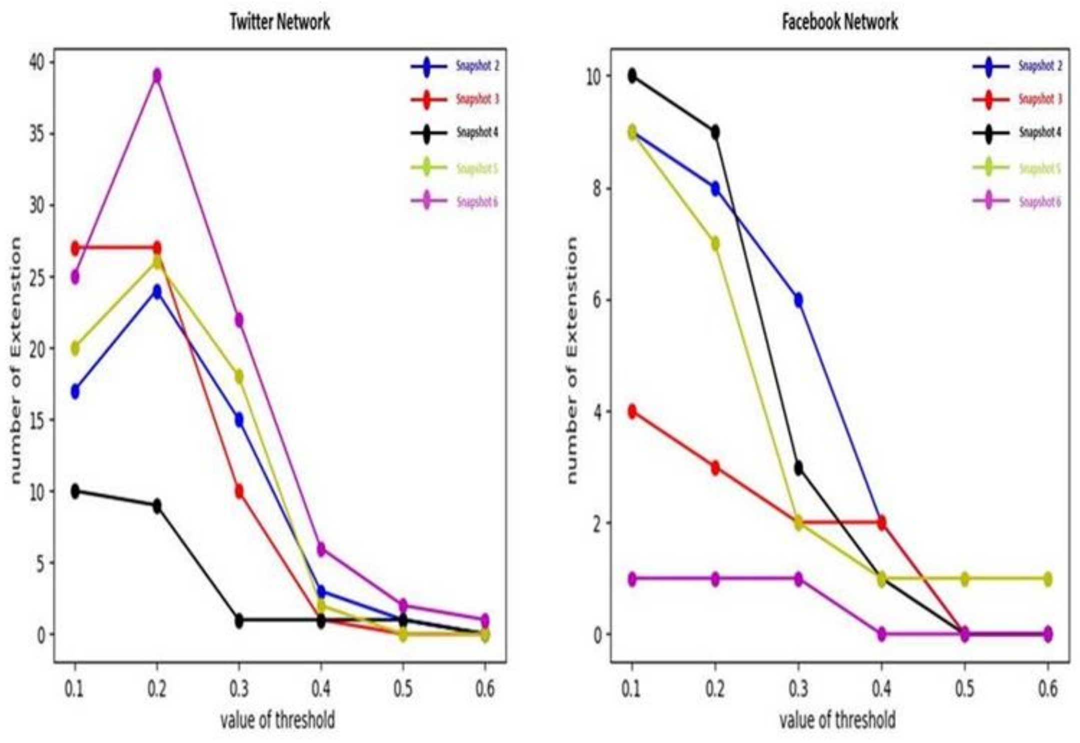


Figure 4-14 Number of Extension Events with Varied Threshold Values

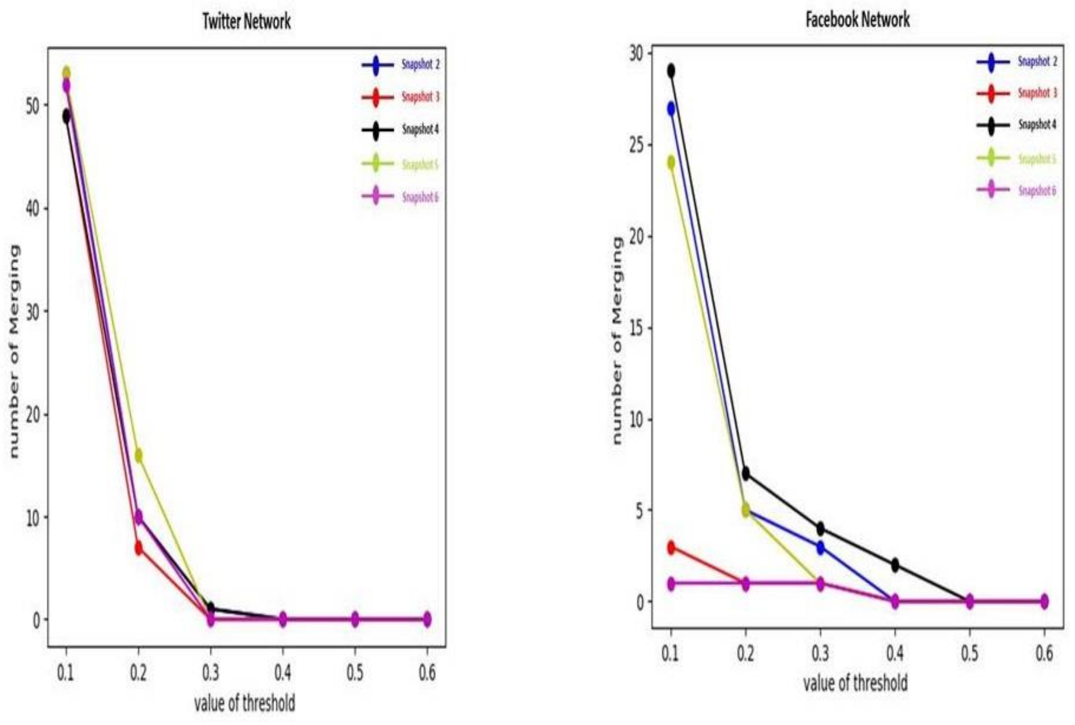


Figure 4-15 Number of Merging Events with Varied Threshold Values

From Figure 4-12 to Figure 4-15 show that the number of identified events drops as the similarity threshold is raised. When it is more than 0.3, the number of observed events decreases dramatically, and just a few events can be recognized. This means that few people communicate in real-world networks.

On the other hand, because were interested in the network's overall structural changes, a certain amount of noise is acceptable, which implies the threshold can be smaller than 0.6 in practice. When a similarity threshold is set low, a lot of noise may be accepted, and more events can be identified. As a result, the similarity threshold value can be adjusted anywhere between 0 and 1 depending on how much noise is acceptable. This study uses a 0.3 threshold to provide a clear picture of the evolution events.

The structure of communities’ changes over time, affected by the interaction of individuals with each other, and the events reflect the life cycle of the community. A new community may be born in the current snapshot, and after a period of time (in the next snapshot), this community's growth and the number of its members increase, or vice versa, this community shrinks and loses some of its elements, all of which depends on the interaction of individuals with each other in this snapshot. Table 4-2 and Table 4-3 reflect the behavior of events across snapshots in the Facebook and Twitter datasets respectively, where each snapshot represents the number of times events appear in this period.

Table 4-2 The Number of Events with Facebook Dataset Snapshots

Date of each snapshot	The events of community					
	Birth	Death	Growth	Shrink	Split	Merge
14/1/2007 to 14/5/2007	18	22	6	1	2	0
14/5/2007 to 14/9/2007	2097	23	0	2	3	0
14/9/2007 to 14/1/2008	30	2098	6	0	1	0
14/1/2008 to 14/5/2008	24	28	8	2	0	0
14/5/2008 to 14/9/2008	4041	30	0	1	2	0

Table 4-3 The Number of Events with Twitter Dataset Snapshots

Date of each snapshot	The events of community					
	Birth	Death	Growth	Shrink	Split	Merge
15/10/2010 to 1/11/2010	115	107	15	5	0	1
1/11/2010 to 15/11/2010	126	121	10	4	1	0
15/11/2010 to 1/12/2011	117	118	7	12	1	1
1/12/2010 to 15/12/2010	107	110	18	9	1	0
15/12/2010 to 1/1/2011	123	106	22	8	1	0

4.4 Discussion Tracking Influence Nodes for Each Community

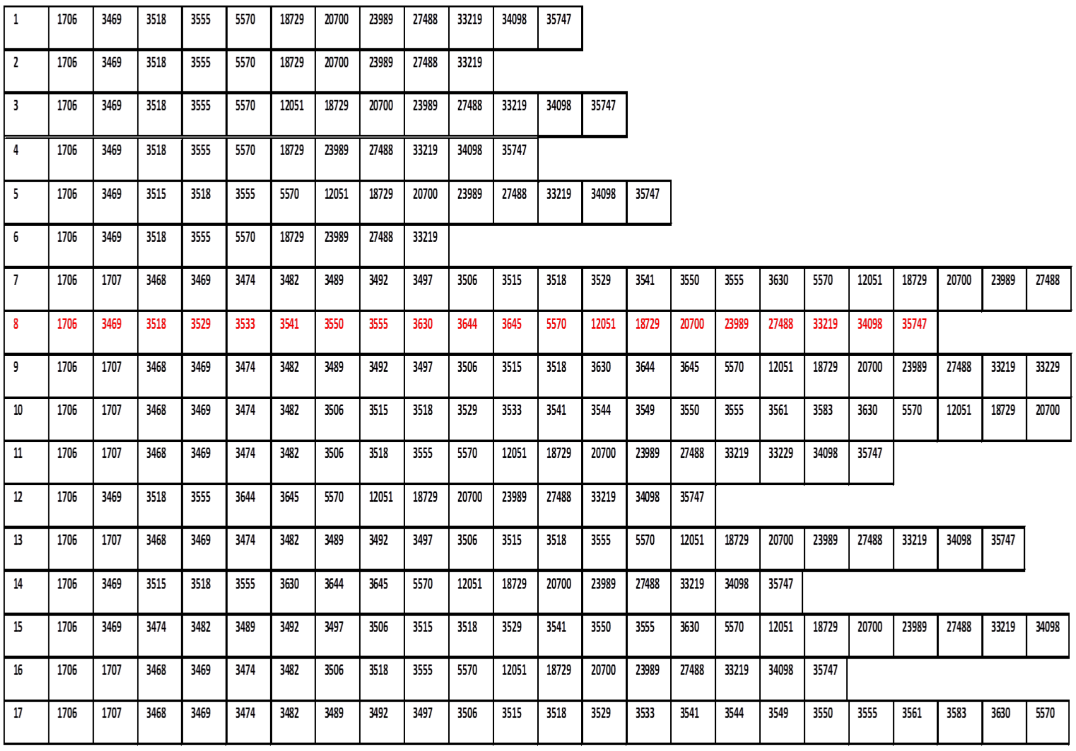
The detected communities and the events they experienced were considered when the NSGAII-IM algorithm was implemented in each community with two conflicting objectives. The first objective is influence maximization, and the second objective is to reduce the number of seeds. A series of steps were carried out to achieve these two opposing goals, and the outcome of each step is explained in subsequent sections.

4.4.1 Analysis of Explore Pareto Front from Community

Pareto Fronts are considered to be the influential users in the current community who have implemented two conflicting objectives of the NSGAII algorithm. Furthermore, the events that happened to this community can be traced from the previous snapshot and used the Pareto Front as nominees for the community in the current snapshot.

The output of the NSGAII-IM algorithm in each snapshot is Pareto Fronts. As a matter of fact, Pareto Fronts are the proposed solutions with two objectives so that Pareto Fronts have different lengths. For example, Figure 4-16 represents the sample of the result afterward applied NSGAII-IM algorithm in the first snapshot for the Twitter dataset and displays a set of Pareto Fronts of community 92.

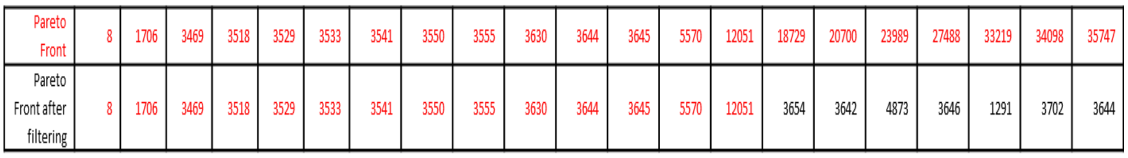
In the current snapshot (2nd snapshot), we match community 1 using Jaccard similarity Equation (2-5) with all previous snapshot communities. Given these matches, it is seen that community 92 exceeds the matching threshold with community 1, in which the former shrank by more than 10% to produce the latter. It is possible to take advantage of this Pareto Front and include them as pre-existing individuals when forming the initial population in community 1.



1	1706	3469	3518	3555	5570	18729	20700	23989	27488	33219	34098	35747											
2	1706	3469	3518	3555	5570	18729	20700	23989	27488	33219													
3	1706	3469	3518	3555	5570	12051	18729	20700	23989	27488	33219	34098	35747										
4	1706	3469	3518	3555	5570	18729	23989	27488	33219	34098	35747												
5	1706	3469	3515	3518	3555	5570	12051	18729	20700	23989	27488	33219	34098	35747									
6	1706	3469	3518	3555	5570	18729	23989	27488	33219														
7	1706	1707	3468	3469	3474	3482	3489	3492	3497	3506	3515	3518	3529	3541	3550	3555	3630	5570	12051	18729	20700	23989	27488
8	1706	3469	3518	3529	3533	3541	3550	3555	3630	3644	3645	5570	12051	18729	20700	23989	27488	33219	34098	35747			
9	1706	1707	3468	3469	3474	3482	3489	3492	3497	3506	3515	3518	3630	3644	3645	5570	12051	18729	20700	23989	27488	33219	33229
10	1706	1707	3468	3469	3474	3482	3506	3515	3518	3529	3533	3541	3544	3549	3550	3555	3561	3583	3630	5570	12051	18729	20700
11	1706	1707	3468	3469	3474	3482	3506	3518	3555	5570	12051	18729	20700	23989	27488	33219	33229	34098	35747				
12	1706	3469	3518	3555	3644	3645	5570	12051	18729	20700	23989	27488	33219	34098	35747								
13	1706	1707	3468	3469	3474	3482	3489	3492	3497	3506	3515	3518	3555	5570	12051	18729	20700	23989	27488	33219	34098	35747	
14	1706	3469	3515	3518	3555	3630	3644	3645	5570	12051	18729	20700	23989	27488	33219	34098	35747						
15	1706	3469	3474	3482	3489	3492	3497	3506	3515	3518	3529	3541	3550	3555	3630	5570	12051	18729	20700	23989	27488	33219	34098
16	1706	1707	3468	3469	3474	3482	3506	3518	3555	5570	12051	18729	20700	23989	27488	33219	34098	35747					
17	1706	1707	3468	3469	3474	3482	3489	3492	3497	3506	3515	3518	3529	3533	3541	3544	3549	3550	3555	3561	3583	3630	5570

Figure 4-16 Pareto Front of Community 92

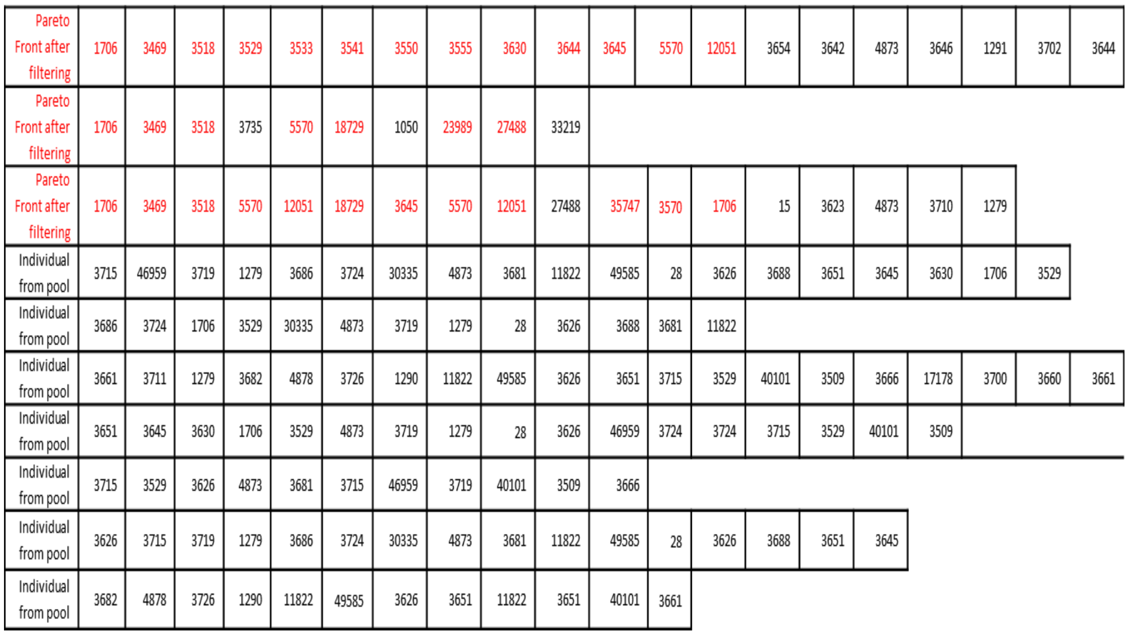
When the NSGAII-IM algorithm is implemented in community 1, the initial population must have Pareto Front from community 92. Pareto Front may have users who have left the current community, which means they should be filtered as shown in Figure 4-17.



Pareto Front	8	1706	3469	3518	3529	3533	3541	3550	3555	3630	3644	3645	5570	12051	18729	20700	23989	27488	33219	34098	35747
Pareto Front after filtering	8	1706	3469	3518	3529	3533	3541	3550	3555	3630	3644	3645	5570	12051	3654	3642	4873	3646	1291	3702	3644

Figure 4-17 Filtering the Pareto Front of Community 92

Thus, the rest of the individual can be supplemented by the pool, as shown in Figure 4-18, where the eighth Pareto Front of community 92 is with red IDs before filtering, as well as after filtering, the individual IDs are black and red, and the black IDs are the IDs of nodes provided by the pool. We include the Pareto fronts after filtering as individuals in the initial population when applying the NSGAII-IM algorithm to community 1, and the rest of the individuals are selected from the pool with different lengths as seen in a sample of individuals of community 1 in Figure 4-18.



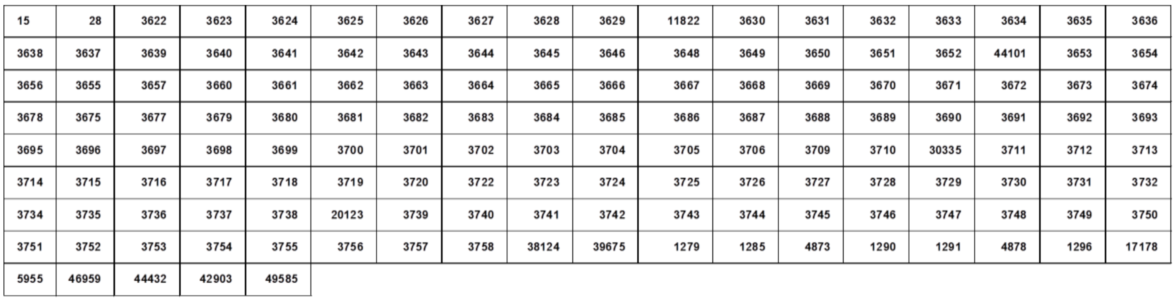
Pareto Front after filtering	1706	3469	3518	3529	3533	3541	3550	3555	3630	3644	3645	5570	12051	3654	3642	4873	3646	1291	3702	3644
Pareto Front after filtering	1706	3469	3518	3735	5570	18729	1050	23989	27488	33219										
Pareto Front after filtering	1706	3469	3518	5570	12051	18729	3645	5570	12051	27488	35747	3570	1706	15	3623	4873	3710	1279		
Individual from pool	3715	46959	3719	1279	3686	3724	30335	4873	3681	11822	49585	28	3626	3688	3651	3645	3630	1706	3529	
Individual from pool	3686	3724	1706	3529	30335	4873	3719	1279	28	3626	3688	3681	11822							
Individual from pool	3661	3711	1279	3682	4878	3726	1290	11822	49585	3626	3651	3715	3529	40101	3509	3666	17178	3700	3660	3661
Individual from pool	3651	3645	3630	1706	3529	4873	3719	1279	28	3626	46959	3724	3724	3715	3529	40101	3509			
Individual from pool	3715	3529	3626	4873	3681	3715	46959	3719	40101	3509	3666									
Individual from pool	3626	3715	3719	1279	3686	3724	30335	4873	3681	11822	49585	28	3626	3688	3651	3645				
Individual from pool	3682	4878	3726	1290	11822	49585	3626	3651	11822	3651	40101	3661								

Figure 4-18 Sample of an Initial Population of Community 1

4.4.2 Creating Influencer Pool

The initial population of the NSGAII-IM algorithm consists of several individuals. Each individual is a group of nodes belonging to the current network and considered a candidate solution (influencers set). After dividing the network into communities, the centrality for measures of degree, closeness, and eigenvector centrality is calculated for each node in the community, then choosing the nodes whose centrality is high based on the threshold.

A threshold is not possible to set a fixed value because the size of the communities changes in each snapshot. Therefore, the average centrality measure of each degree, closeness, and eigenvector centrality is the threshold. For example, when the node's degree is higher than its average degree, it is added to the pool. Similarly, the nodes with an eigenvector or (and) closeness centrality that passes the threshold are added to the pool once. Figure 4-19 presents the pool nodes of community 1 that are shrinking from community 92.



15	28	3622	3623	3624	3625	3626	3627	3628	3629	11822	3630	3631	3632	3633	3634	3635	3636
3638	3637	3639	3640	3641	3642	3643	3644	3645	3646	3648	3649	3650	3651	3652	44101	3653	3654
3656	3655	3657	3660	3661	3662	3663	3664	3665	3666	3667	3668	3669	3670	3671	3672	3673	3674
3678	3675	3677	3679	3680	3681	3682	3683	3684	3685	3686	3687	3688	3689	3690	3691	3692	3693
3695	3696	3697	3698	3699	3700	3701	3702	3703	3704	3705	3706	3709	3710	30335	3711	3712	3713
3714	3715	3716	3717	3718	3719	3720	3722	3723	3724	3725	3726	3727	3728	3729	3730	3731	3732
3734	3735	3736	3737	3738	20123	3739	3740	3741	3742	3743	3744	3745	3746	3747	3748	3749	3750
3751	3752	3753	3754	3755	3756	3757	3758	38124	39675	1279	1285	4873	1290	1291	4878	1296	17178
5955	46959	44432	42903	49585													

Figure 4-19 The Pool Nodes

4.4.3 Applying NSGAII-IM Algorithm to choose the Influencer Nodes

The NSGAII-IM algorithm had to be adapted to optimize the influence maximization problem according to an improved model based on two opposing objectives. The hidden additional information within the network structure (degree, closeness, eigenvector) can be used when feasible and benefit from the events that the community has experienced from the previous snapshot. Using the modified NSGAII-IM algorithm, the influence maximization problem can be improved by following the steps in Algorithm 3-4. The algorithm selects a set of influential individuals of different lengths (known as the Pareto Front) based on a graph of the detected community after a series of steps that will be remembered. When using NSGAII-IM algorithm, make sure that the public parameters are set as shown in Table 4-2 for all results.

Table 4-2 NSGAII-IM Algorithm Parameters

Property	Symbol	Value
Population size	N	10% of size community
Number of generations	G	100,500
Length of individual	K	Variable size
Mutation probability	P_m	0.1
Crossover probability	P_c	0.9

4.4.3.1 Individuals Representation

The first step of the NSGAII-IM algorithm with the proposed scenario is to represent population consisting of individuals. The individual representing the candidate solution can be described as a set of nodes (seeds). The size of the initial population is about 10% of the total size of each community. The individuals are characterized by a variable length instead of a constant, depending on the number of ID nodes inside the initial individuals, which vary in length from 5% to 7% of the community, to meet the two contradictory aims of an algorithm.

The individuals who make up the initial population are individuals who were imported from a community in a previous snapshot with new individuals or may only new ones depending on the nature of the event. Suppose the current community has gone through events such as Growth, Contraction, Merge, and Split from the previous snapshot.

In that case, the influential individuals (Pareto Front) of that community are imported to be individuals who fall within the initial population of the current community after deleting the non-existing nodes and adding some nodes to match the length of an individual with the size of the present community.

The nodes that make up the individual are chosen randomly from the pool if the community appears for the first time in the current snapshot (birth event), or if the current snapshot is the initial snapshot, lastly if the community is going through an event and the imported Pareto Front count does not cover the size of the population. Algorithm 3-2 of representing individuals is explained in Figure 4-20.

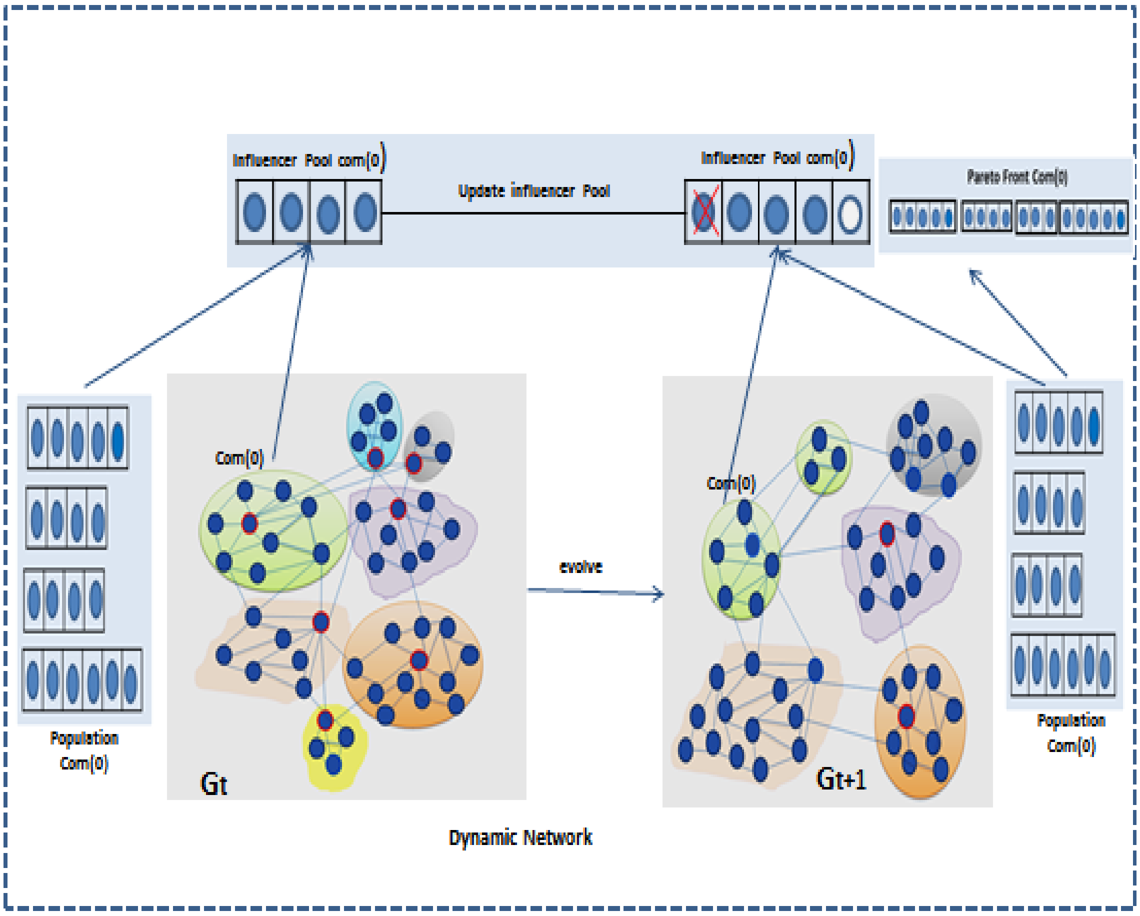


Figure 4-20 The Mechanism of Individuals Representing

4.4.3.2 Evolutionary Operators

Crossover and mutation are the two evolutionary operators used to reproduce a new offspring population whose features have been inherited from those parent's population. The crossover and mutation operators occur when the random probability exceeds the crossover and mutation probability, respectively; the crossover probability chosen in this work is 0.9, while the mutation probability value is 0.1.

4.4.3.3 Fitness Function

The first fitness seeks to keep the number of nodes in a seed set as low as possible, whereas the second fitness function increases the influence of the identified seed set across the network. It is worth mentioning that the diffusion modeling by using the improved IC model has been implemented as a second fitness function that will explain the results in Section 4.4.4 with different datasets.

For Algorithm 3-4, a comparison of the performance of NSGAII-IM algorithm has been made with the NSGAII algorithm and three of the following algorithms: high degree (Degree), Degree Discount (HIGHDEG), and Single Discount (SDISC), as demonstrated in Figure 4-21. It is worth mentioning that the previous algorithms have been replicated to measure their performance on the same datasets that the NSGAII-IM algorithm applied on which. Additionally, the NSGAII-IM algorithm has been used on the whole network without community detection in these experiments to fit the comparison conditions.

Figure 4-21 shows the application of the algorithms to the Facebook-Wall posts and Twitter datasets in two experiments; the number of generations in the first is 100, whereas the second experiment has 500 generations, as presented in Figure 4-21a and Figure 4-21b. The X-axis shows the average influence spread, whereas the Y-axis indicates the number of seed nodes in the candidate solution. The candidate solution is represented in green, while the red color represents the Pareto Front found by the NSGAII-IM algorithm. The remaining colors represent the four methods.

The experiment has been provided with the results of the NSGAII algorithm and three heuristic algorithms, namely, Degree, HIGHDEG, and SDISC versus NSGAII-IM algorithm. In Figure 4-21a, the evolution with 100 generations, the NSGAII algorithm outperformed three algorithms in some seed sizes and not in others in the Facebook network. In contrast, the NSGAII algorithm was able to superior three algorithms in the Twitter network, regardless of the size of the seed. As for Figure 4-21b, the evolution with 500 generations the NSGAII algorithm can outperform three algorithms in all seed sizes in both networks while maintaining the spread size versus reducing the seed size.

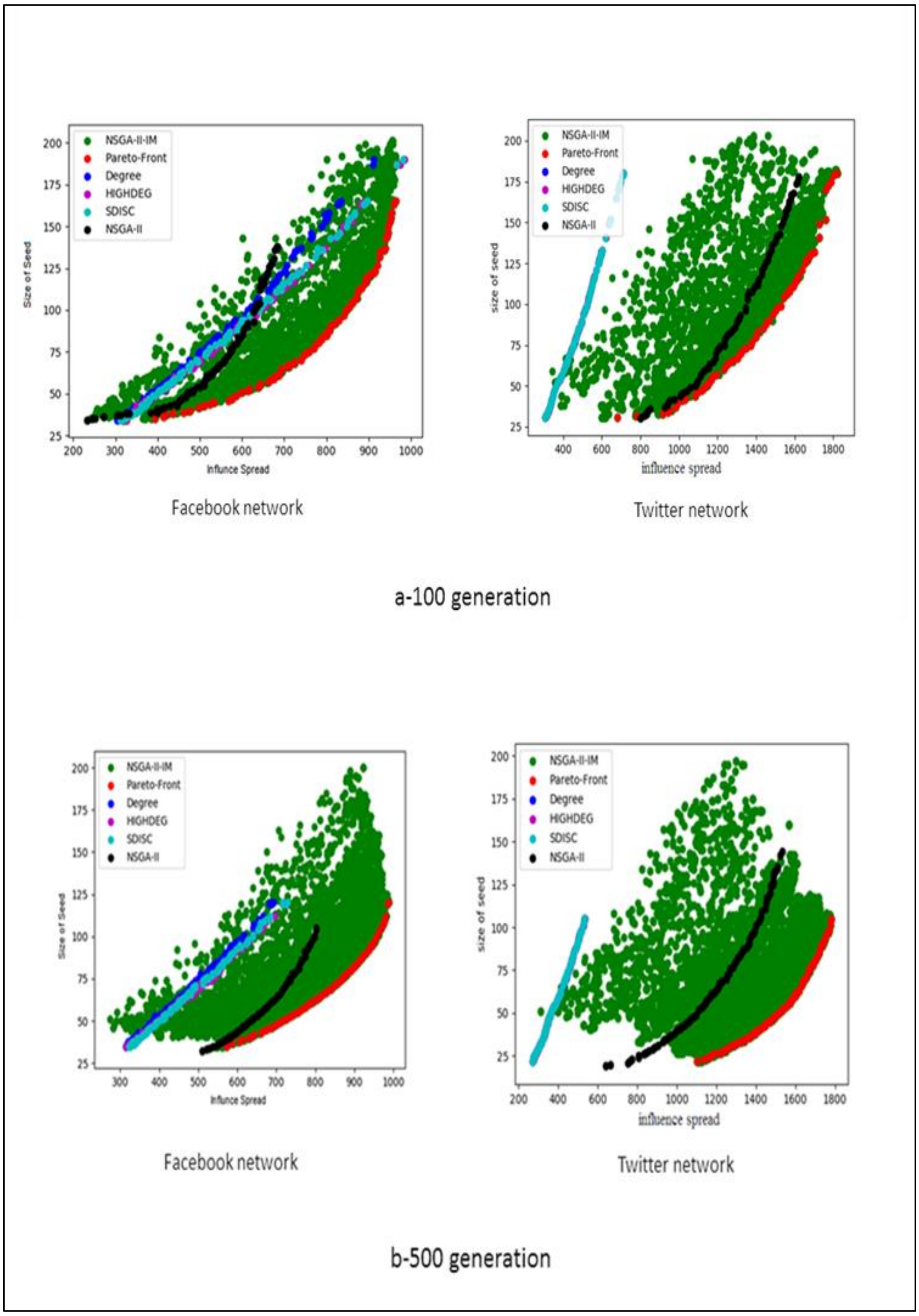


Figure 4-21 Experimental Results for the Benchmark Facebook Wall Post and Twitter networks.

Generally, the behavior of the NSGAII-IM algorithm in both experiments is superior to the other algorithms, including the baseline NSGAII algorithm. It is important to say the NSGAII-IM algorithm scores progress at 100 generations over the performance of the NSGAII algorithm at 500 generations in terms of the maximization of influence rate. In particular, the NSGAII algorithm excels the heuristics algorithms when the seed set (individual) is between 35-90, so that the black curve intersects with the other curves at other seed sets. In other words, this indicates that the algorithm is not better than heuristics algorithms with all seed sets. However, the NSGAII algorithm needs 500 generations to completely outperform the heuristic algorithms with all sizes of seed groups. At the same time, the proposed NSGAII-IM algorithm outperforms all methods with a percentage of 21% in the spread of influence through only 100 generations. Properly configuring the first generation leads to a faster convergence on the better Pareto Fronts (red curve).

In the Twitter network, as illustrated in Figure 4-21a, the NSGAII algorithm outperforms other methods in maximizing influence across all individual sizes, except for a limited number of individuals of tiny size, where NSGAII algorithm and NSGAII-IM algorithms perform equally well. The NSGAII-IM algorithm surpasses the NSGAII algorithm by 100 generations. It's worth noticing that the NSGAII algorithm in Figure 4-21b enhances its influence maximization performance according to 500 generations. With a smaller seed set size, the NSGAII-IM algorithm surpasses the NSGAII algorithm method in terms of influence rate; this is a significant improvement over the NSGAII. In comparison, the suggested NSGAII-IM algorithm outperforms all approaches by a factor of 10% in a spread of influence over only 100 generations.

A benchmark Digg network features a large network. The application of the NSGAII-IM algorithm is displayed in Figure 4-22 on the Digg network.

It is significant to state that the spreading size is greater than in the Facebook network, besides the fact that the algorithm outperformed the other algorithms with an influence spread of around 18% at 100 generations.

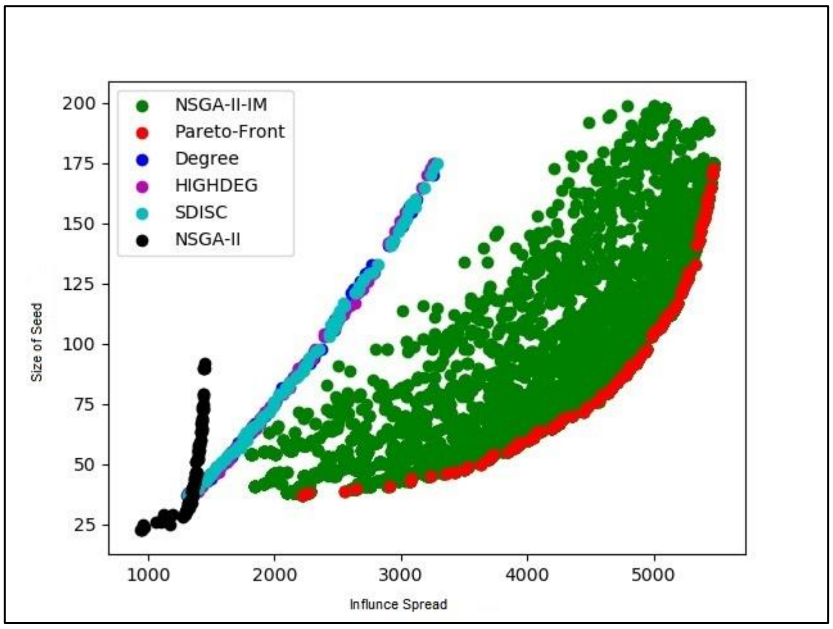


Figure 4-22 Experimental Results for the Benchmark Digg Network for 100 Generations.

Concerning the NSGAII algorithm, it outperforms the other heuristics ones at some sizes of the individuals (often small individual sizes), with the highest average impact spread of about 2000 affected nodes in the entire network. However, the NSGAII-IM algorithm outperforms all algorithms to reach the highest average spread of 5000 nodes.

4.4.4 Applying the WIC Model

The improved Independent Cascade model (i.e., WIC) has simulated the diffusion process. In this diffusion model, each edge must have a weight that acts as the degree of influence of the source node on the destination node, or it can be interpreted as a strength of the relationships between nodes that can lead to the effect.

These weights must first be learned to perform the propagation simulation. Mainly, estimating the weight of each edge in the social graph is the critical point in this stage; hence, it represents the degree of influence that can affect or change the other user's behaviour. In the experiment, the graph weights are calculated using only the structural-based feature Equation (2-1) or interaction feature only. Then by utilizing the Equation (3-1), the weights are calculated using structural and interaction-based features; as illustrated in Algorithm 3-5, the hybrid process between structural and interaction-based methods has been implemented as described in the results to be presented later.

This became necessary to emphasize the importance of the proposed diffusion model WIC and to compare it to the WC model. Figure 4-23 demonstrates the performance of the NSGAII-IM algorithm using both models of diffusion.

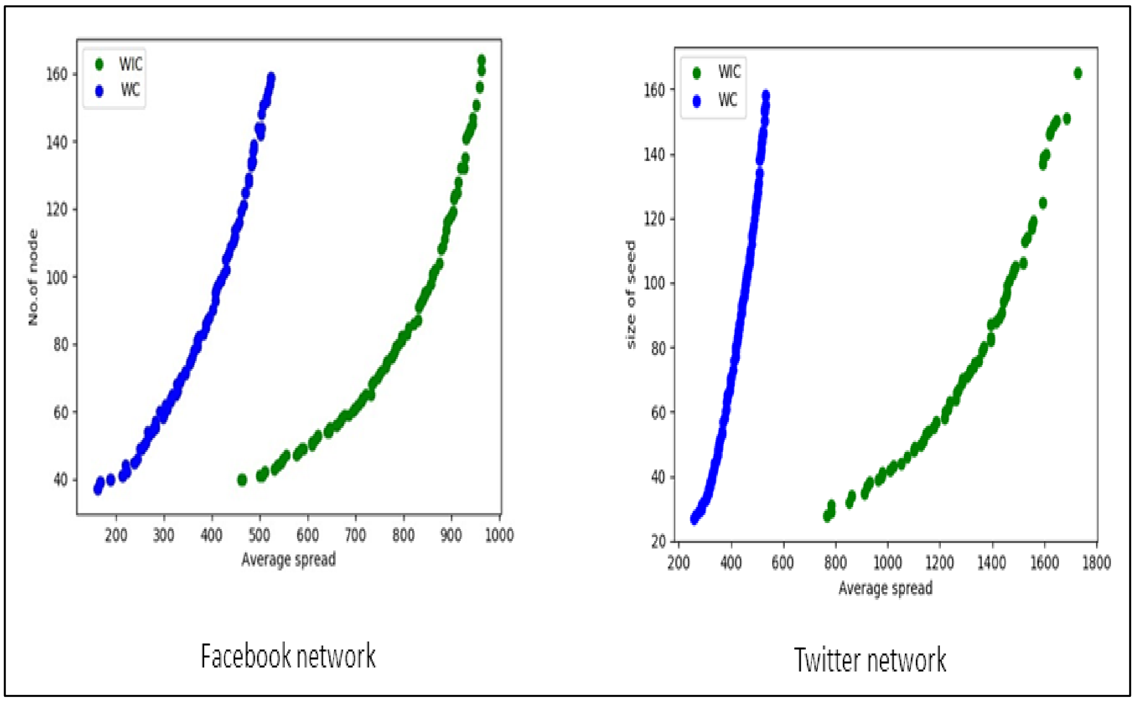


Figure 4-23 Influence Spreads of NSGAII-IM Algorithm for Wall Post-Facebook Dataset and Twitter Dataset under Two Different Diffusion

The X-axis presents the average influence spread, whereas the Y-axis indicates the number of seed nodes. It is clear that the latter algorithm with the proposed diffusion WIC model can improve the influence spread from 10% to 21% and from 3% to 10% in Facebook and Twitter networks, respectively if compared with the NSGAII-IM algorithm, but with the WC diffusion model.

Again, the performance of the NSGAII-IM algorithm in the Digg network has been evaluated through two different diffusion models. Obviously, the behaviour of the NSGAII-IM algorithm under the proposed diffusion model WIC showed an improved performance from 6% to 18% compared to the WC diffusion model, as can be noticed in Figure 4-24.

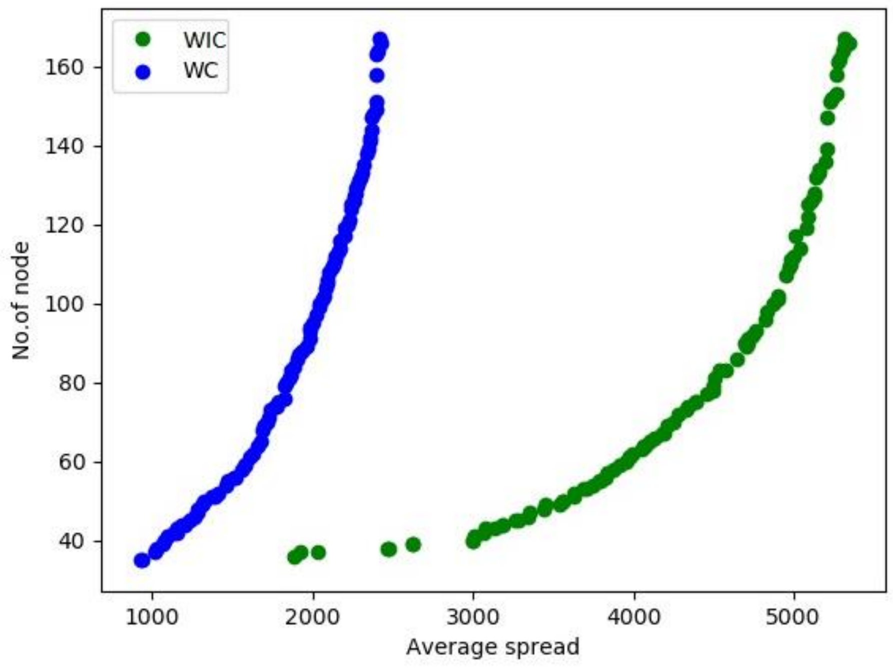


Figure 4-24 Influence Spreads of NSGAII-IM Algorithm for Digg Dataset under Two Different Diffusion

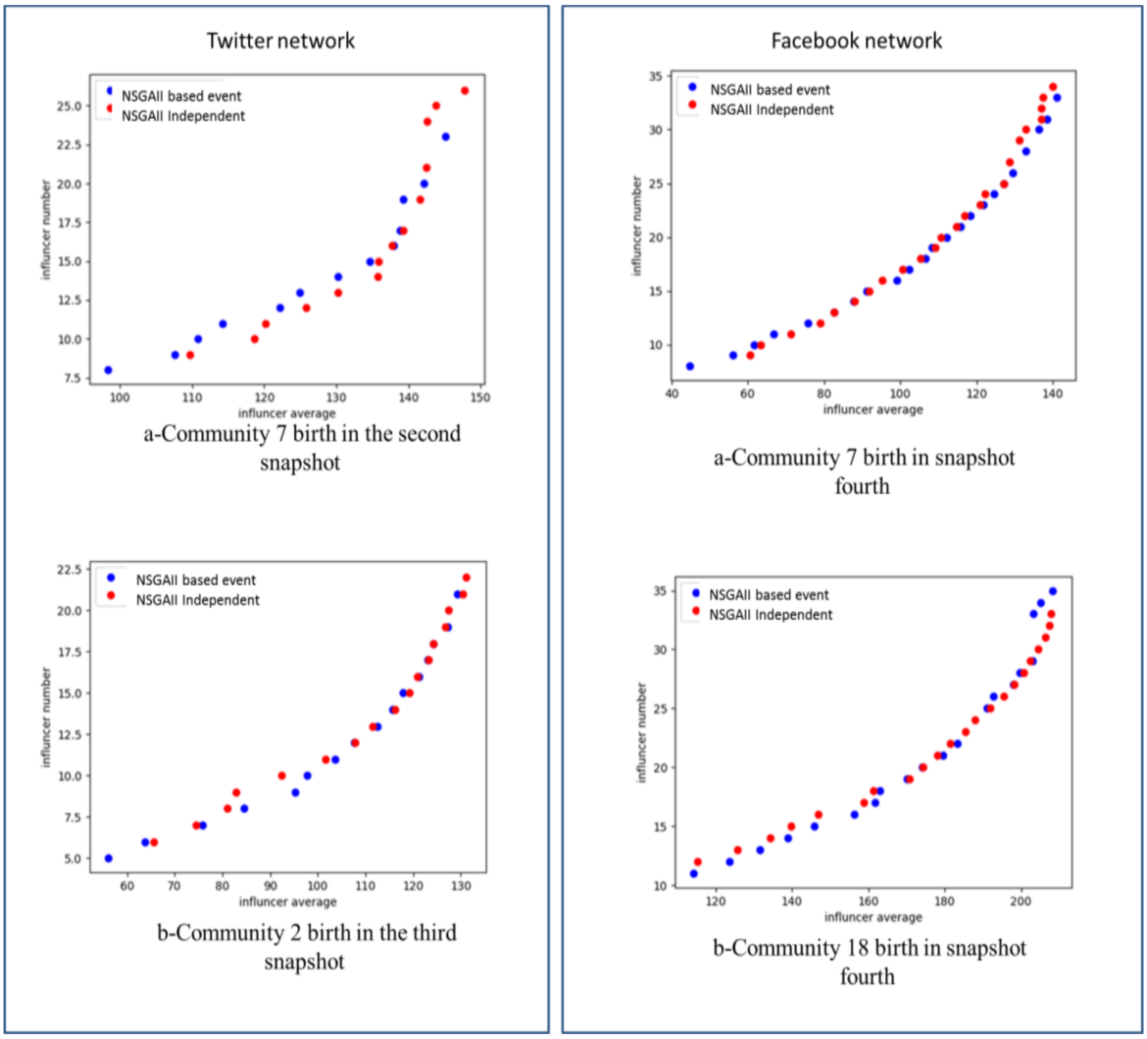
It is clear that the role of the proposed propagation model WIC cannot be ignored, as it is credited with the superiority of the algorithm when fixing the other conditions. Finally, the nature of the network affects the influence propagation rate. The Digg network is denser in nature than the wall post-Facebook and Twitter networks. Thus the interactions between nodes are more in a shorter period of time, which affects the nature of the influence diffusion.

4.4.5 Evaluate the performance NSGAII-IM Algorithm based on an Event of Community

The results obtained using the NSGAII-IM algorithm for each community in snapshot depended on the event of community in the previous snapshot by exploring the Pareto Front are compared with the same algorithm but without relying on the Pareto Front in the previous snapshot.

For each network, six snapshots represent the network's dynamic. For each snapshot, several communities go through a series of events over time. It is possible to benefit from the community's individuals nominated as influencers in the previous period, provided that this community is present in the current snapshot after going through one of these events; Growth, Shrink, Continue, Split, and Merge. Following is an explanation of the effect of each event on improving the NSGAII-IM algorithm's work.

The blue curve represents the performance of the algorithm when it is supported by an event, which is almost superior to the red curve representing the performance of the algorithm independently. As for the event of Birth and Death, it cannot be benefited from, because in the case of the delivery of a new community, it means its appearance for the first time, and it is not possible to import Pareto Front from a previous period, so when applying the two algorithms, the results are close, as in Figure 4-25 that represents the Twitter and Facebook networks. It should be mentioned that the preferred criterion here is the coverage size of influence:



a-Community 7 birth in the second snapshot

a-Community 7 birth in snapshot fourth

b-Community 2 birth in the third snapshot

b-Community 18 birth in snapshot fourth

Figure 4-25 NSGAII-IM based on Birth Event

Growth Event

Many communities are experiencing growth on Facebook networks and Twitter, as seen in the six snapshots. The following figures show the difference between using the NSGAII-IM algorithm, which is based on this Growth event, and benefits from the influencers of this community from the previous snapshot and using the NSGAII-IM algorithm, which is independent of the previous snapshot's influencers. This event appears across several communities in the six snapshots representing the dynamic Facebook network. Figure 4-26 shows the Growth event in the second, fourth and fifth snapshots, whereas this event does not appear in the third and sixth snapshots.

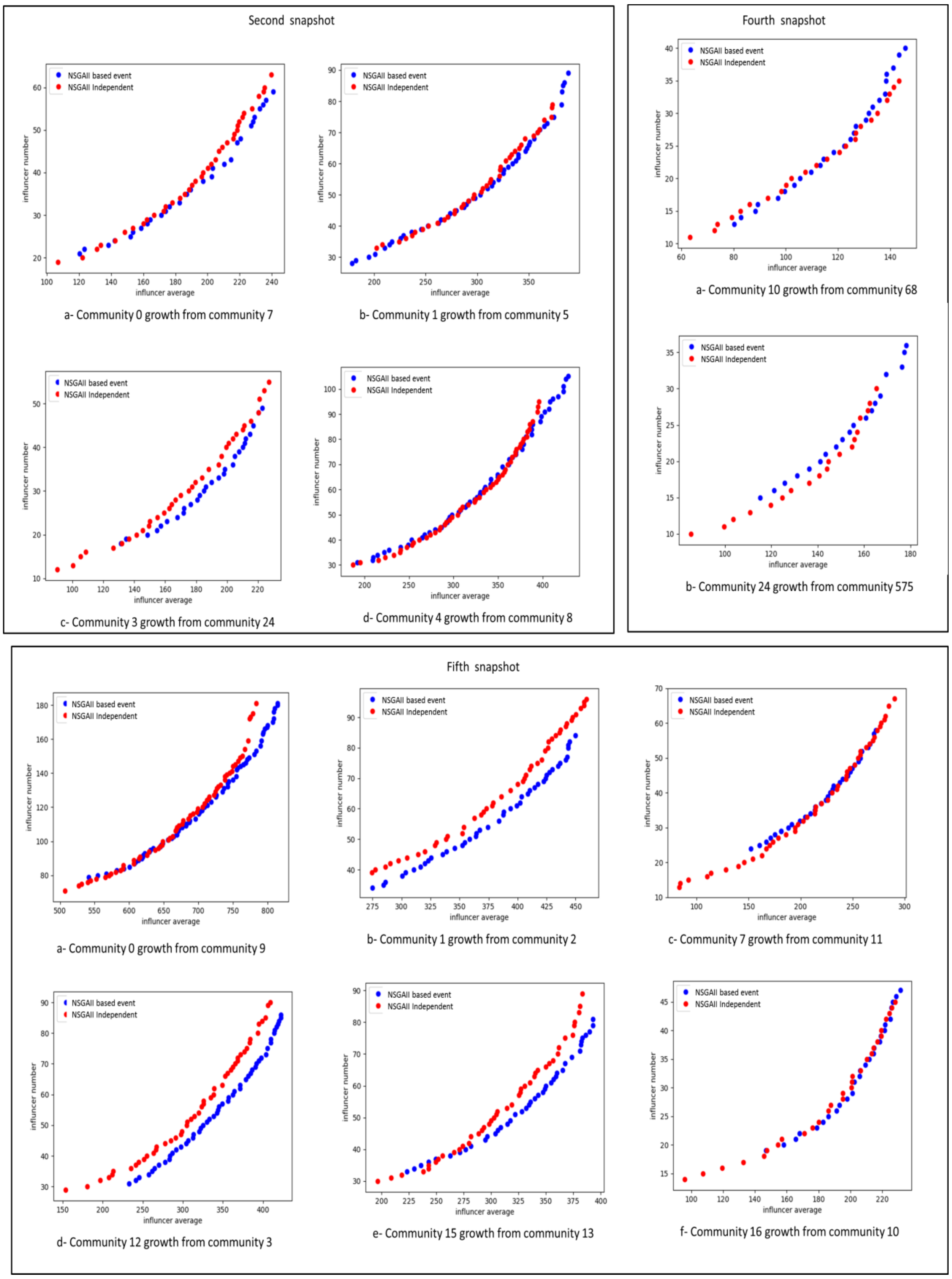
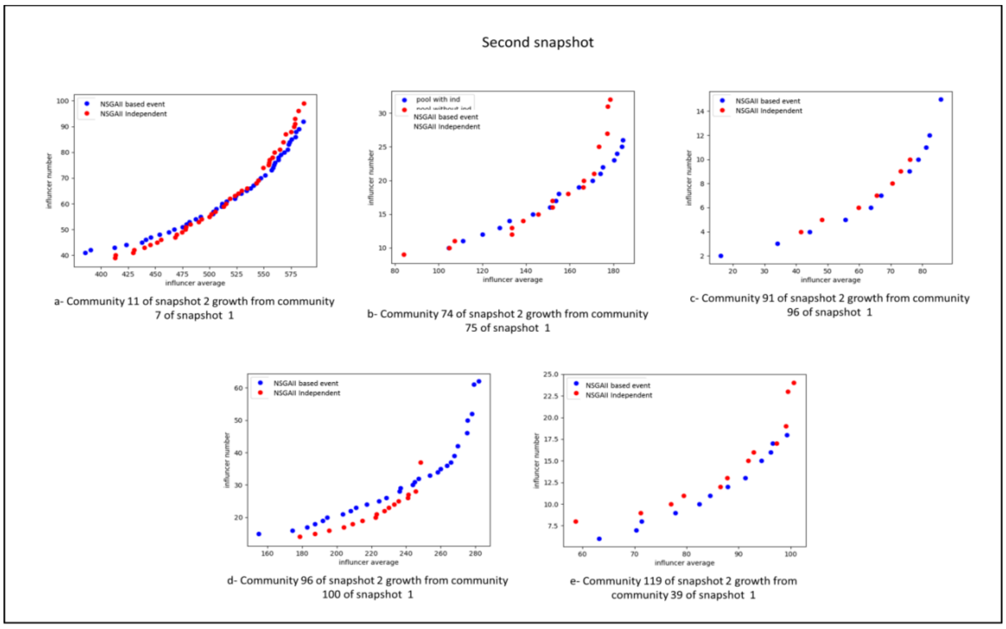


Figure 4-26 Pareto Front of a Facebook Dataset Using NSGAII-IM based on Growth Event Versus NSGAII-IM Independent

The Growth event appears in several communities in the six snapshots in the Twitter network. Figure 4-27 shows the Growth event in the second and third snapshots.



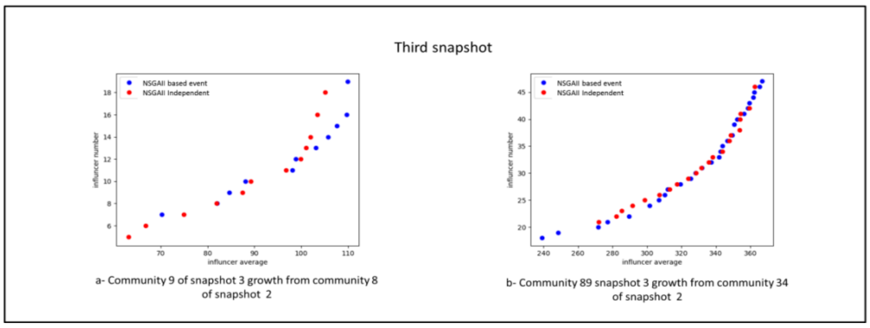


Figure 4-27 Pareto Front of a Twitter dataset using NSGAII-IM based on Growth Event Versus NSGAII-IM Independent

While Figure 4-28 illustrates the Growth events in the fourth snapshot.

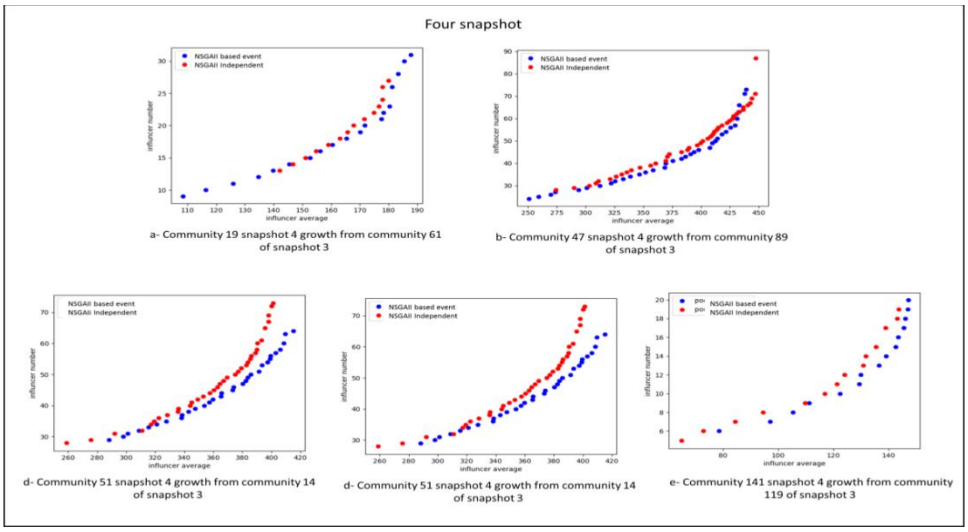


Figure 4-28 Pareto Front of a Twitter Dataset Using NSGAII-IM based on Growth Event Versus NSGAII-IM Independent in the Third and the Fourth Snapshots

As shown in Figure 4-29, the Growth event happens in the fifth snapshot.

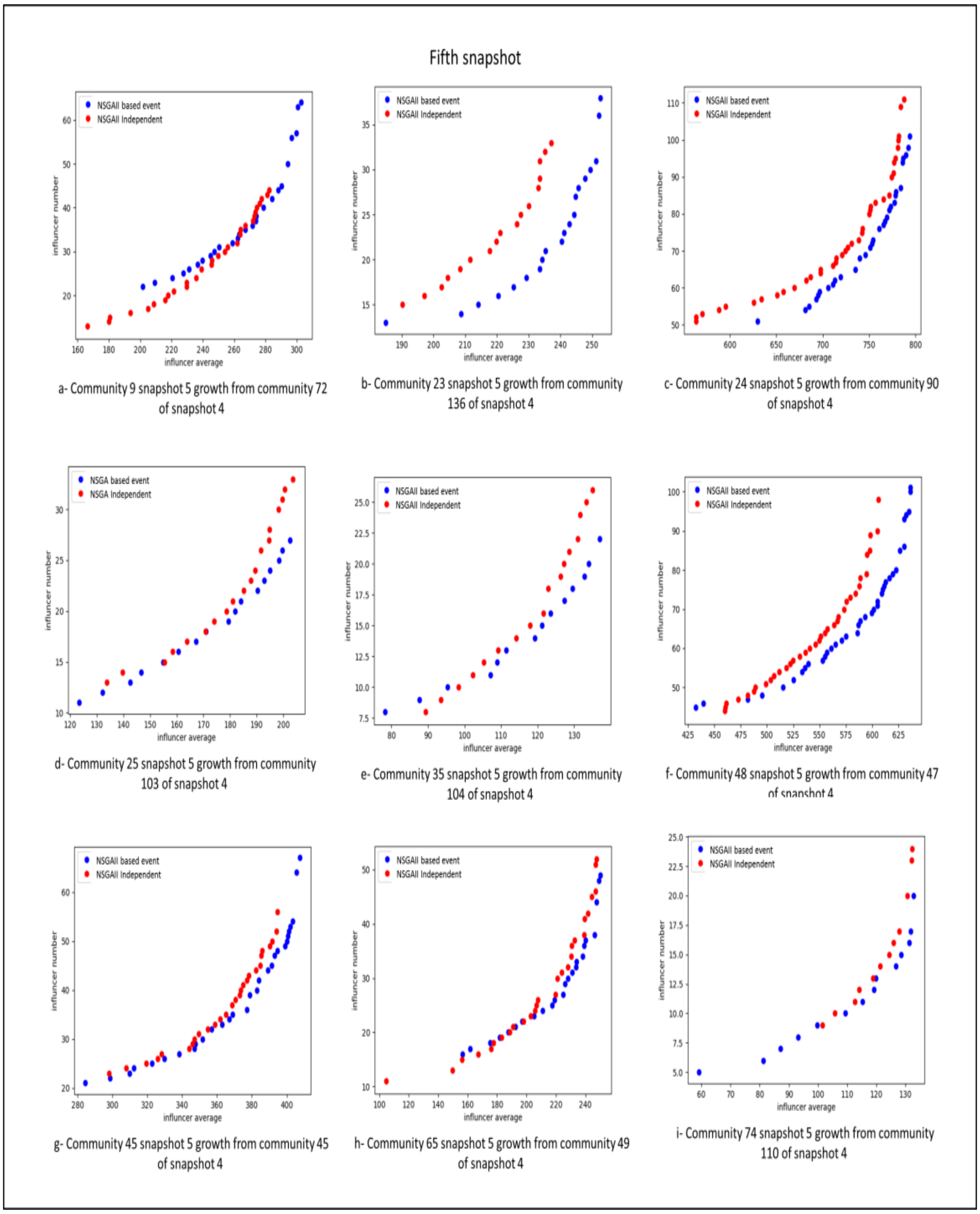


Figure 4-29 Pareto Front of a Twitter Dataset Using NSGAII-IM Based on Growth Event Versus NSGAII-IM Independent in the Fifth Snapshots

Finally, the Growth event occurs in the sixth snapshot, as demonstrated in Figure 4-30.

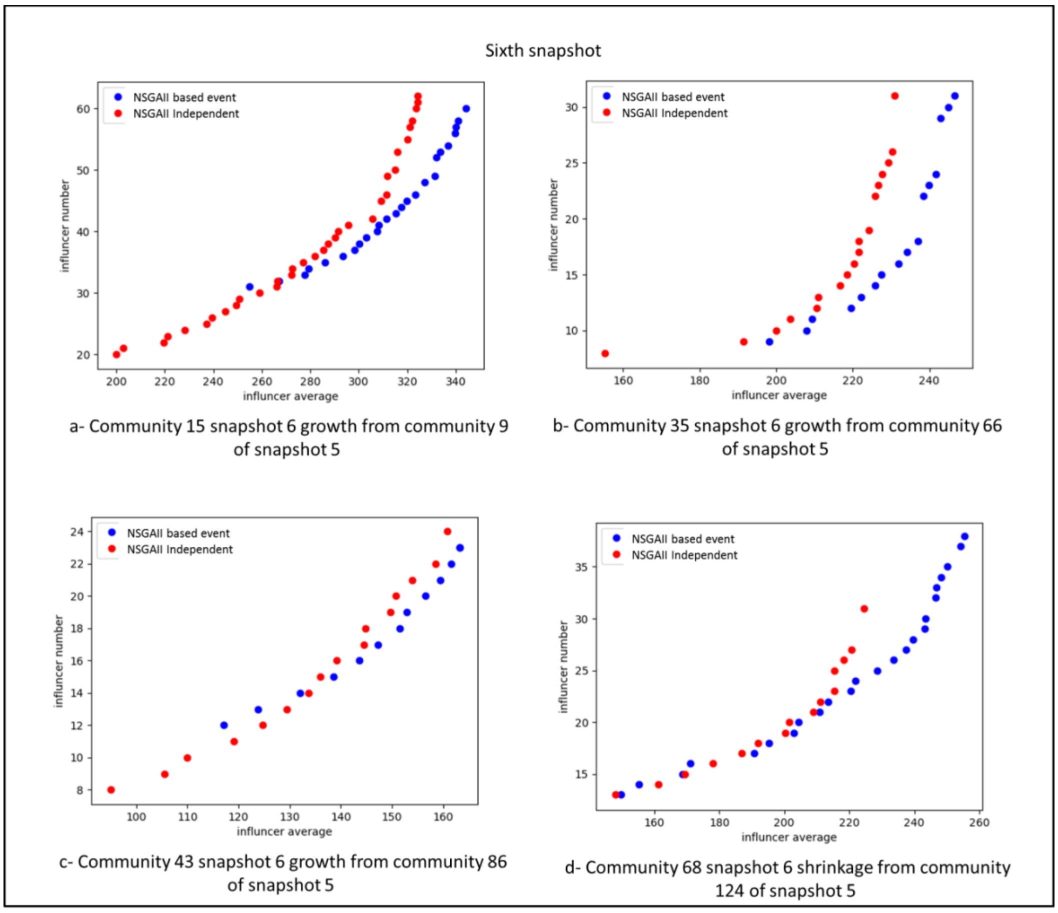


Figure 4-30 Pareto Front of a Twitter Dataset Using NSGAII-IM Based on Growth Event Versus NSGAII-IM Independent in the Sixth Snapshot

In general, it is clear from the previous figures for the two data sets show that implementing the NSGAII-IM algorithm with the Pareto Front earlier is superior to implementing the NSGAII-IM algorithm without Pareto Front. The results vary between good and very good in the snapshots, and in some cases, the performance is equal between the two algorithms; the size of the community could be the cause. When a community expands greatly, which means that a large number of new members belong to that community, the influence of influencers on the members of the expanding community may decrease.

Shrink Event

The six snapshots include many communities going through the event of shrinking on Facebook networks and Twitter. The following figures show the difference between using the NSGAII-IM algorithm, which is based on this shrinkage event, and benefiting from the influencers of this community from the previous snapshot and the NSGAII-IM algorithm, which does not depend on the influencers of the prior snapshot. This event occurs only in community 17 of the fifth snapshot in the Facebook network, since community 25 shrinks in the fourth snapshot, as shown in Figure 4-31.

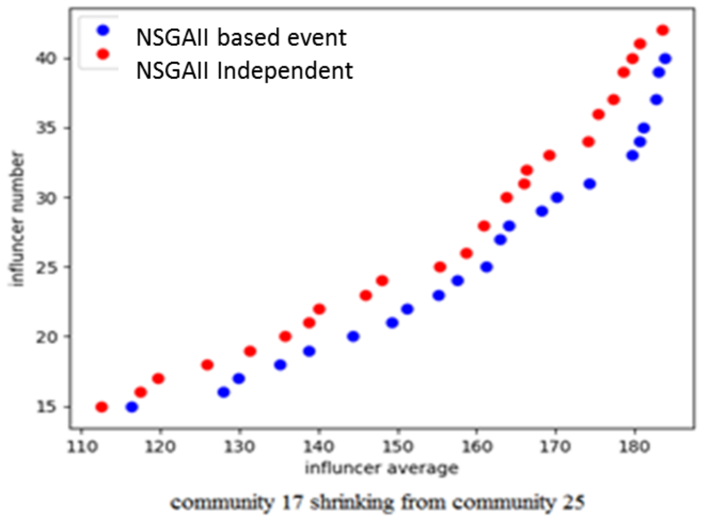


Figure 4-31 Pareto Front of a Facebook Dataset Using NSGAII-IM Based on Shrinkage Event Versus NSGAII-IM Independent in the Fourth Snapshot

The shrinkage event appears on the Twitter network across multiple snapshots. The following Figure 4-32 illustrates the shrinkage events in the different snapshots.

If a community undergoes a shrinkage event, it means that it loses many of its members, and these members are either influential or affected members. If the shrinkage of the community is large, the performance of the NSGAII-IM algorithm with Pareto Front can be equal NSGAII-IM algorithm without Pareto Front. This is because the imported members of the community in the previous snapshot are not present in the current community or have lost their influence over the community members the rest. But if the community is reasonably shrunk, importing the Pareto Front of the community from the previous snapshot contributes to improving the performance of the NSGAII-IM algorithm as it is noticeable in the previous figures in multiple snapshots and for both Facebook and Twitter datasets.

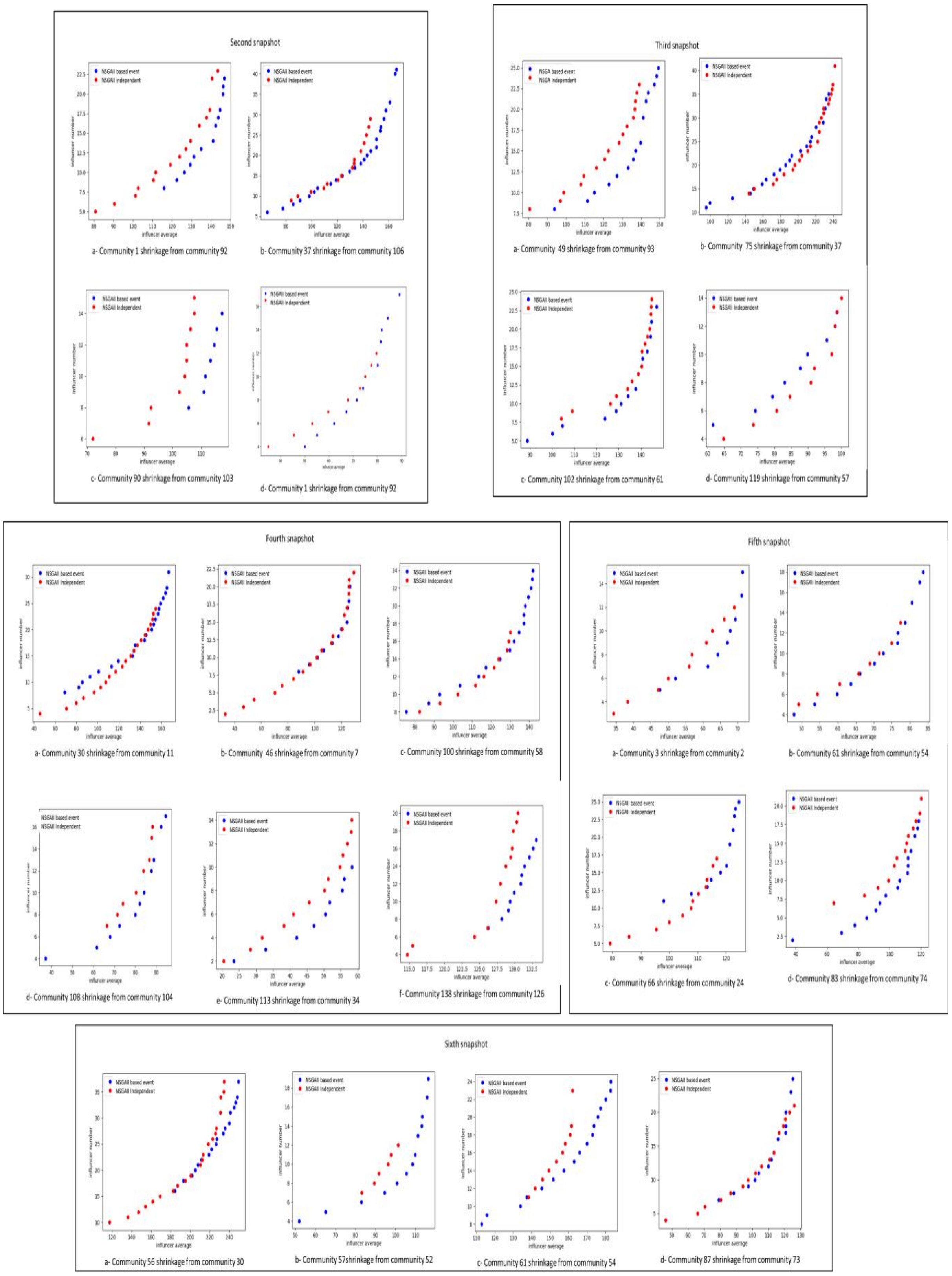


Figure 4-32 Pareto Front of a Twitter Dataset Using NSGAII-IM Based on Shrinkage Event Versus NSGAII-IM Independent in Different Snapshots

Splitting Event

Many communities, even if they are not very large, can be subject to the event of splitting into more than one community, and these splits appear in many communities for the networks of Facebook and Twitter. The differences between using the NSGAII-IM algorithm, which is based on this splitting event and benefits from the influencers of this community from the previous snapshot, and the NSGAII-IM algorithm, which does not rely on the influencers from the previous snapshot, are shown in the following figures. The following Figure 4-33 and Figure 4-34 shows the Pareto front of Facebook and Twitter networks based on the split event and without it.

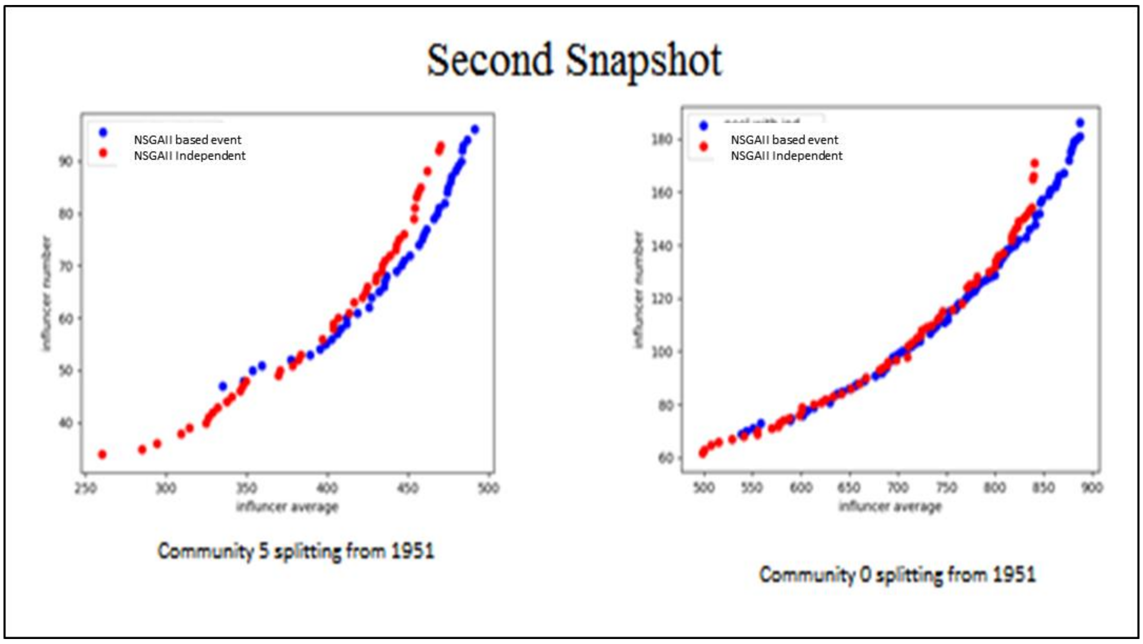


Figure 4-33 Pareto Front of a Facebook Dataset Using NSGAII-IM Based on Split Event Versus NSGAII-IM Independent

The splitting event appears only in the second snapshot with the Facebook network, whereas in the Twitter networks, the splitting event appears in the several snapshots.

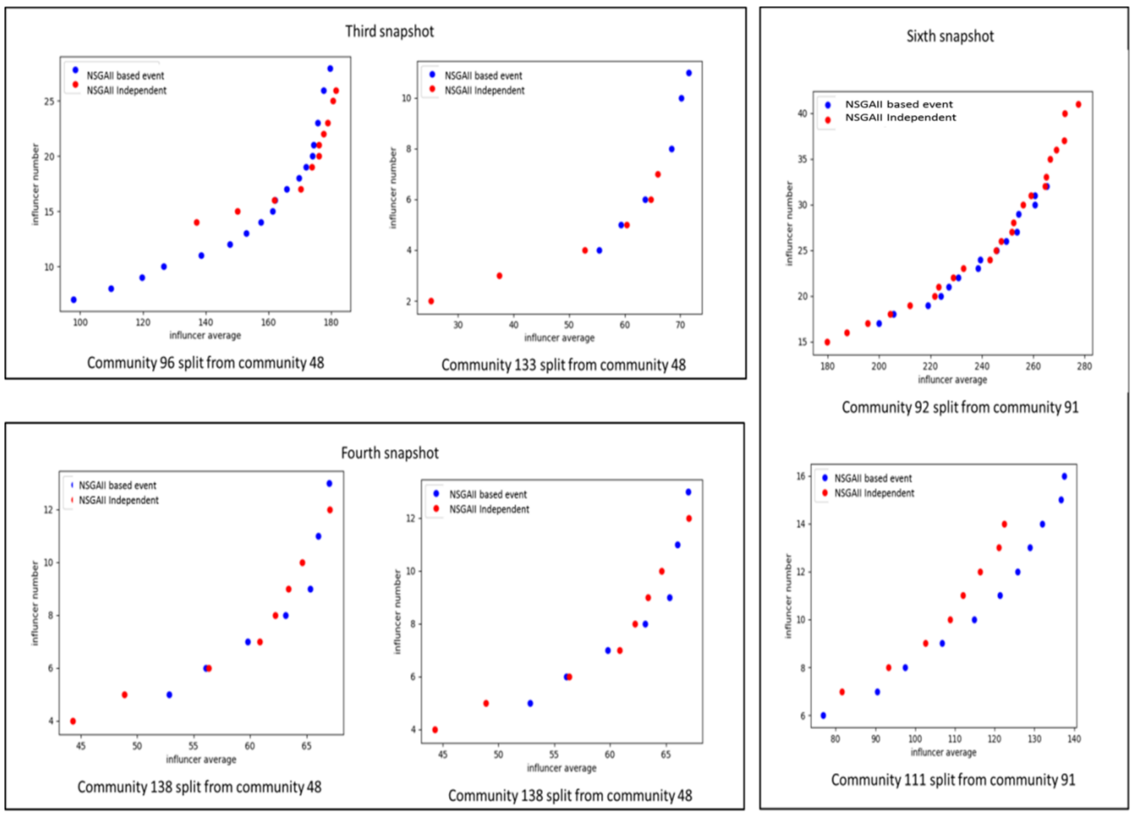


Figure 4-34 Pareto Front of a Twitter Dataset Using NSGAII-IM Based on Split Event Versus NSGAII-IM Independent

Previous figures illustrate the effect of taking advantage of influential users of the community that has been splitting into multi communities and can see the perform NSGAII-IM algorithm with a better Pareto Front from NSGAII-IM algorithm without Pareto Front in some snapshots. Meanwhile, in other snapshots, the performance of NSGAII-IM algorithm with Pareto is equal to NSGAII-IM algorithm without Pareto Front. Because the event of the division of communities affects the influential members and the strength of their influence, it is natural that new influential members appear according to the new structure of the community and in proportion to its new nature.

Merging Event

The last event that communities can experience is the merging. Two or more communities from the previous snapshot can merge into one community in the current snapshot. Each community has a set of influencers and former candidates that can be included as individuals in the initial population of the NSGAIM-II algorithm. The following figures show implementing the NSGAII-IM algorithm, which is based on this merging event and takes advantage of the influential users of this community from the previous snapshot, and the NSGAII-IM algorithm, which does not depend on the influencers from the previous snapshot.

The merge events that occur in the Facebook network are shown in the following Figure 4-35.

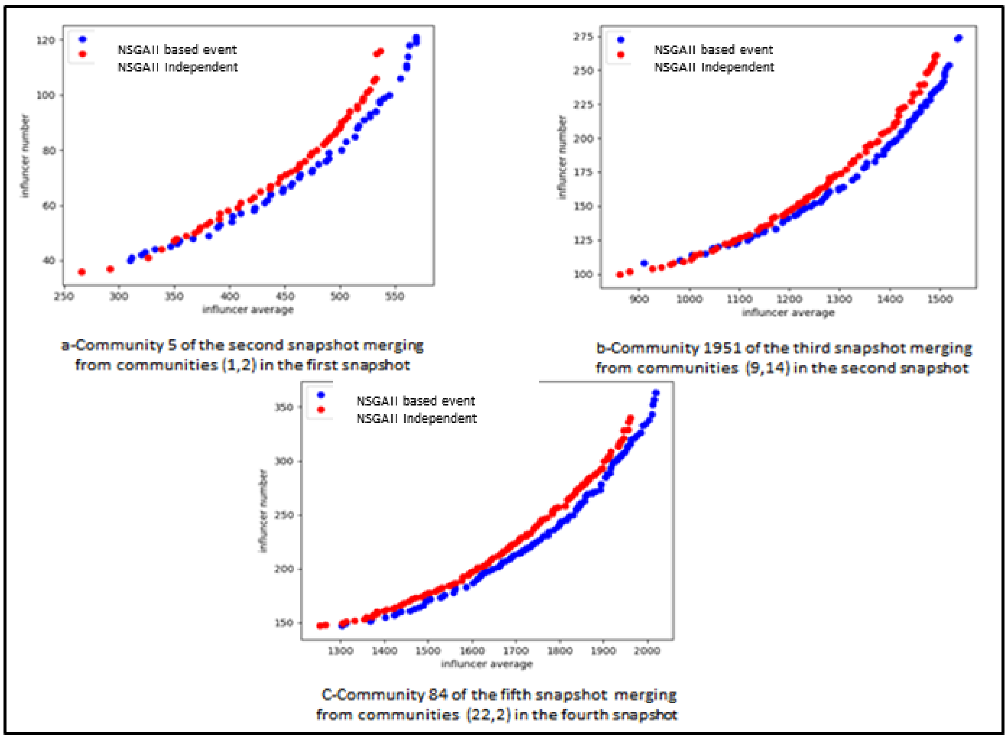


Figure 4-35 Pareto Front of a Facebook Dataset Using NSGAII-IM based on Merge Event Versus NSGAII-IM Independent

Concerning the Twitter network, the merge events have occurred in several snapshots over time, as shown in the subsequent Figure 4-36. The preceding figures demonstrate the impact of leveraging influential users of communities that have been merging into a single community.

In the Facebook and Twitter datasets, candidate solutions by that NSGAII-IM algorithm supported by the previous Pareto Front provide greater influence than NSGAII-IM algorithm without independently.

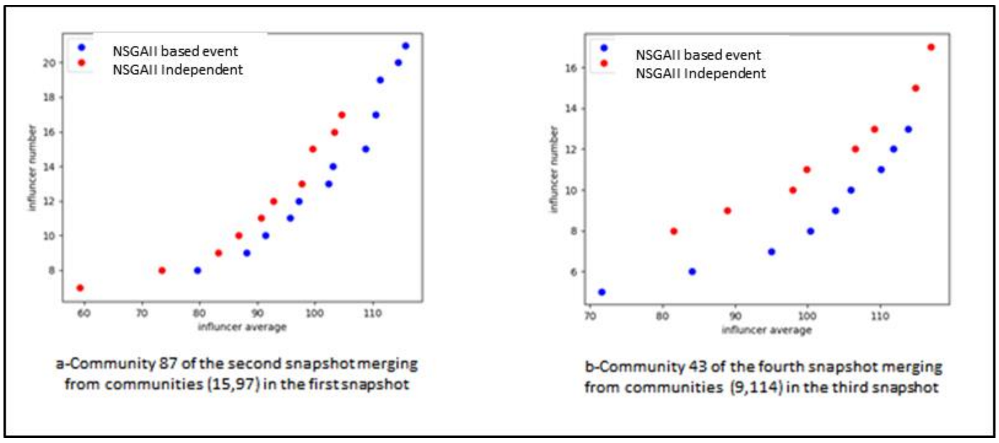


Figure 4-36 Pareto front of a Twitter dataset using NSGAII-IM based on Merge event versus NSGAII-IM independent

4.5 Study and Analysis

The promoters of a particular product target influential nodes that retain their influence through snapshots and maybe help to predict them in the future, which motivated us to conduct this study. We try to shed light on the influential nodes for more than one snapshot. When tracking the influencers which remain influential for more than two consecutive snapshots, even if those snapshots are not sequential, the influencers can be tracked. The influencer can be affected by circumstances for a certain period that reduces his interaction in the network, which affects the strength of his influence on the rest of the users in this period, but when he resumes his interaction in the network, the power of his influence increases. Figure 4-37 represents the percentage of influencers node through snapshot one and others, where S refers to the snapshot, and the number next to it represents the index of it. For example, S1 represents the first snapshot. From Figure 4-37, we can see that the influencers who stay for two snapshots, whether consecutive or not, their percentage is more than those who remain for several snapshots. As the largest percentage of the remaining influencers are between the first and last snapshots in the Facebook network, whereas a Twitter network has the largest percentage of influencers remain between snapshot one and three, as for three snapshots, the greater proportion of the staying influencers are between the first, second, and fourth snapshots in the Facebook network and a Twitter network among first, third, and sixth snapshots.

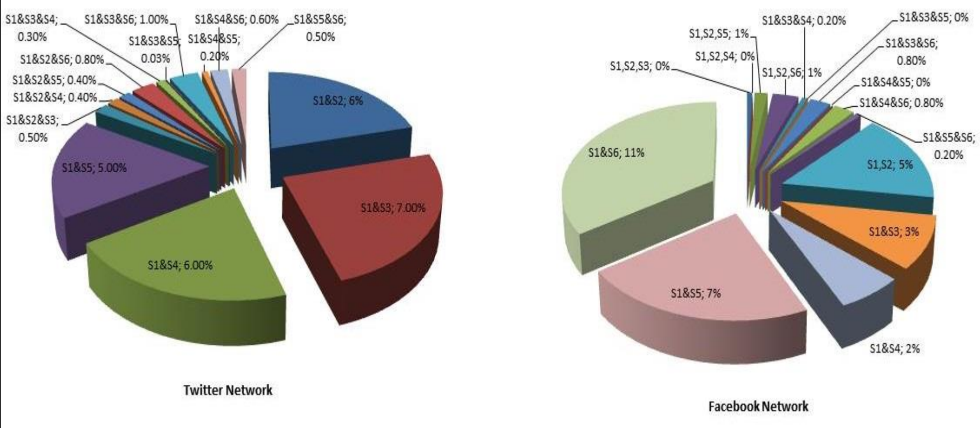


Figure 4-37 The Percentage of Influencers Node through Snapshot One and other Snapshots

Figure 4-38 shows the percentage of influencers who stay from the second snapshot to the rest of the snapshots. There is a noticeable difference in the rate of influencers who stay for two snapshots compared to those who continue for three snapshots.

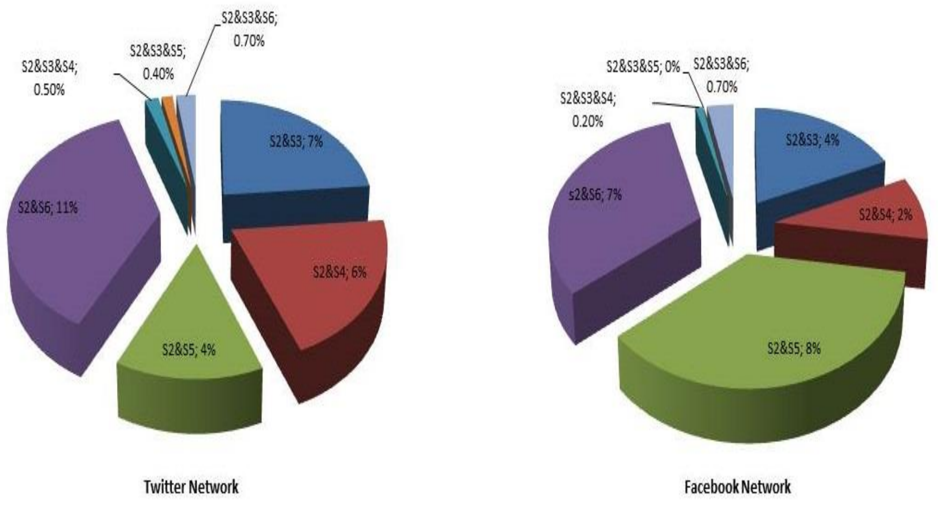


Figure 4-38 The Percentage of Influencers Node through Snapshot 2 and other Snapshots

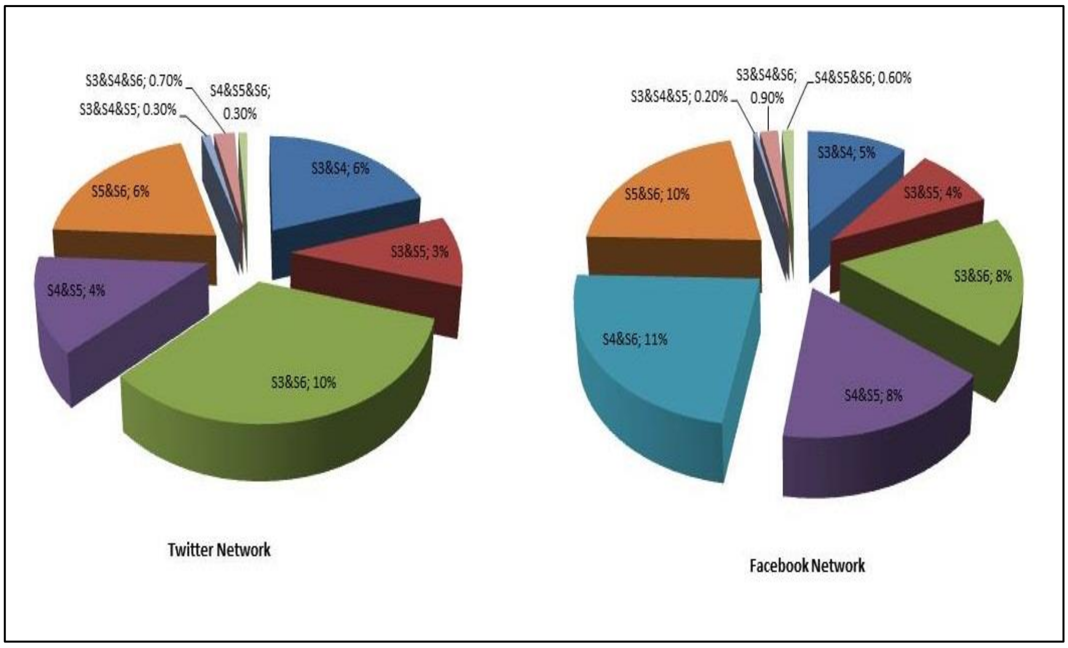


Figure 4-39 The Percentage of Influencers Node through Snapshot Three and other Snapshots

The target is the influencer that maintains its strong influence for the longest possible period because it is still classified as influential despite the change of circumstances and the acceleration of events. The previous Figure 4-39 shows the percentage of influencers between snapshots. For example, the rate of influencers between the third and sixth snapshots in Twitter data is 10%, higher than the percentage of the influencers between the third snapshots with the fourth or fifth. This means that a number of influencers lost the power of their influence in the fourth and fifth periods and regained it in the sixth period.

Chapter Five:

Conclusion and Future Works

5.1 Conclusion

This work verifies the usefulness of events experienced by communities in dynamic social networks by tracking influencer nodes over time. The research was carried out on large empirical networks. Since most social networks are large, dynamic, and evolving, identifying significant people for influence spread via traditional methods will be difficult and time-consuming. We have suggested a dynamic modified NSGAII-IM algorithm that analyzes the communities due to their importance in the diffusion process. At the conclusion of this study, some significant conclusions were reached.

1. The Clique-Louvain algorithm was proposed to detect communities in social networks. Its modularity is almost identical to that of the traditional Louvain algorithm when applied to synthetic networks. In the same scenario of modularity, the matter is a bit different when applying both algorithms to real networks. In sum, the reason can be attributed to the structures of the networks. The overall results of the clique-Louvain algorithm show that it reduces the execution time of the original Louvain algorithm by 23.91% for the Facebook dataset, 37.22% for the DIGG dataset, and decreases the execution time of the Twitter dataset by 23.77%. Figures 4-6 can be reviewed. Detecting communities in networks became easier after finding maximal cliques and developing their maximal clique graph. Because the Louvain algorithm starts from the clique rather than the node, this pretreatment of edges permits the connection of maximal cliques of graphs, reducing the complexity of the connections. Densely organized networks, it may be concluded, are the best fit for this algorithm.
2. According to one experiment on real networks, the suggested NSGAII-IM algorithm method beat three heuristics (Degree, SDISC,

and HIGHDEG) with the same seed size and fewer generations than the NSGAII algorithm. The findings of this experiment suggest that the proposed strategy for picking individuals (seeds) influences the algorithm's performance. The fact that guiding the first generation of NSGAII-IM algorithm leads to a faster convergence on superior Pareto Fronts is significant. Individuals with strong centrality measures can be assigned to the starting population.

3. Considering the results obtained from applying the NSGAII-IM algorithm to three real networks, the maximization of the influence spread is improved by 7% for the Facebook wall post and Twitter networks. At the same time, the proposed approach demonstrated the improvement in influence spread by 14% for the Digg network. It seems that the percentage of improvement in the Facebook and Twitter networks is identical and differed in the Digg network (see Figure 4-21 and Figure 4-22). It can be concluded from these percentages that the nature of the network plays the primary role in improving the influence spread using the NSGAII-IM algorithm.
4. From another side, the role of the proposed propagation model WIC cannot be ignored, as it is credited with the superiority of the NSGAII-IM algorithm also when the other conditions are fixed. Furthermore, the nature of the network affects the rate of influence propagation. The Digg networks are denser in nature than the wall post-Facebook and Twitter networks (see Figure 4-23 and Figure 4-24). Thus, the interactions between nodes are greater in a shorter period of time, affecting the effect propagation's nature.
5. The tracking of the events of communities in a dynamic network is based on the time intervals of chosen snapshots. Each time interval snapshot depicts the network at a specific point in time. Short time intervals chosen in Facebook show the formation of many small

groups, as shown in Figure 4-9. In contrast, longer time intervals in Twitter may illustrate incorrect graph structure because they do not consider the small changes in snapshots that have been merged together, as shown in Figure 4-11.

6. It is possible to conclude that the time period during which the data for the Facebook network was collected affects the nature of events, especially since it was at the beginning of the network’s fame in 2006, which means that the number of members was increasing, and this affected the birth and death events as shown in Table 4-2 which fluctuate from one snapshot to the next. On the contrary, the data from the Twitter network was collected in 2010. In other words, because it is not the time of the Twitter platform's creation, the number of communities that go through birth and death events is nearly the same as illustrated in Table 4-3.
7. The NSGAII-IM algorithm tries to find the shortest effective set of nodes that should have the most impact on users in social networks at a given timestamp. Instead of creating an initial set from scratch, the algorithm starts from the pre-existing influent seed set (Pareto Front) depending on the event the community was going through from the previous snapshot, except for the birth event where the community formed in the current snapshot did not exist in the previous snapshot, so the algorithm is forced to create an initial set from scratch. This explains why the blue and red curves are congruent in Figure 4-25. The community size may be to blame for the growth and contraction events. When a community expands or shrinks greatly, it means that many new members belong to that community or a significant number of members leave that community, respectively. Hence, the influence of influencers on the members of the community may decrease, and new influencers

appear. Results in the snapshots vary between good and very good, and in some cases, the performance is equal between the NSGAII-IM algorithm with Pareto Front, and the NSGAII-IM algorithm started from scratch

8. As the split event in a community affects the strength of the influential members. According to the new structure of the community, and as a result of the community losing some of its members, it is natural that new influential members appear in a way that is commensurate with its new nature. If we compare the effect of merge events (see Figure 4-35 and Figure 4-36) with grow, contraction, and splitting events, we can see that NSGAII-IM algorithm with Pareto Front performs better than NSGAII-IM algorithm without Pareto Front in all the snapshots that include merging event. This is because we take the influencers of the communities before they merge as candidates into the current community. It is usual for some influencers to retain their influence power over the rest of the individuals or may increase their strength in some cases.
9. In the study that has been carried out, some influencers may last for a while. In other words, the influencers retain their influence on the network members for a long time. The study might be a promising trend in predicting influencers in dynamic social networks. The influencer may not interact with other users in the network for a certain amount of time because of a set of circumstances. This reduces the strength of his influence on other users during this time. Still, when he gets back into the network, he regains his influence's strength, which explains their appearance for a while and then their disappearance and appearance again. The highest percentage of

influencers staying for two periods is 11%. We also note that the percentages are somewhat similar in both networks.

10. As a final point, we say that the expansion event is the most common in both Facebook and Twitter, while other events vary in frequency from network to network. For example, the contraction event appears in one snapshot in the Facebook network and several snapshots in the Twitter network.

5.2 Future Works

Some related future works that can be achieved further are listed below:

1. Identify the influencer nodes, taking into account the types of diffused topics. Accordingly, the influence among the users will be modeled in a different way.
2. Optimize the initial population's structure which reduces the cost of computation using the deep learning and new centrality criteria along to community discovery.
3. Develop an improved version of the NSGAII-IM algorithm to accommodate a more complex influence dispersion procedure that takes into account the context of the situation.
4. Predicting how influential nodes change in dynamic social networks whereby the nodes and edges evolve.

References

Abbasi, F., & Fazl-Ersi, E. (2022). Identifying Influentials in Social Networks. *Applied Artificial Intelligence*, *00*(00), 1–25. https://doi.org/10.1080/08839514.2021.2010886

Abo-Elnaga, Y., & Nasr, S. (2020). Modified evolutionary algorithm and chaotic search for Bilevel programming problems. *Symmetry*, *12*(5), 767.

Agarwal, S., & Mehta, S. (2018). Social Influence Maximization Using Genetic Algorithm with Dynamic Probabilities. *2018 11th International Conference on Contemporary Computing, IC3 2018*, 1–6. https://doi.org/10.1109/IC3.2018.8530626

Aggarwal, C. C., Lin, S., & Yu, P. S. (2012). On influential node discovery in dynamic social networks. *Proceedings of the 12th SIAM International Conference on Data Mining, SDM 2012*, 636–647. https://doi.org/10.1137/1.9781611972825.55

Alasadi, M. K., & Almamory, H. N. (2019). Diffusion model based on shared friends-aware independent cascade. *Journal of Physics: Conference Series*, *1294*(4). https://doi.org/10.1088/1742-6596/1294/4/042006

Alrajebah, N., Luczak-Roesch, M., Carr, L., & Tiropanis, T. (2017). Deconstructing diffusion on Tumblr: Structural and temporal aspects. *WebSci 2017 - Proceedings of the 2017 ACM Web Science Conference*, 319–328. https://doi.org/10.1145/3091478.3091491

Alshahrani, M., Fuxi, Z., Sameh, A., Mekouar, S., & Huang, S. (2020). Efficient algorithms based on centrality measures for identification of top-K influential users in social networks. *Information Sciences*, *527*, 88–107. https://doi.org/10.1016/j.ins.2020.03.060

Aslay, C., Lu, W., Lakshmanan, L. V. S., & Xiao, X. (2018). Influence maximization in online social networks. *WSDM 2018 - Proceedings of the 11th ACM International Conference on Web Search and Data Mining*, *2018–Febua*, 775–776. https://doi.org/10.1145/3159652.3162007

Bao, Z. K., Ma, C., Xiang, B. B., & Zhang, H. F. (2017). Identification of influential nodes in complex networks: Method from spreading probability viewpoint. *Physica A: Statistical Mechanics and Its Applications*, *468*, 391–397. https://doi.org/10.1016/j.physa.2016.10.086

Behera, R. K., Rath, S. K., Misra, S., Damaševičius, R., & Maskeliunas, R. (2019). Distributed centrality analysis of social network data using MapReduce. *Algorithms, 12*(8). https://doi.org/10.3390/a12080161

Bilal, S., & Abdelouahab, M. (2017). Evolutionary algorithm and modularity for detecting communities in networks. *Physica A: Statistical Mechanics and Its Applications, 473,* 89–96.

Blondel, V. D., Guillaume, J.-L., Lambiotte, R., & Lefebvre, E. (2008). Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment, 2008*(10), P10008.

Bucur, D., & Iacca, G. (2016). Influence maximization in social networks with genetic algorithms. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 9597,* 379–392. https://doi.org/10.1007/978-3-319-31204-0_25

Bucur, D., Iacca, G., Marcelli, A., Squillero, G., & Tonda, A. (2017). Multi-objective evolutionary algorithms for influence maximization in social networks. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 10199 LNCS,* 221–233. https://doi.org/10.1007/978-3-319-55849-3_15

Burbach, L., Halbach, P., Ziefle, M., & Calero Valdez, A. (2020). Opinion Formation on the Internet: The Influence of Personality, Network Structure, and Content on Sharing Messages Online. *Frontiers in Artificial Intelligence, 3*(July), 1–16. https://doi.org/10.3389/frai.2020.00045

Can, U., & Alatas, B. (2019). A new direction in social network analysis : Online social network analysis problems and applications. *Physica A, 535,* 122372. https://doi.org/10.1016/j.physa.2019.122372

Cui, L., Hu, H., Yu, S., Yan, Q., Ming, Z., Wen, Z., & Lu, N. (2018). DDSE: A novel evolutionary algorithm based on degree-descending search strategy for influence maximization in social networks. *Journal of Network and Computer Applications, 103,* 119–130. https://doi.org/10.1016/j.jnca.2017.12.003

Dakiche, N., Benbouzid-Si Tayeb, F., Slimani, Y., & Benatchba, K. (2019). Tracking community evolution in social networks: A survey. *Information Processing and Management, 56*(3), 1084–1102. https://doi.org/10.1016/j.ipm.2018.03.005

Dawar, S., Bera, D., & Goyal, V. (2018). High-utility itemset mining for subadditive monotone utility functions. *ArXiv Preprint ArXiv:1812.07208.*

Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, *6*(2), 182–197.

Dey, P., Chaterjee, A., & Roy, S. (2019). Influence maximization in online social network using different centrality measures as seed node of information propagation. *Sadhana - Academy Proceedings in Engineering Sciences*, *44*(9). https://doi.org/10.1007/s12046-019-1189-7

Ding, J., Sun, W., Wu, J., & Guo, Y. (2020). Influence maximization based on the realistic independent cascade model. *Knowledge-Based Systems*, *191*(xxxx), 105265. https://doi.org/10.1016/j.knosys.2019.105265

Domingos, P., & Richardson, M. (2001). Mining the network value of customers. *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 57–66.

Elhishi, S., Abu-Elkheir, M., & Abou Elfetouh, A. (2019). Perspectives on the evolution of online communities. *Behaviour and Information Technology*, *38*(6), 592–608. https://doi.org/10.1080/0144929X.2018.1546901

Foroozani, A., & Ebrahimi, M. (2021). Nonlinear anomalous information diffusion model in social networks. *Communications in Nonlinear Science and Numerical Simulation*, *103*, 106019. https://doi.org/10.1016/j.cnsns.2021.106019

Gokturk, G., & Kaya, K. (2021). Fast and Error-Adaptive Influence Maximization based on Count-Distinct Sketches. *ArXiv Preprint ArXiv:2105.04023*.

Guo, J., Chen, F., & Li, M. (2019a). *A Multi-objective Optimization Approach for In fl uence Maximization in Social Networks* (Vol. 2). Springer Singapore. https://doi.org/10.1007/978-981-13-3402-3

Guo, J., Chen, F., & Li, M. (2019b). A Multi-objective Optimization Approach for Influence Maximization in Social Networks. *Proceeding of the 24th International Conference on Industrial Engineering and Engineering Management 2018*, 706–715.

Gursoy, F. (2019). Predicting Diffusion Reach Probabilities via Representation Learning on Social Networks. *CoRR*, *abs/1901.0*(Icm), 1–7. https://doi.org/10.6084/m9.figshare.7565894

Hafiene, N., Karoui, W., & Ben Romdhane, L. (2020). Influential nodes detection in dynamic social networks: A survey. *Expert Systems with Applications*, *159*, 113642. https://doi.org/10.1016/j.eswa.2020.113642

Hafiene, N., Karoui, W., & Romdhane, L. Ben. (2019). *Influential Nodes Detection in Dynamic* (Vol. 2). Springer International Publishing. https://doi.org/10.1007/978-3-030-20482-2

Han, J., Li, W., Zhao, L., Su, Z., Zou, Y., & Deng, W. (2017). Community detection in dynamic networks via adaptive label propagation. *PLoS ONE*, *12*(11), 1–16. https://doi.org/10.1371/journal.pone.0188655

Hong, T., & Liu, Q. (2019). Seeds selection for spreading in a weighted cascade model. *Physica A*, *526*, 120943. https://doi.org/10.1016/j.physa.2019.04.179

Hu, B., Li, W., Huo, X., Liang, Y., Gao, M., & Pei, P. (2016). Improving Louvain algorithm for community detection. *2016 International Conference on Artificial Intelligence and Engineering Applications*.

Huang, H., Meng, Z., & Shen, H. (2021). Competitive and complementary influence maximization in social network: A follower's perspective. *Knowledge-Based Systems*, *213*, 106600. https://doi.org/10.1016/j.knosys.2020.106600

Javadi, S. H. S., Gharani, P., & Khadivi, S. (2018). Detecting community structure in dynamic social networks using the concept of leadership. *Studies in Systems, Decision and Control*, *145*, 97–118. https://doi.org/10.1007/978-3-319-74412-4_7

Journal, I., Singh, M., & Malik, A. (2019). A Study of Genetic Algorithm and Crossover Techniques Cite this paper Related papers Fundament als of Genet ic Algorit hms Art ificial Int elligence Ret urn t o Websit e Fundament als … Raj Baraiya Genet ic Algorit hms basics Debabrat a Singh Aut omat ic . *International Journal of Computer Science and Mobile Computing*, *8*(3), 335–344. www.ijcsmc.com

Ju, W., Chen, L., Li, B., Liu, W., Sheng, J., & Wang, Y. (2020). A new algorithm for positive influence maximization in signed networks. *Information Sciences*, *512*(xxxx), 1571–1591. https://doi.org/10.1016/j.ins.2019.10.061

Katoch, S., Chauhan, S. S., & Kumar, V. (2021). A review on genetic algorithm: past, present, and future. *Multimedia Tools and Applications*, *80*(5), 8091–8126.

Krömer, P., & Nowaková, J. (2017). Guided genetic algorithm for the influence maximization problem. *International Computing and Combinatorics Conference*, 630–641.

Kumar, S., Saini, M., Goel, M., & Panda, B. S. (2021). Modeling information diffusion in online social networks using a modified forest-fire model.

Journal of Intelligent Information Systems, *56*(2), 355–377. https://doi.org/10.1007/s10844-020-00623-8

Li, W., Hu, Y., Wu, S., Bai, Q., & Lai, E. (2021). ABEM: An Adaptive Agent-based Evolutionary Approach for Mining Influencers in Online Social Networks. *Woodstock '18: ACM Symposium on Neural Gaze Detection, June 03â•fi05, 2018, Woodstock, NY*, *1*(1), 1–22. http://arxiv.org/abs/2104.06563

Li, X., Cheng, X., Su, S., & Sun, C. (2018). Community-based seeds selection algorithm for location aware influence maximization. *Neurocomputing*, *275*, 1601–1613. https://doi.org/10.1016/j.neucom.2017.10.007

Li, X., Zhang, X., Zhao, C., Yi, D., & Li, G. (2020). Identifying highly influential nodes in multilayer networks based on global propagation. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, *30*(6), 61107.

Li, Y., Fan, J., Wang, Y., & Tan, K.-L. (2018). Influence maximization on social graphs: A survey. *IEEE Transactions on Knowledge and Data Engineering*, *30*(10), 1852–1872.

Liu, X., Wu, S., Liu, C., & Zhang, Y. (2021). Social network node influence maximization method combined with degree discount and local node optimization. *Social Network Analysis and Mining*, *11*(1), 1–18.

Lotf, J. J., Azgomi, M. A., & Dishabi, M. R. E. (2022). An improved influence maximization method for social networks based on genetic algorithm. *Physica A: Statistical Mechanics and Its Applications*, *586*, 126480.

Lü, L., Chen, D., Ren, X.-L., Zhang, Q.-M., Zhang, Y.-C., & Zhou, T. (2016). Vital nodes identification in complex networks. *Physics Reports*, *650*, 1–63.

Maier, H. R., Razavi, S., Kapelan, Z., Matott, L. S., Kasprzyk, J., & Tolson, B. A. (2019). Introductory overview: Optimization using evolutionary algorithms and other metaheuristics. *Environmental Modelling and Software*, *114*, 195–213. https://doi.org/10.1016/j.envsoft.2018.11.018

Mirjalili, S. (2019). Evolutionary algorithms and neural networks. *Studies in Computational Intelligence*, *780*.

Mohammadi, A., & Saraee, M. (2018). Finding influential users for different time bounds in social networks using multi-objective optimization. *Swarm and Evolutionary Computation*, *40*(February), 158–165. https://doi.org/10.1016/j.swevo.2018.02.003

Mohapatra, D., Panda, A., Gouda, D., & Sahu, S. S. (2020). A Combined Approach for k-Seed Selection Using Modified Independent Cascade Model.

In *Computational Intelligence in Pattern Recognition* (pp. 775–782). Springer.

Newman, M. E. J., & Girvan, M. (2004). Finding and evaluating community structure in networks. *Physical Review E*, *69*(2), 26113.

Olivares, R., Muñoz, F., & Riquelme, F. (2021). A multi-objective linear threshold influence spread model solved by swarm intelligence-based methods. *Knowledge-Based Systems*, *212*, 106623. https://doi.org/10.1016/j.knosys.2020.106623

Ozaki, N., Tezuka, H., & Inaba, M. (2016). A simple acceleration method for the Louvain algorithm. *International Journal of Computer and Electrical Engineering*, *8*(3), 207.

Palla, G., Derényi, I., Farkas, I., & Vicsek, T. (2005). Uncovering the overlapping community structure of complex networks in nature and society. *Nature*, *435*(7043), 814–818. https://doi.org/10.1038/nature03607

Panizo-LLedot, A., Bello-Orgaz, G., & Camacho, D. (2020). A Multi-Objective Genetic Algorithm for detecting dynamic communities using a local search driven immigrant's scheme. *Future Generation Computer Systems*, *110*(November), 960–975. https://doi.org/10.1016/j.future.2019.10.041

Peng, C. (2020). Channel Optimization of Marketing Based on Users' Social Network Information. *Complexity*, *2020*(Ic). https://doi.org/10.1155/2020/8833780

Rodrigues, R. F., da Silva, A. R., da Fonseca Vieira, V., & Xavier, C. R. (2018). Optimization of the choice of individuals to be immunized through the genetic algorithm in the SIR model. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *10961 LNCS*(November 2020), 62–75. https://doi.org/10.1007/978-3-319-95165-2_5

Sabour, S., & Moeini, A. (2019). Creating a Maximal Clique Graph to Improve Community Detection in SCoDA and OSLOM Algorithms. *International Journal of Information and Communication Technology Research*, *11*(4), 48–56. http://ijict.itrc.ac.ir/article-1-378-en.html

Salavati, C., Abdollahpouri, A., & Manbari, Z. (2019). Ranking nodes in complex networks based on local structure and improving closeness centrality. *Neurocomputing*, *336*, 36–45. https://doi.org/10.1016/j.neucom.2018.04.086

Saxena, B., & Kumar, P. (2019). A node activity and connectivity-based model for influence maximization in social networks. *Social Network Analysis and Mining*, *9*(1). https://doi.org/10.1007/s13278-019-0586-6

Shang, J., Liu, L., Li, X., Xie, F., & Wu, C. (2016). Targeted revision: A learning-based approach for incremental community detection in dynamic networks. *Physica A*, *443*, 70–85. https://doi.org/10.1016/j.physa.2015.09.072

Shang, Y. (2019). Resilient consensus for expressed and private opinions. *IEEE Transactions on Cybernetics*, *51*(1), 318–331.

Silva, A. R. da, Rodrigues, R. F., Vieira, V. da F., & Xavier, C. R. (2018). Influence maximization in network by genetic algorithm on linear threshold model. *International Conference on Computational Science and Its Applications*, 96–109.

Song, G., Li, Y., Chen, X., He, X., & Tang, J. (2017). Influential Node Tracking on Dynamic Social Network: An Interchange Greedy Approach. *IEEE Transactions on Knowledge and Data Engineering*, *29*(2), 359–372. https://doi.org/10.1109/TKDE.2016.2620141

Sun, J., Deng, J., & Li, Y. (2020). Indicator & crowding distance-based evolutionary algorithm for combined heat and power economic emission dispatch. *Applied Soft Computing Journal*, *90*, 106158. https://doi.org/10.1016/j.asoc.2020.106158

Traag, V. A., Waltman, L., & van Eck, N. J. (2019). From Louvain to Leiden: guaranteeing well-connected communities. *Scientific Reports*, *9*(1), 1–12.

Tsai, C. W., Yang, Y. C., & Chiang, M. C. (2016). A Genetic NewGreedy Algorithm for Influence Maximization in Social Network. *Proceedings - 2015 IEEE International Conference on Systems, Man, and Cybernetics, SMC 2015*, *1*, 2549–2554. https://doi.org/10.1109/SMC.2015.446

Vardasbi, A., Faili, H., & Asadpour, M. (2017). A Behavioral Analysis on the Reselection of Seed Nodes in Independent Cascade Based Influence Maximization. *ArXiv Preprint ArXiv:1708.01891*.

Wang, F., Jiang, W., Li, X., & Wang, G. (2018). Maximizing positive influence spread in online social networks via fluid dynamics. *Future Generation Computer Systems*, *86*, 1491–1502. https://doi.org/10.1016/j.future.2017.05.050

Wang, X., Chen, G., & Xu, S. (2022). Cleaner Logistics and Supply Chain Bi-objective green supply chain network design under disruption risk through an extended NSGA-II algorithm. *Cleaner Logistics and Supply Chain*, *3*(October 2021), 100025. https://doi.org/10.1016/j.clscn.2021.100025

Wang, Z., Li, Z., Yuan, G., Sun, Y., Rui, X., & Xiang, X. (2018). Tracking the Evolution of Communities in Dynamic Social Networks. *Knowledge-Based Systems*, *157*, 81–97. https://doi.org/10.1016/j.knosys.2018.05.026

Weskida, M., & Michalski, R. (2019). Finding influentials in social networks using evolutionary algorithm. *Journal of Computational Science*, *31*, 77–85.

Wilder, B., & Vorobeychik, Y. (2018). Controlling elections through social influence. *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS*, *1*, 265–273.

Wu, Q., & Hao, J. K. (2015). A review on algorithms for maximum clique problems. *European Journal of Operational Research*, *242*(3), 693–709. https://doi.org/10.1016/j.ejor.2014.09.064

Xu, Z., Rui, X., He, J., Wang, Z., & Hadzibeganovic, T. (2020). Superspreaders and superblockers based community evolution tracking in dynamic social networks. *Knowledge-Based Systems*, *192*, 105377. https://doi.org/10.1016/j.knosys.2019.105377

Yang, K., Guo, Q., Li, S. N., Han, J. T., & Liu, J. G. (2017). Evolution properties of the community members for dynamic networks. *Physics Letters, Section A: General, Atomic and Solid State Physics*, *381*(11), 970–975. https://doi.org/10.1016/j.physleta.2017.01.030

Yang, Y., Wang, Z., Jin, T., Pei, J., & Chen, E. (2018). Tracking top-k influential vertices in dynamic networks. *ArXiv Preprint ArXiv:1803.01499*.

Yin, G., Chi, K., Dong, Y., & Dong, H. (2017). An approach of community evolution based on gravitational relationship refactoring in dynamic networks. *Physics Letters, Section A: General, Atomic and Solid State Physics*, *381*(16), 1349–1355. https://doi.org/10.1016/j.physleta.2017.01.059

Zhang, J., Fei, J., Song, X., & Feng, J. (2021). An Improved Louvain Algorithm for Community Detection. *Mathematical Problems in Engineering*, *2021*. https://doi.org/10.1155/2021/1485592

Zhang, K., Du, H., & Feldman, M. W. (2017). Maximizing influence in a social network: Improved results using a genetic algorithm. *Physica A: Statistical Mechanics and Its Applications*, *478*, 20–30. https://doi.org/10.1016/j.physa.2017.02.067

Zhang, Z., Li, X., & Gan, C. (2021). Identifying influential nodes in social networks via community structure and influence distribution difference. *Digital Communications and Networks*, *7*(1), 131–139. https://doi.org/10.1016/j.dcan.2020.04.011

Zhao, B., Xue, Y., Xu, B., Ma, T., & Liu, J. (2018). Multi-objective classification based on NSGA-II. *International Journal of Computing Science and Mathematics*, *9*(6), 539–546. https://doi.org/10.1504/IJCSM.2018.096325

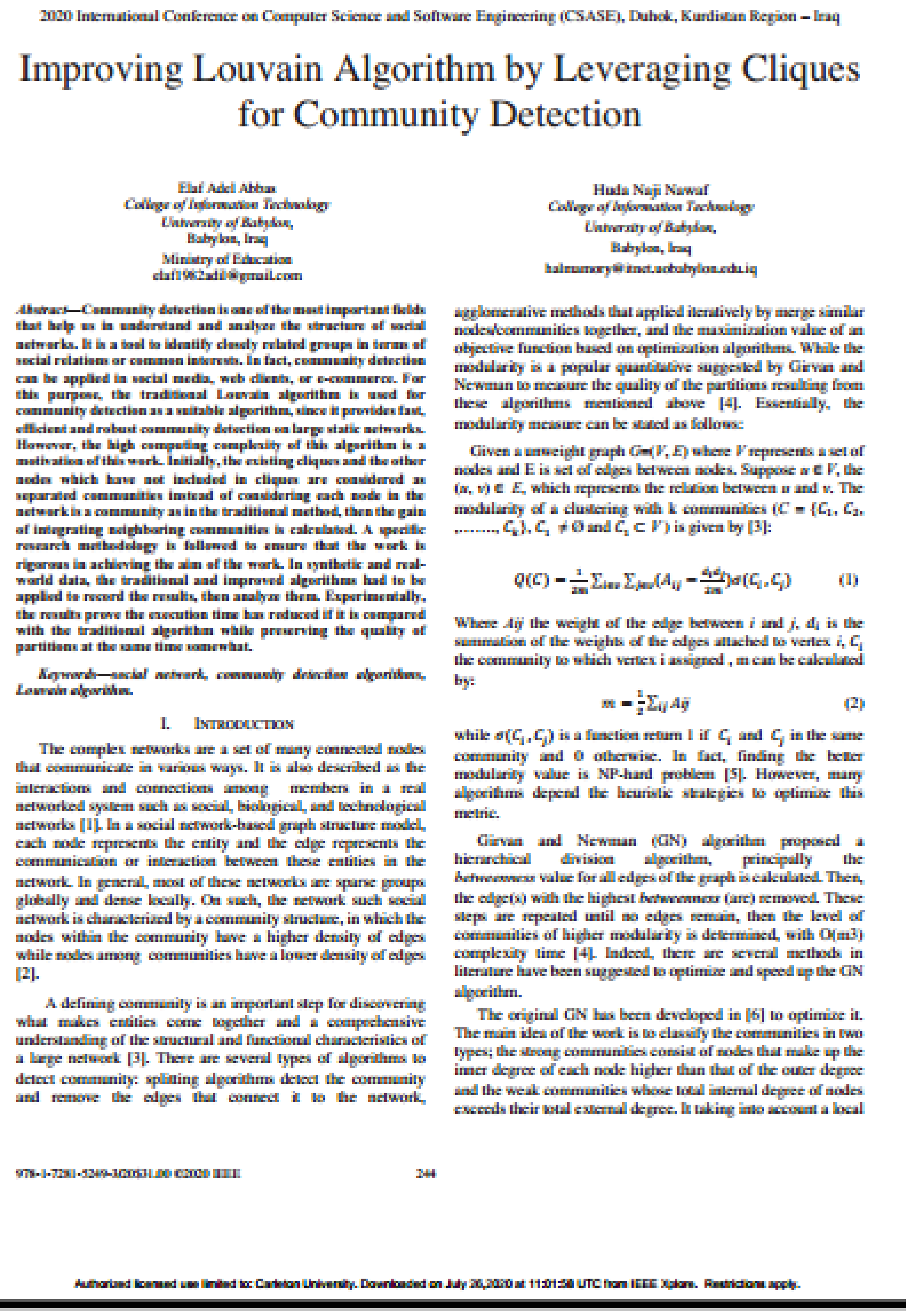
Zhou, X., Liang, W., Luo, Z., & Pan, Y. (2021). Periodic-Aware Intelligent Prediction Model for Information Diffusion in Social Networks. *IEEE Transactions on Network Science and Engineering*, *8*(2), 894–904. https://doi.org/10.1109/TNSE.2021.3064952

Ziwei He. (2018). *Community Tracking in Evolving Social Networks*. Université de Sherbrooke.

Appendix A

Published Papers

(1)



2020 International Conference on Computer Science and Software Engineering (CSASE), Duhok, Kurdistan Region – Iraq

Improving Louvain Algorithm by Leveraging Cliques for Community Detection

Elaf Adel Abbas
College of Information Technology
University of Babylon,
Babylon, Iraq
Ministry of Education
elaf1982adil@gmail.com

Huda Naji Nawaf
College of Information Technology
University of Babylon,
Babylon, Iraq
halmamory@itnet.uobabylon.edu.iq

***Abstract*—Community detection is one of the most important fields that help us in understand and analyze the structure of social networks. It is a tool to identify closely related groups in terms of social relations or common interests. In fact, community detection can be applied in social media, web clients, or e-commerce. For this purpose, the traditional Louvain algorithm is used for community detection as a suitable algorithm, since it provides fast, efficient and robust community detection on large static networks. However, the high computing complexity of this algorithm is a motivation of this work. Initially, the existing cliques and the other nodes which have not included in cliques are considered as separated communities instead of considering each node in the network is a community as in the traditional method, then the gain of integrating neighboring communities is calculated. A specific research methodology is followed to ensure that the work is rigorous in achieving the aim of the work. In synthetic and real-world data, the traditional and improved algorithms had to be applied to record the results, then analyze them. Experimentally, the results prove the execution time has reduced if it is compared with the traditional algorithm while preserving the quality of partitions at the same time somewhat.**

Keywords*—*social network, community detection algorithms, Louvain algorithm.

I. Introduction

The complex networks are a set of many connected nodes that communicate in various ways. It is also described as the interactions and connections among members in a real networked system such as social, biological, and technological networks [1]. In a social network-based graph structure model, each node represents the entity and the edge represents the communication or interaction between these entities in the network. In general, most of these networks are sparse groups globally and dense locally. On such, the network such social network is characterized by a community structure, in which the nodes within the community have a higher density of edges while nodes among communities have a lower density of edges [2].

A defining community is an important step for discovering what makes entities come together and a comprehensive understanding of the structural and functional characteristics of a large network [3]. There are several types of algorithms to detect community: splitting algorithms detect the community and remove the edges that connect it to the network, agglomerative methods that applied iteratively by merge similar nodes/communities together, and the maximization value of an objective function based on optimization algorithms. While the modularity is a popular quantitative suggested by Girvan and Newman to measure the quality of the partitions resulting from these algorithms mentioned above [4]. Essentially, the modularity measure can be stated as follows:

Given a unweight graph $G=(V, E)$ where V represents a set of nodes and E is set of edges between nodes. Suppose $u \in V$, the $(u, v) \in E$, which represents the relation between u and v. The modularity of a clustering with k communities ($C = \{C_1, C_2, ,\dots\dots, C_k\}$, $C_i \neq \emptyset$ and $C_i \subset V$) is given by [3]:

$$Q(C) = \frac{1}{2m}\sum_{i\in V}\sum_{j\in V}\left(A_{ij} - \frac{d_i d_j}{2m}\right)\sigma(C_i, C_j) \qquad (1)$$

Where Aij the weight of the edge between i and j, d_i is the summation of the weights of the edges attached to vertex i, C_i the community to which vertex i assigned , m can be calculated by:

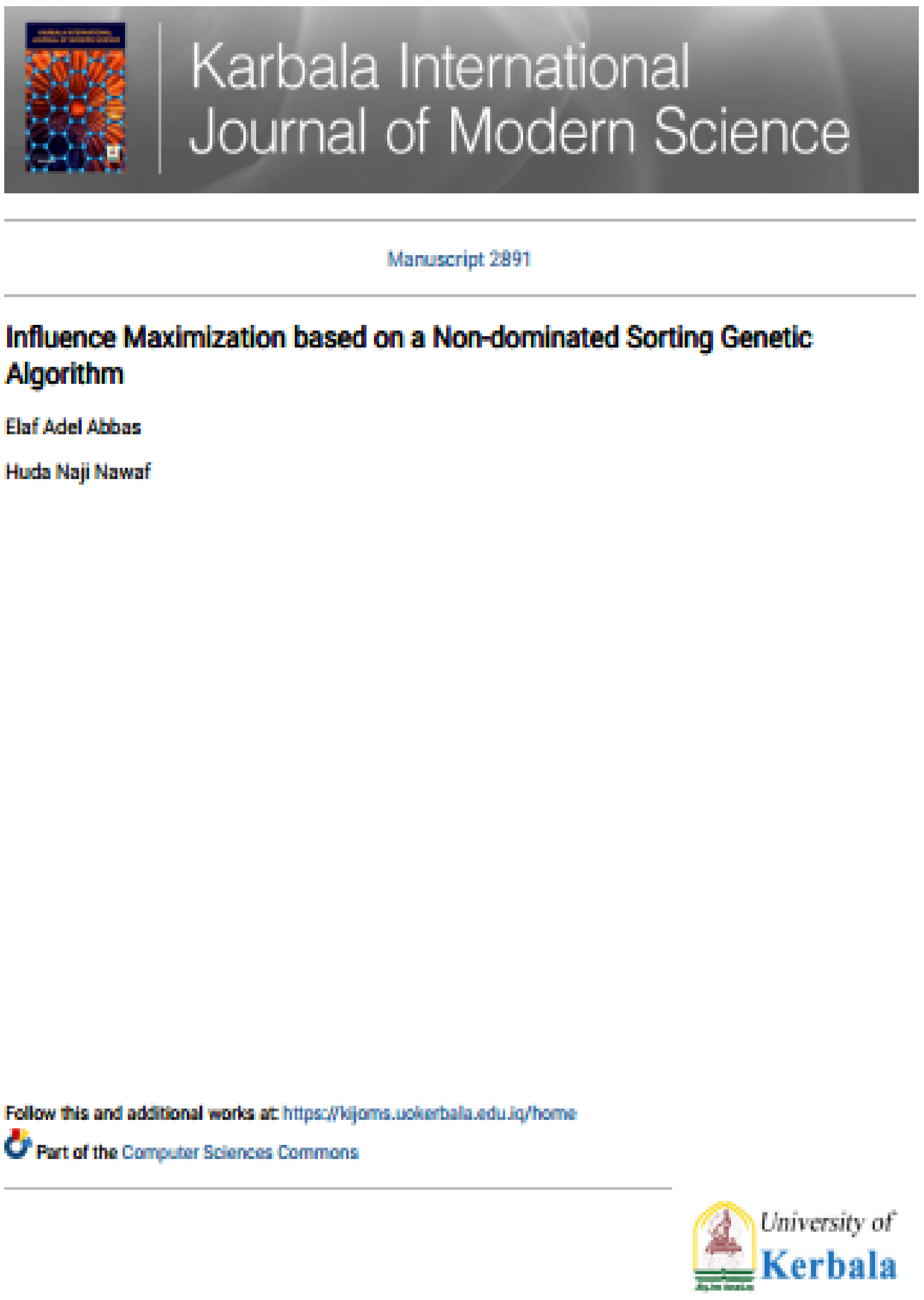
$$m = \frac{1}{2}\sum_{ij} Aij \qquad (2)$$

while $\sigma(C_i, C_j)$ is a function return 1 if C_i and C_j in the same community and 0 otherwise. In fact, finding the better modularity value is NP-hard problem [5]. However, many algorithms depend the heuristic strategies to optimize this metric.

Girvan and Newman (GN) algorithm proposed a hierarchical division algorithm, principally the *betweenness* value for all edges of the graph is calculated. Then, the edge(s) with the highest *betweenness* (are) removed. These steps are repeated until no edges remain, then the level of communities of higher modularity is determined, with O(m3) complexity time [4]. Indeed, there are several methods in literature have been suggested to optimize and speed up the GN algorithm.

The original GN has been developed in [6] to optimize it. The main idea of the work is to classify the communities in two types; the strong communities consist of nodes that make up the inner degree of each node higher than that of the outer degree and the weak communities whose total internal degree of nodes exceeds their total external degree. It taking into account a local

(2)



Influence Maximization based on a Non-dominated Sorting Genetic Algorithm

Elaf Adel Abbas

Huda Naji Nawaf

الخلاصة

تعتبر بنية الشبكات الاجتماعية في حالة تغير مستمر بمرور الوقت وينعكس هذا على المجتمعات وتبعا لهذا التغير يمر المجتمع بسلسلة من الاحداث وهي (الولادة ، الموت ، التوسع ،التقلص ، الانقسام والدمج). تركز الكثير من البحوث على تتبع المؤثرين في هذه الشبكات الديناميكية ككل ، بينما قد يكون هذا المؤثر ذو تأثير اكبر على اعضاء مجتمعه فيجب تتبعه في مجتمعه . تلعب الاحداث التي يخضع لها المجتمع دورا اساسيا في تحديد الاشخاص المؤثرين في الشبكة فيمكن الاستفادة من الاشخاص المؤثرين الذين ظهروا في الفترة السابقة لأنه من المحتمل الاحتفاظ بقوة تأثيرهم في الفترة الحالية شريطة ان يتواجد مجتمعهم في الفترة الحالية وظهر له احد الاحداث التالية (توسع ،تقلص ،انقسام او دمج).

يركز هذا البحث على تتبع المؤثرين الذين يكون لهم اكبر تأثير على اعضاء مجتمعاتهم ويكون ضمن مرحلتين:

المرحلة الاولى :اكتشاف المجتمعات الديناميكية حسب خوارزمية ليوفان المحسنة والتي تستغل مفهوم الزمر في تكوين المجتمعات.

المرحلة الثانية :تحديد المؤثرين في هذه المجتمعات باستخدام خوارزمية NSGAII-IM بهدفين متضاربين فيمثل الهدف الاول تحديد اقل عدد من الاشخاص المؤثرين في الشبكة والهدف الثاني تحقيق اعلى معدل انتشار في الشبكة.

يتم تمثيل السكان الاولين لخوارزمية NSGAII-IM بأفراد ذو أطوال متغيرة كمجموعة البذور ، ويجب تصميم نموذج الانتشار باسم WIC لنمذجة التأثير بين كل زوج من العقد. بالنسبة للوظيفة متعددة الأهداف ، تم تعيين زيادة حجم تغطية التأثير وتقليل عدد العقد الأولية قدر الإمكان كهدفين متعارضة لهذه الخوارزمية.

تم اجراء العديد من التجارب على مجموعة بيانات حقيقية واظهرت النتائج ان الطريقة المقترحة في اكتشاف المجتمعات قللت الوقت 24,25% في بيانات الفيس بوك، اما خوارزمية NSGAII-IM قد حققت تحسن في معدل انتشار التأثير بنسبة ١٤% في شبكة الدك وهي من الشبكات الثابتة. من ناحية أخرى ، يؤدي تنفيذ خوارزمية NSGAII-IM المقترحة مع الشبكات الديناميكية والاستفادة من أحداث المجتمع إلى تحسين الأداء بنسبة 11٪ في حدث الدمج عند مقارنته بالأداء بدون أحداث. تعكس هذه النسبة أعلى معدل تحسن عند مقارنتها بالأحداث الأخرى.

جمهورية العراق
وزارة التعليم العالي والبحث العلمي
جامعة بابل
كلية تكنولوجيا المعلومات
قسم البرمجيات

تحديد العقد المؤثرة على أساس تتبع تطور المجتمعات في الشبكات الاجتماعية الديناميكية

اطروحة
مقدمة إلى مجلس كلية تكنولوجيا المعلومات في جامعة بابل
كجزء من متطلبات الحصول على درجة الدكتوراه في الفلسفة في
تكنولوجيا المعلومات - البرمجيات

من قبل
ايلاف عادل عباس عيسى

بأشراف
أ. د. هدى ناجي نواف عمران

1443 2022