Chapter 5

Multiple Influence Maximization across Multiple Social Networks

In chapter 3 and chapter 4, the seed users are identified in the networks having single connections among nodes. In this chapter, multiple featured networks are considered for influence maximization. Also, consider the scenario that a company may intend to promote several kinds of products in the same social networks simultaneously.

5.1 Introduction

Nearly most of the previous research works only focused on network topology and ignored some critical factors like multiple product advertisement, users engagement across networks, different channels of interaction, and network of networks etc. For example, consider a

scenario that an advertising company wants to promote multiple products across multiple social networks simultaneously. In real-world, distinct users have different influence or interest for distinct product in social networks. Thus, each user has different influence probabilities from their neighbors. Nowadays, many users in networks are actively involved in multiple networks simultaneously. Therefore, *overlapping users*[1] play an essential role in information spreading across networks. Thus, the study of IM problem in each network separately underestimates the social influence of overlapping users in other networks. Hence, an advertising company needs a promotional strategy that seed users can advocate multiple products together and non-seed users can accept different products across networks simultaneously.

Correspondingly, this chapter proposes a framework multiple influence maximization across multiple social networks (MIM2) for the above scenario. Compared to traditional IM approaches, multiple influence maximization across multiple social networks introduces several new challenges:

1 How to fix the number of items for each product from budget k and how to finds seed users S, $|S| \leq k$.

2 How to identify overlapping users on different social networks and how to coupled these networks into a single one. Also, how to

[1] Users who actively engage in multiple networks simultaneously.

measure the social influence of overlapping and non-overlapping users accurately in a coupled network for each product.

3 How to decide, for which product a user is easily influenced and in which network. Also, in which network information is propagated better.

To tackle these challenges, a novel algorithm is presented to illustrate multiple influence maximization across multiple social networks. The major contributions of the chapter are as follow.

- Dealing with fourth objective, a novel algorithm MIM2 is presented for influence maximization across multiple social networks over multiple products simultaneously.
- Proved that MIM2 problem is NP-hard and expected influence spread $\sigma(S)$ is sub-modular under traditional diffusion models.
- A model representation for MIM2 problem equivalent to classical IM problem is presented. Also, A coupling strategy is presented to form a multiplex network from multiple single networks. The proposed coupling methods can be applied to traditional diffusion models.
- The proposed algorithm finds seed set across multiple networks with multiple products simultaneously and also fixes the budget for each product to maximize influence spread.

- The empirical analysis on real-world networks validate the efficiency and influence spread of the proposed algorithm against the compared methods. It also covers the advantage of the proposed framework over classical IM problem.

5.1.1 Graph Notations

Suppose, there are l graphs $G_1, G_2, \ldots, G_l$, each of the graph is represented as $G_i(V_i, E_i, W_i)$, $1 \leq i \leq l$. The set V_i, E_i, and W_i denote the vertex set, edge set, and strength of their relationship respectively in graph G_i. The $N^i_{in}(x)$ and $N^i_{out}(x)$ represent the set of incoming and outgoing neighbors of a user x in network G_i respectively. We modeled these l networks into a single multiplex network $G(V, E, W)$ by a coupling strategy through overlapping users to formulate MIM2 problem. The overlapping users can be identified using methods in [163–165]. Then, we construct a graph $G^i(V, E, W^i)$, $1 \leq i \leq m$ from G for each product $i \in m$ to propagate each product influence independently.

5.1.2 Influence Propagation Model

The algorithm incorporated classical diffusion models to compute the influence spread of seed set S in multiplex network G. First, we start the computation of influence spread for IC model. Similar to [11], there are several paths $path_{x,y}$ exist between a pair of nodes x and y. Let $P(path^j_{x,y})$

represents the influence of an individual x to y along a path $path^j_{x,y} = (x = y_1, y_2, \ldots, y_t = y)$, $path^j_{x,y} \in path_{x,y}$, given as follows.

$$P(path^j_{x,y}) = \prod_{i=1}^{i=t-1} p(y_i, y_{i+1}) \tag{5.1}$$

where $p_{x,y}$ is the influence probability of x to adjacent node y. The influence probability $P(x,y)$ of non-adjacent nodes x to y based on assumption that influence spreads independently along different paths, is given as follows.

$$P(x,y) = 1 - \prod_{path^j_{x,y} \in path_{x,y}} (1 - P(path^j_{x,y})) \tag{5.2}$$

Now, we can estimate the influence spread of seed set S on the network under IC. The influence spread of node x can be computed by sum up the influence of each reachable path from node x, given as follows.

$$\sigma(x) = 1 + \sum_{y \in V \setminus x} P(x,y) \tag{5.3}$$

Similarly, we can compute $\sigma(S)$, as follows.

$$\sigma(S) = |S| + \sum_{y \in V \setminus S} P(S,y) \tag{5.4}$$

Theorem 5.1. *The expected influence spread function $\sigma(S)$ of MIM2 is sub-modular under traditional diffusion models.*

Proof. The proof of Theorem 5.1 is given in Appendix A.2. □

Theorem 5.2. *The MIM2 problem under traditional influence diffusion models is NP-hard.*

Proof. The proof of Theorem 5.2 is given in Appendix A.2. □

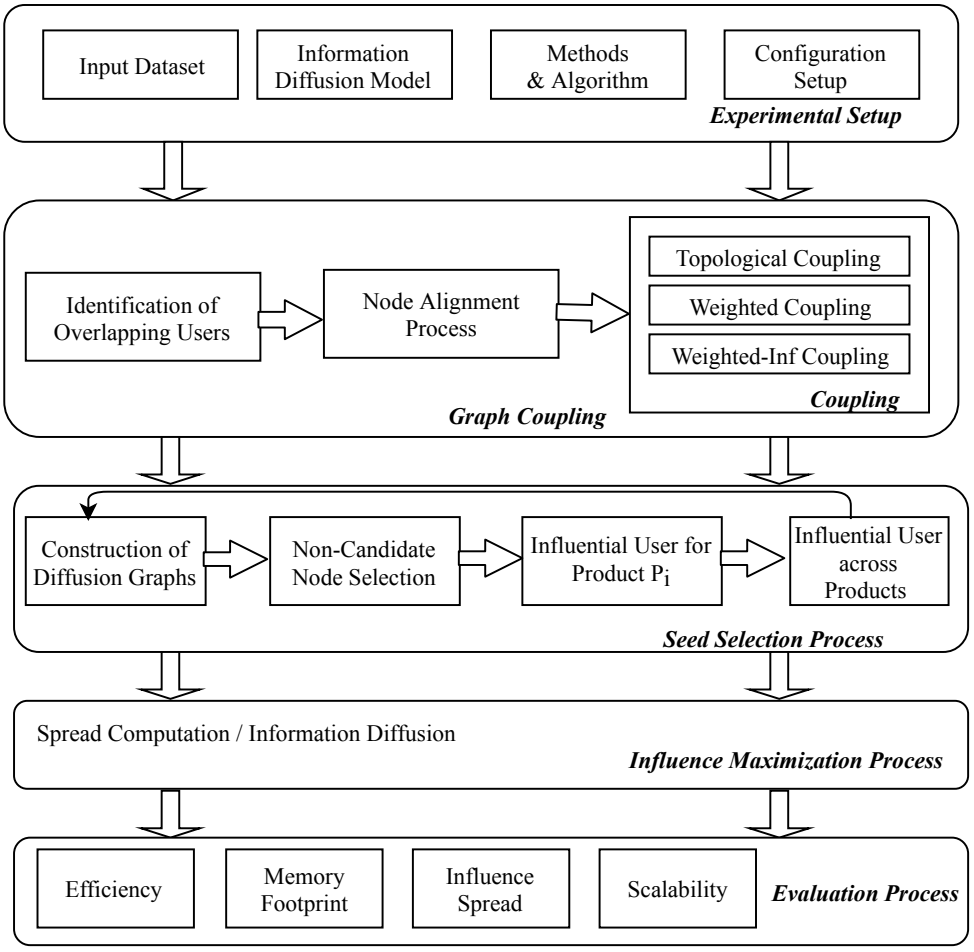


FIGURE 5.1: The Proposed MIM2 Framework

5.2 Proposed Approach

Suppose, an advertising company wants to select a small set of influential users who interact among a finite set of users through different channels

of interaction. For example, the Twitter network consists of three types of interaction networks, such as reply, re-tweet, and mention networks. Each interaction network has the same set of users with the different channel of interaction as shown in FIGURE 5.2. FIGURE 5.1 presents the proposed MIM2 framework. Let graph $G = (V, E, L)$ represents a complex social network with the different type of relationships, where V is a finite set of users, E is a finite set of relationship between a pair of users with L channels of interaction. The **Algorithm 8** explained the proposed MIM2 framework.

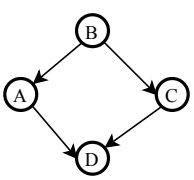
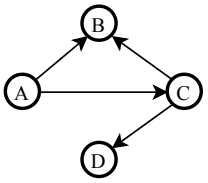
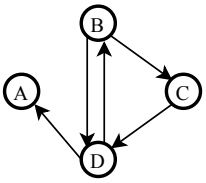
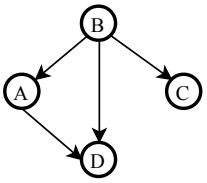
	Follower Network	Retweet Network	Reply Network	Mention Network
Network Structure				
Adjacency Matrix	$\begin{pmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$

FIGURE 5.2: Example Graph of Twitter Network: Follower (FN) Network with Three Type of Interaction Networks (Retweet (RTN), Reply(REN), and Mention (MEN) Networks)

5.2.1 Identification of Overlapping Users

MIM2 problem considers users with accounts on multiple social networking sites like Facebook, Twitter, etc., and it brings the opportunity to account the user's influence across multiple networks. This leads to the generation of more effective seed nodes. To perform information

diffusion across multiple networks simultaneously, we need to find users who actively involved in multiple networks. These users are useful in graph coupling and known as overlapping or identical users. The identification of overlapping users is a challenging task because of the diversification of users information across networks and also network data is noisy, big, unstructured and incomplete. The overlapping users can be identified using methods present in [163–165]. The identification of overlapping users is not in the focus of this paper. We consider an assumption that a user has a single account in a network.

5.2.2 Node Alignment Process

Node alignment is the process of reassigning a *universal identification number* (Uid) to every node u in each network so that every overlapping user has the same Uid across networks. In general, every network has its own node naming system. Therefore, it might be possible that a user has different identification in the different networks. As a consequence, we need to take care of user states across networks repeatedly and it results in extra effort and a complicated mechanism. Therefore, We ease this problem by reassigning a universal Uid to each individual in each network.

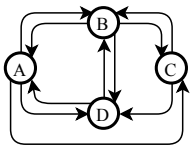
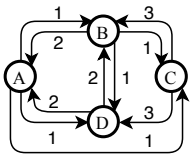
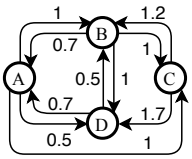
	Topological Coupling	Weighted Coupling	Weighted-influence Coupling
Coupled Multiplex Network			
Multiplex Adjacency Matricx	$\begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & 1 & 1 & 1 \\ 2 & 0 & 1 & 1 \\ 0 & 3 & 0 & 3 \\ 2 & 2 & 0 & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & 1 & 1 & 0.5 \\ 0.7 & 0 & 1 & 0.5 \\ 0 & 1.2 & 0 & 1.7 \\ 0.7 & 1 & 0 & 0 \end{pmatrix}$
Normalized Adjacency Matricx	$\begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & 0.3 & 0.3 & 0.3 \\ 2 & 0 & 0.3 & 0.3 \\ 0 & 1 & 0 & 1 \\ 0.6 & 0.6 & 0 & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & 0.5 & 0.5 & 0.2 \\ 0.4 & 0 & 0.5 & 0.2 \\ 0 & 0.7 & 0 & 1 \\ 0.4 & 0.5 & 0 & 0 \end{pmatrix}$

FIGURE 5.3: The Coupling of Networks

5.2.3 Graph Coupling Scheme: Direct Linkage

In the linkage strategy, each relationship graph $L_i \in L$ is considered as an individual network. In order to transform these relationship networks into a single multiplex network $G = (V, E, W)$, we need to aggregate all relationships between each pair u and v to form a single relationship. Let us consider a complex system $G = (V, E, L)$, $L = \{L_1, L_2, \ldots, L_l\}$ with l types of relationship. Each interaction network $L_i \in L$ is integrated with an adjacency matrix $A^{L_i} = \{a_{uv}^{L_i}\}$. Therefore, such a complex system can be expressed as $A = [A^{L_1}, A^{L_2}, \ldots, A^{L_l}]$. Similarly, the degree of an individual u can be represented as a vector $D(u) = [D^{L_1}(u), D^{L_2}(u), \ldots, D^{L_l}(u)]$, where $D^{L_i}(u) = \sum_{v \in V} a_{uv}^{L_i}$. Both vector A and $D(u)$ are requisite to represent the complex system correctly. FIGURE 5.2 shows an example Twitter graph with four types of relationship and also

presents their corresponding adjacency matrices. From FIGURE 5.2, we can see that $l = 4$ and $D(A) = [1,2,0,1]$.

In order to aggregate these relationships to a single relationship, the direct linkage strategy presents two types of adjacency matrix: topological and weighted adjacency matrices. The topological adjacency matrix ignores the fact that different type of relationships may present in such a network between a pair of individuals. It ignores the possibility of multi-link and it shows a connection between u and v if a link present in any one of the interaction network. The topological adjacency matrix $A_{TMN} = \{a_{uv}\}$ of a coupled multiplex network is defined as similar to [166], where

$$a_{uv} = \begin{cases} 1 & if \quad \exists a_{uv}^{L_i} = 1, L_i \in L \\ 0 & \text{otherwise} \end{cases} \tag{5.5}$$

For example, FIGURE 5.3 shows the topological multiplex network of example graph as shown in FIGURE 5.2 with their corresponding adjacency matrix.

The weighted adjacency matrix of coupled multiplex network considers the possibility of multi-link between a pair of individual. The weighted adjacency matrix $A_{WMN} = \{w(u,v)\}$ is defined as similar to [166], where

$$w(u,v) = \sum_{L_i \in L} a_{uv}^{L_i}; 0 \leq w(u,v) \leq l, \forall u,v \in V \tag{5.6}$$

For example, FIGURE 5.3 shows the weighted multiplex network of

example graph as shown in FIGURE 5.2 with their corresponding weighted adjacency matrix.

To construct real dynamics of information propagation in a social network, we need to identify which type of relationship is best suited for information diffusion and which is worst. For this purpose, we uses a probability vector $\alpha = [\alpha^{L_1}, \alpha^{L_2}, \ldots, \alpha^{L_l}]$ to represent importance of each type of relationship. Therefore, weight index $w(u,v)$ of each link (u,v) is calculated as follows.

$$w(u,v) = \sum_{L_i \in L} a_{uv}^{L_i}.\alpha^{L_i}; 0 \leq \alpha^{L_i} \leq 1, 0 \leq \sum_{L_i \in L} \alpha^{L_i} \leq l \tag{5.7}$$

For example, let us consider that probability vector $\alpha = [\alpha^{SN}, \alpha^{RTN}, \alpha^{REN}, \alpha^{MTN}] = [0.2, 1, 0.5, 0.5]$. FIGURE 5.3 shows the weighted-influence multiplex network with probability vector and corresponding weighted matrix. We need to perform normalization of edge weight $w(u,v) \in [0,1]$ of every individual u and v in weighted and weighted-influence coupling. The overall time complexity of the coupling scheme is $O(\sum_{i=1}^{i=l}(|V_i| + |E_i|))$.

5.2.4 Multiple Influence Maximization

In this section, we discuss the IM process for multiple products in a multiplex network achieved from the above discussed coupling strategy.

5.2.4.1 Generation of product diffusion graph

To maximize the influence spread in the multiplex network for m products simultaneously, MIM2 constructs a social graph corresponding to each product. The edge weight in the social graph of each product follow a probability distribution for information diffusion and all graphs share the same set of users. The influence propagation for each product is independent in the network. The objective of MIM2 is to select k seed nodes from these weighted product diffusion graphs to maximize the overall influence spread of products. The overall complexity of generating m diffusion graph is $O(m(|V|+|E|))$.

5.2.4.2 Finding non-candidate nodes

The non-candidate nodes are those who less likely to be influential in the network. We find the fixed percentage of non-candidate nodes from the network to trace back to maximum influencing seeds using the reverse graph. To identify non-candidate nodes, first, we remove all the edges with weight less than $w\%$ of the average weight of the network. Second, assign the estimated influence of nodes already selected as seed equal to infinity. Next, assign the estimated influence of remaining nodes equal to the sum of weights of outgoing edges. Finally, we select t least weighted nodes as non-candidate. The overall time complexity of finding non-candidate set in a graph is $O(|E|+|V|\log|V|)$.

5.2.4.3 Identifying most influential user for product P_i

To identify the most influential user for a product P_i, MIM2 selects the corresponding product diffusion graph and performs backward propagation to find the most influential user. First, the algorithm selects non-candidate nodes and then it traverses the graph in reverse order from these nodes using Breadth First Search (BFS) to accumulate the expected influence spread of each node. Finally, the algorithm selects the node with the maximum expected influence as the most influential user for the the corresponding product. The time complexity of finding the most influential user over a graph is $O(|E| + |V| \log |V|)$.

5.2.4.4 Identifying most influential user among all product diffusion graph

The algorithm finds seed nodes by selecting most influential node among all product diffusion graphs iteratively. To find most influential user among all product, it selects most influential user for each product P_i, $1 \leq i \leq m$ by reverse tracing using BFS and compare with each other to find maximal spread node. After finding most influential node, add it to seed set and delete this node from the corresponding product diffusion graph. This process is continued until k nodes are selected.

5.2.4.5 Influence estimation

To evaluate the performance of the proposed algorithm, we need to compute the influence spread of selected seed nodes in the coupled multiplex network. We use the same propagation strategy as discussed in Section 5.1.2 under traditional diffusion models.

Algorithm 8: MIM2 (G_i, k, l, m): The Proposed Algorithm

Input: $G_i(V_i, E_i, W_i)$, $1 \leq i \leq l$: Social graphs, k: Size of the seed set, l: Number of social graphs, m: Number of products
Output: S: Seed set

$S \leftarrow \phi$
$Remaining \leftarrow k$
$G \leftarrow$ Apply one of the coupling strategy using **Algorithm 9**, **10**, or **11**
Find product diffusion graph G^i of multiplex graph G for each product i, $1 \leq i \leq m$
while $Remaining > 0$ **do**
    $seed\_node \leftarrow \text{NextSeedB}(G^i, update_i, Acc_i, seed_i)$ ▷ See **Algorithm 14**
    $seed\_node.Graph.update \leftarrow 0$
    $seed\_node.deleted = True$
    $S[seed\_node.Graph].insert(seed\_node)$
    $Remaining \leftarrow Remaining - 1$
Return S

Algorithm 9: Topological_Coupling$(G_i(V_i, E_i, W_i), l)$

Input: $G_i(V_i, E_i, W_i)$, $1 \leq i \leq l$: Social graphs
Output: $G(V, E, W)$: Topological multiplex social graph

$V\_set \leftarrow \phi$
$Edge\_set \leftarrow \phi$
$Weight \leftarrow \phi$
for *each graph* G_i, $1 \leq i \leq l$ **do**
    for *each edge* $(u, v) \in E_i$ **do**
        $Edge\_set \leftarrow Edge\_set \cup (u, v)$
        $V\_set \leftarrow V\_set \cup \{u, v\}$
for *each edge* $(u, v) \in Edge\_set$ **do**
    $Weight[u, v] \leftarrow 1$
Return $G(V\_set, Edge\_set, Weight)$

Algorithm 10: Weighted_Coupling$(G_i(V_i,E_i,W_i),l)$

Input: $G_i(V_i,E_i,W_i)$, $1 \leq i \leq l$: Social graphs
Output: $G(V,E,W)$: Weighted multiplex social graph

$V\_set \leftarrow \phi$
$Edge\_set \leftarrow \phi$
$Weight \leftarrow \phi$
for *each graph* G_i, $1 \leq i \leq l$ **do**
  for *each edge* $(u,v) \in E_i$ **do**
    if $(u,v) \in Edge\_set$ **then**
      $Weight[u,v] \leftarrow Weight[u,v] + 1$
    else
      $Edge\_set \leftarrow Edge\_set \cup (u,v)$
      $V\_set \leftarrow V\_set \cup \{u,v\}$
      $Weight[u,v] \leftarrow 1$
$Max\_weight \leftarrow max(Weight)$ ▷ Normalize edge weight
for *each edge* $(u,v) \in Edge\_set$ **do**
  $Weight[u,v] \leftarrow \frac{Weight[u,v]}{Max\_weight}$
Return $G(V\_set, Edge\_set, Weight)$

Algorithm 11: Weighted_Influence_Coupling(G_i,l,α)

Input: $G_i(V_i,E_i,W_i)$, $1 \leq i \leq l$: Social graphs, α: Probability vector
Output: $G(V,E,W)$: Weighted influence multiplex graph

$V\_set \leftarrow \phi$
$Edge\_set \leftarrow \phi$
$Weight \leftarrow \phi$
for *each graph* G_i, $1 \leq i \leq l$ **do**
  for *each edge* $(u,v) \in E_i$ **do**
    if $(u,v) \in Edge\_set$ **then**
      $Weight[u,v] \leftarrow Weight[u,v] + \alpha^{L_i}$
    else
      $Edge\_set \leftarrow Edge\_set \cup (u,v)$
      $V\_set \leftarrow V\_set \cup \{u,v\}$
      $Weight[u,v] \leftarrow \alpha^{L_i}$
$Max\_weight \leftarrow max(Weight)$ ▷ Normalize edge weight
for *each edge* $(u,v) \in Edge\_set$ **do**
  $Weight[u,v] \leftarrow \frac{Weight[u,v]}{Max\_weight}$
Return $G(V\_set, Edge\_set, Weight)$

Algorithm 12: Finding_Non_Candidates(G_i, t, w)

Input: $G_i(V_i, E_i, W_i)$: Social graph, w: Threshold weight percentage, t: Number of non-candidate nodes

Output: Non_Cand: Non-candidate nodes

1 $Sum \leftarrow 0$
2 **for** *each edge* $(u,v) \in E_i$ **do**
3 $\quad Sum \leftarrow Sum + W_i[u,v]$
4 $Average \leftarrow \frac{Sum}{|E|}$
5 $Threshold \leftarrow Average * w$
6 $Est\_inf \leftarrow$ Initialize estimated influence of each node to 0
7 **for** *each edge* $(u,v) \in E_i$ **do**
8 $\quad$ **if** $W_i[u,v] > Threshold$ **then**
9 $\quad\quad Est\_inf[u] \leftarrow Est\_inf[u] + W_i[u,v]$
10 $V\_Sort \leftarrow G_i.V_i$ in ascending order with respect to estimated influence Est_inf
11 $Non\_Cand \leftarrow$ Select first t nodes of V_Sort as non-candidate nodes
12 **Return** Non_Cand

5.3 Algorithm

In this section, we provide a detailed description of each algorithm. The main **Algorithm 8** MIM2 takes four inputs, l social graphs, budget k, number of graphs l, and number of products m. Line 1 initializes seed set S with an empty set. Line 2 assigns *Remaining* to budget k. Line 3 performs coupling on l social graphs and gets a multiplex network G. Line 4 constructs product diffusion graph using normal distribution. Lines 5-10 iteratively finds the most influential node across diffusion graph and adds it to seed set S. Line 11 returns the seed set S as the final output.

The **Algorithm 9** Topological_Coupling takes l influence graphs as input and it returns a coupled multiplex network based on topological structure as output. First of all, lines 1-3 assign an empty set to V_set, $Edge\_set$,

Algorithm 13: NextSeedA($G^i, update, Acc, seed$)

Input: $G_i(V,E,W^i)$: Social graph, $update$: Vector indicated whether influence value is updated, Acc: Vector store accumulated influence so far, $seed$: Current seed set so far

Output: x: Most influential seed

```
1  if G^i.update = 1 then
2      Return max(Acc)
3  Nds ← Finding_Non_Candidates(G^i, t, w)
4  Queue Q
5  for each node u ∈ Nds do
6      Q.push(u)
7      u.visited ← 1
8  Initialize Acc value of each node to 1
9  while (!Q.empty()) do
10     r ← Q.front()
11     Q.pop()
12     for each edge (v, r) ∈ E do
13         x ← v
14         if x ∉ seed then
15             if !(x.visited = 1) then
16                 x.visited = 1
17                 Q.push(x)
18             Acc[x] ← Acc[x] + Acc[r] * W^i[v, r]
19 G^i.update ← 1
20 Return max(Acc)
```

Algorithm 14: NextSeedB($G^i, update_i, Acc_i, seed_i$)

Input: $G^i(V,E,W^i)$, $1 \le i \le m$: Social graphs, $update_i$: Vector indicated whether influence value is updated, Acc_i: Accumulated influence so far, $seed_i$: Current seed set so far

Output: x: Most influential seed among all m-graphs

```
1  x ← Null
2  for each graph G^i, 1 ≤ i ≤ m do
3      x ← max(x, NextSeedA(G^i, update_i, Acc_i, seed_i))      ▷ See Algorithm 13
4  Return x
```

and *Weight*. The **for** loop in lines 4-7 add edge and vertex of individual graphs in $Edge\_set$ and V_set. The **for** loop in lines 8-9 perform coupling

based on their topological structure. Line 10 returns a coupled multiplex network.

The **Algorithm 10** Weighted_Coupling takes l influence graphs as input and it returns a coupled multiplex network based on their weighted topological structure as output. Lines 1-3 assign an empty set to V_set, $Edge\_set$, and $Weight$. Lines 4-11 perform coupling based on their weighted topological structure by considering the different type of relationships. Line 12 finds the maximum possible weight of an edge. Line 13-14 performs normalization of edge weight. Line 15 returns a coupled multiplex network.

The **Algorithm 11** Weighted_Influence_Coupling takes l influence graphs as input and it returns a coupled multiplex network based on their weighted topological structure as output. Lines 1-3 assign an empty set to V_set, $Edge\_set$, and $Weight$. Lines 4-11 perform coupling based on their weighted influence by considering their different diffusion behavior. Line 12 finds the maximum possible weight of an edge. Line 13-14 performs normalization of edge weight. Line 15 returns a coupled multiplex network.

The **Algorithm 12** Finding_Non_Candidates takes three inputs, a social graph G_i, threshold weight percentage w, and the number of non-candidates t. Line 1 initializes the *Sum* with 0. Lines 2-3 calculate the sum of influence weight of each edge in the graph G_i. Line 4 estimates the average influence of an edge in the graph. Line 5 sets a threshold value for

the selection of non-candidate nodes. Line 6 initializes the estimated influence Est_inf of each node u to 0. Lines 7-9 compute Est_inf for each node u iteratively. Line 10 performs sorting in ascending order based on Est_inf. Line 11 selects t non-candidate seed nodes and store in a vector Non_Cand. Line 12 returns the Non_Cand as an output.

The **Algorithm 13** NextSeedA takes four inputs, a social graph G_i, update status vector $update$, accumulated influence vector Acc, and current seed set $seed$. Line 1-2 checks that the influence values of nodes are already updated or not. Line 3 calls Finding_Non_Candidates algorithm and finds non-candidate nodes Nds. Line 4 maintains a queue Q. Lines 5-7 marks non-candidate nodes as visited and push Nds nodes into Q. Line 8 initializes accumulated influence Acc of each node to 1. Line 9-19 updates Acc iteratively for each node. Line 10-11 pop the node from the front and store it to r, if queue Q is not empty. Lines 12-18 updates Acc iteratively based on each edge. Line 19 marks the graph as updated. Line 20 returns maximum accumulated influence node as most influential node in the graph.

The **Algorithm 14** NextSeedB takes four inputs, m social graphs, update status vector $update_i$, accumulated influence vector Acc_i, and current seed set $seed_i$ for each graph G_i. Line 1 initializes most influential node x to null. Lines 2-3 find the most influential user across m social graphs and store it to x. Line 4 returns x as most influential user.

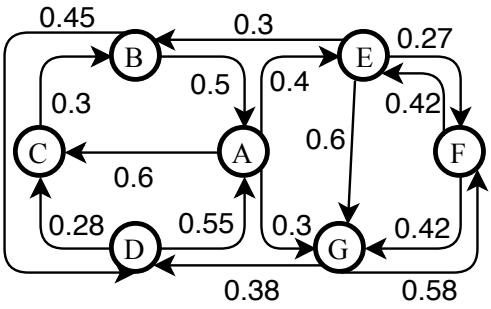


FIGURE 5.4: A Running Example

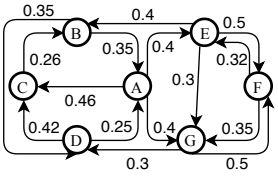
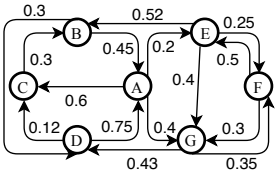
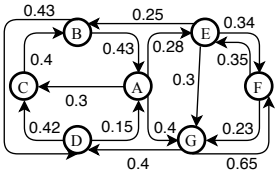
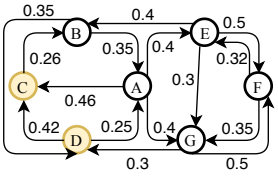
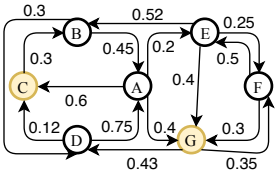
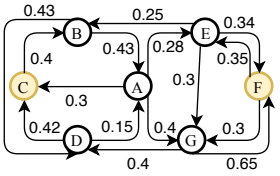
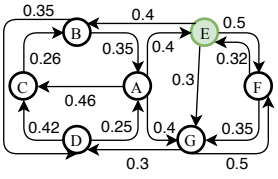
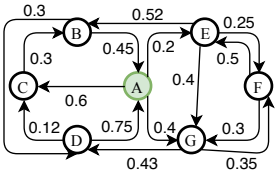
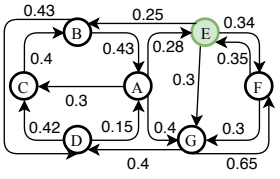
Product	P_1	P_2	P_3
Product Diffusion Graphs			
Non Candidate Nodes			
Seed Nodes Selection			

FIGURE 5.5: The Working of Seed Selection Process in MIM2 Framework

5.3.1 Applying the Algorithm

In this section, we present a running example to explain the working of the proposed algorithm for MIM2 problem. Let consider a coupled multiplex network and $m = 3$, which is constructed using the weighted-influence coupling scheme and shown in FIGURE 5.4. In the seed selection process of MIM2, first, we construct the product diffusion

graphs for each product P_i. Let assume that the product diffusion graphs are constructed using normal distribution and given in FIGURE 5.5. Next, MIM2 selects non-candidate nodes in each product diffusion graph G^i. Each edge (u,v) in G^i has edge weight $W^i[u,v] \leftarrow W_i[u,v] * W_D^{P_i}[u,v]$, where $W_i[u,v]$ and $W_D^{P_i}[u,v]$ denote edge weight of edge (u,v) in the coupled multiplex graph and normal distribution edge in diffusion graph of product P_i. To find non-candidate nodes, we consider $w = 80\%$, $t = 2$ and calculates expected influence $Average$ for P_1 as $Average \leftarrow Sum/|E|$, i.e. $Average \leftarrow ((0.4+0.4+0.46)+(0.35+0.35)+(0.26)+(0.25+0.42)+(0.5+0.4+0.3)+(0.32+0.35)+(0.5+0.3))/15 = 0.37$. Before calculating estimated influence $Est\_Inf[u]$ of a node u, we compute threshold value $Threshold \leftarrow 0.37 * 0.8 = 0.29$. Now, we compute $Est\_Inf[u]$ of each node u. The Est_Inf is calculated as $Est\_Inf \leftarrow [0.4+0.4+0.46, 0.35+0.35, 0, 0.42, 0.5+0.4+0.3, 0.32+0.35, 0.5+0.3] = [1.26, 0.7, 0, 0.42, 1.2, 0.67, 0.8]$. Next, we select least-$t$ nodes based on Est_Inf value as non-candidate nodes N_C, i.e., $N_C \leftarrow \{C,D\}$ for product diffusion graph G^1. Let non-candidate nodes for graph G^2 and G^3 are $\{C,G\}$ and $\{C,F\}$ respectively. The yellow colored nodes represent non-candidate nodes as in FIGURE 5.5.

Next, MIM2 selects seed by reverse tracing using BFS and assume budget $k = 3$. To select the most influential node in graph G^1, initially, queue Q has N_C nodes, i.e. $Q \leftarrow \{C,D\}$ and marks these node as visited. The accumulated influence Acc for each node is 1 at start of reverse tracing.

TABLE 5.1: The Computation of Accumulated Influence Acc based on Algorithm NextSeedA for Product Diffusion Graph G^1

Queue Q	$< Acc, visited >$ value						
	A	B	C	D	E	F	G
ϕ	1,0	1,0	1,0	1,0	1,0	1,0	1,0
$\{C,D\}$	1,0	1,0	1,1	1,1	1,0	1,0	1,0
$\{D,A\}$	1.46,1	1,0	1,1	1.42,1	1,0	1,0	1,0
$\{A,B,G\}$	1.46,1	1.49,1	1,1	1.42,1	1,0	1,0	1.42,1
$\{B,G\}$	1.46,1	2,1	1,1	1.78,1	1,0	1,0	1.42,1
$\{G,E\}$	1.46,1	2,1	1.52,1	1.78,1	1.8,1	1,0	1.42,1
$\{E,F\}$	2.02,1	2,1	1.52,1	1.78,1	2.22,1	1.49,1	1.42,1
$\{F\}$	2.9,1	2,1	1.52,1	1.78,1	2.22,1	2.2,1	1.42,1
ϕ	2.9,1	2,1	1.52,1	1.78,1	3.32,1	2.2,1	2.52,1

The accumulated influence $Acc \leftarrow [2.9, 2, 1.52, 1.78, 3.32, 2.2, 2.52]$ of G^1 is computed based on algorithm NextSeedA as shown in TABLE 5.1. Therefore, most influential user in G^1 is E. We assume that most influential user for graph G^2 and G^3 are A and E respectively. Using algorithm NextSeedB, MIM2 selects most influential user among all products is E for P_1. Therefore, E is added to seed set S and removes from G^1. Similarly, remaining seed nodes are selected. Node E and A are selected from graph G^3 and G^2 respectively as seed. The green colored nodes represent seed nodes in the respected diffusion graph, as shown in FIGURE 5.5. Finally, MIM2 procedure combines the all seed set S_i for G_i. The overall seed set $S \leftarrow \bigcup_{i=1}^{i=m} = \{E\} \cup \{A\} \cup \{E\} = \{A, E\}$, $\sum_{i=1}^{i=m} |S_i^*| = k = 3$ and $|S| = 2 \leq k$. FIGURE 5.6 shows the final seed nodes in green color under the MIM2 framework.

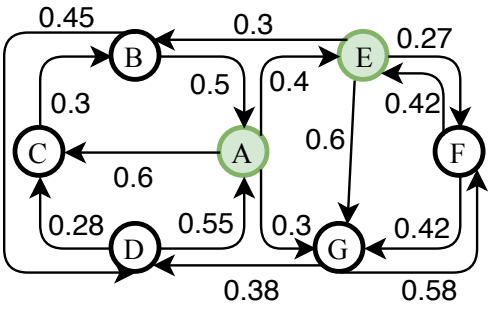


FIGURE 5.6: The Final Seed Nodes

5.3.2 Complexity Analysis

In this section, we analyze the worst-case time complexity of proposed algorithm MIM2. Lines 1-2 take $O(1)$ basic operations to initialize S and *Remaining*. Line 3 performs coupling procedure in $O(\sum_{i=1}^{i=l}(|V_i|+|E_i|))$ basic operations, i.e. $O(|V|+|E|)$, where V and E denotes the node and edge set of a multiplex network (also known as coupled network). Line 4 constructs m product diffusion graph in $O(m(|V|+|E|))$ basic operations. Lines 5-10 performs seed selection process iteratively. First, it selects non-candidate nodes using $O(\sum_{i=1}^{i=m}(|E|+|V|\log|V|))$ basic operations. Then, it finds overall seed set from MIM perspective and take $O(k(|E|+|V|\log|V|))$ basic operations. So overall complexity of lines 5-10 is $O((m+k)(|E|+|V|\log|V|))$. Therefore, the worst-case time complexity of MIM2 algorithm is $O((l+m)(M+N)+(k+m)(M+N\log N))$, where $|V|=N$ and $|E|=M$.

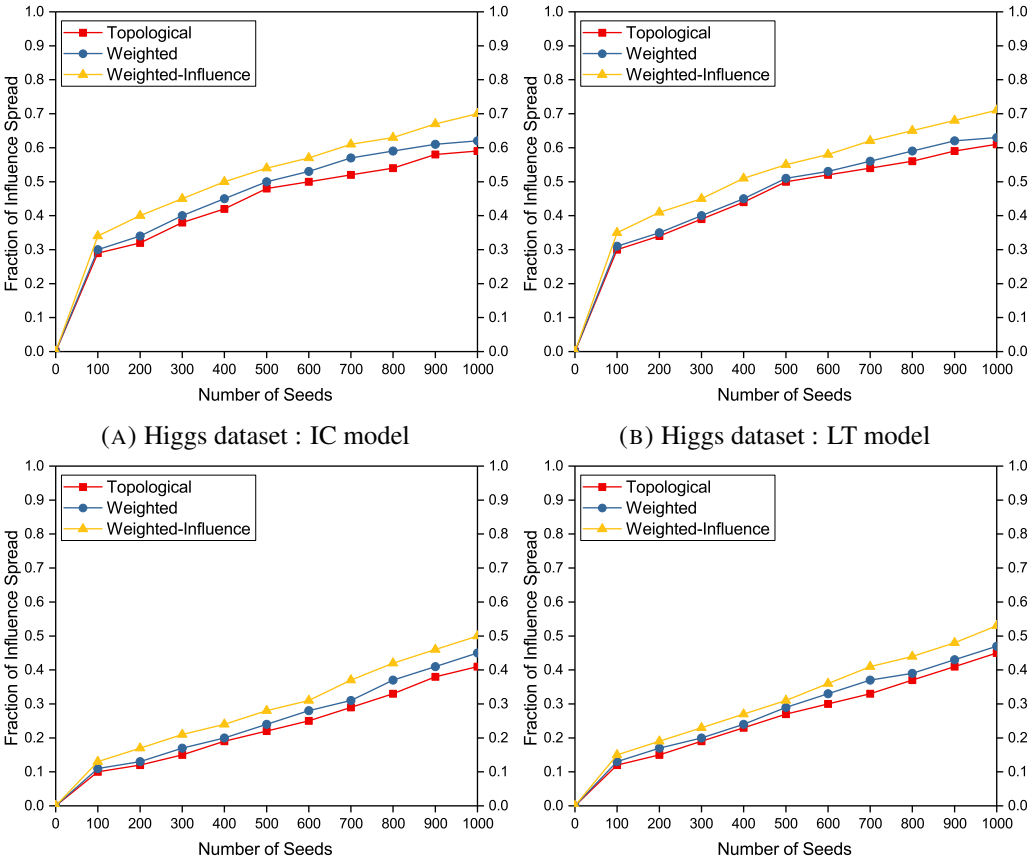
5.4 Empirical Analysis

In this section, firstly, the experimental setup information is provided along with the dataset, seeding strategies, and probability distribution for diffusion models. Further, discussion of the performance analysis regarding influence spread and efficiency along with parameter analysis are provided. Finally, section illustrate the statistical test to show the significant difference between the proposed method and the compared seeding methods.

5.4.1 Experimental Setup

All the experiments performed on two real-world network datasets: Higgs Twitter networks [145] and co-author networks [38]. The performance of proposed algorithm is tested against four seeding strategies: Random [11], MaxDegree [11], Degree Discount [24], and MIM-Greedy [37]. The algorithm utilize well-known probability distributions to assign influence probabilities. The propagation probability for each edge follows a uniform distribution. The propagation probability in IC models assigned based on uniform distribution over $\{0.1,0.01,0.001\}$. For the LT model, propagation probability $p_{x,y}$ is assigned by $\frac{1}{indegree(y)}$. The activation probability of a node x follows uniform distribution over [0,1]. Each of the methods is implemented using C++ language and executed on DEV-C++. All of the algorithms run 20 times independently on a 64-bit

window's PC with Intel(R) Core(TM) i7-7700 CPU@ 3.60GHz processor and 8GB memory.



(A) Higgs dataset : IC model

(B) Higgs dataset : LT model

(C) Co-author dataset : IC model

(D) Co-author dataset : LT model

FIGURE 5.7: The Comparison of Coupling Schemes over MIM2 Influence Spread for $m = 1$

5.4.2 Analyzing Coupling Schemes

The influence spread of MIM2 algorithm utilized to evaluate the performance of coupling schemes. FIGURE 5.7 shows that the weighted-influence coupling scheme activates more users than other two coupling schemes under traditional diffusion models for both Twitter and co-author networks. This is because of weighted-influence coupling

considers the importance of different types of relationships for information diffusion along with the topology structure of the network.

TABLE 5.2: The Influence Spread Comparison of Algorithm under IM2 Framework for Higgs Twitter Dataset at $k = 50$

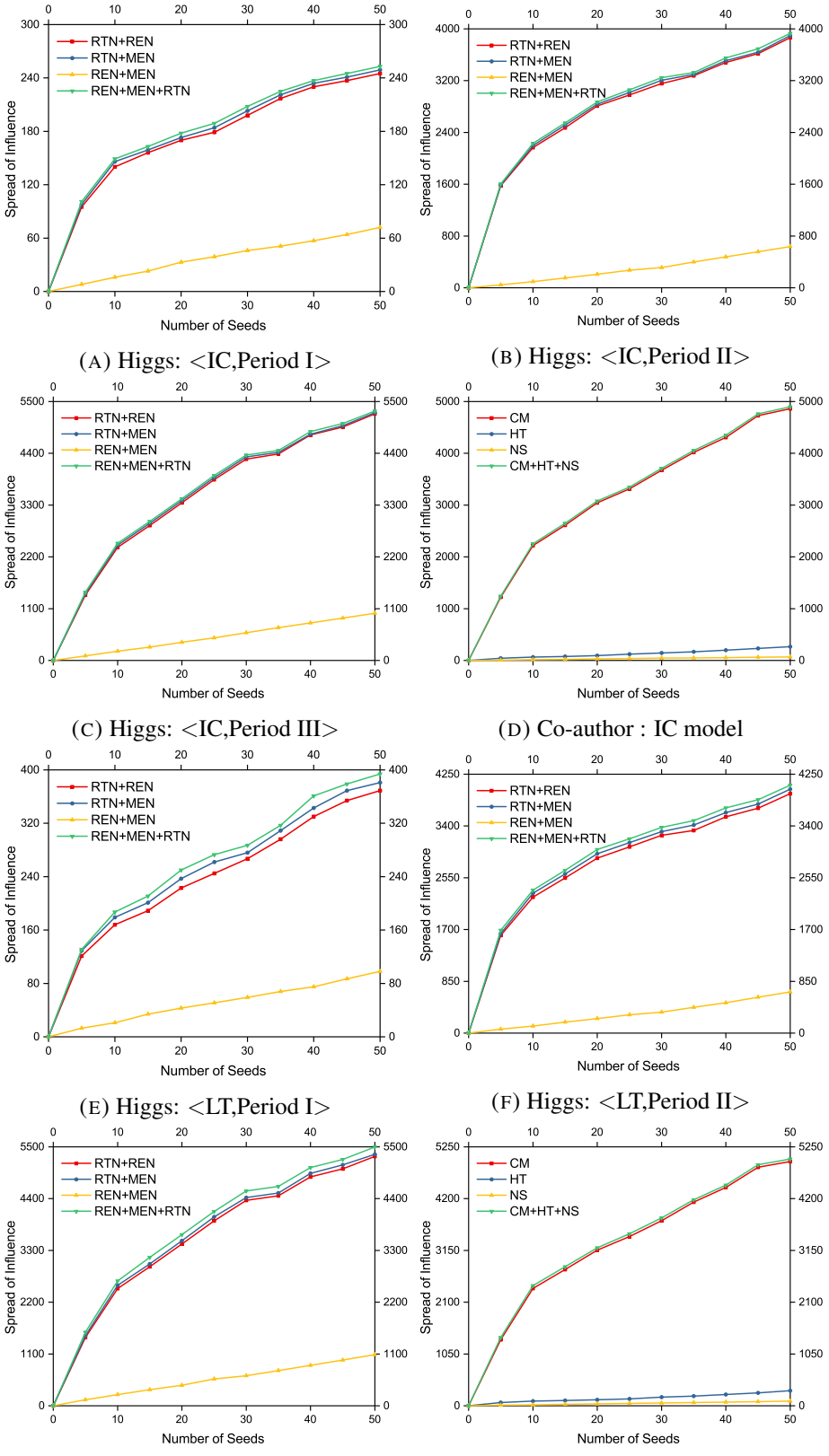
		Influence Spread			
		RTN+REN	RTN+MEN	REN+MEN	RTN+REN+MEN
IC	Period I	245	249	72	253
	Period II	3864	3896	637	3929
	Period III	5238	5262	1003	5298
LT	Period I	369	381	98	394
	Period II	3929	4004	677	4069
	Period III	5298	5342	1085	5498

TABLE 5.3: The Influence Spread Comparison of Algorithm under IM2 Framework for Co-author Dataset at $k = 50$

Diffusion	Influence Spread			
Model	CM	HT	NS	CM+HT+NS
IC	4762	265	68	4857
LT	4952	305	94	5063

5.4.3 Advantages of using IM2

In order to understand the benefit of coupled multiplex networks under IM2 framework, we compare the influence spread of proposed algorithm for different seed size over coupled networks. FIGURE 5.8 shows the effect of avoiding one type of relationship in Higgs Twitter networks and co-author networks under a traditional diffusion model. In all three periods of Higgs dataset under both diffusion models, the avoidance of RTN edges causes the most significant difference in influence spread. The influence spread significantly deteriorate when RTN edges not considered in the influence



(A) Higgs: <IC,Period I>

(B) Higgs: <IC,Period II>

(C) Higgs: <IC,Period III>

(D) Co-author : IC model

(E) Higgs: <LT,Period I>

(F) Higgs: <LT,Period II>

(G) Higgs: <LT,Period III>

(H) Co-author : LT model

FIGURE 5.8: The Comparison of Influence Spread over IM2 Framework for $m = 1$ under Traditional Diffusion Models

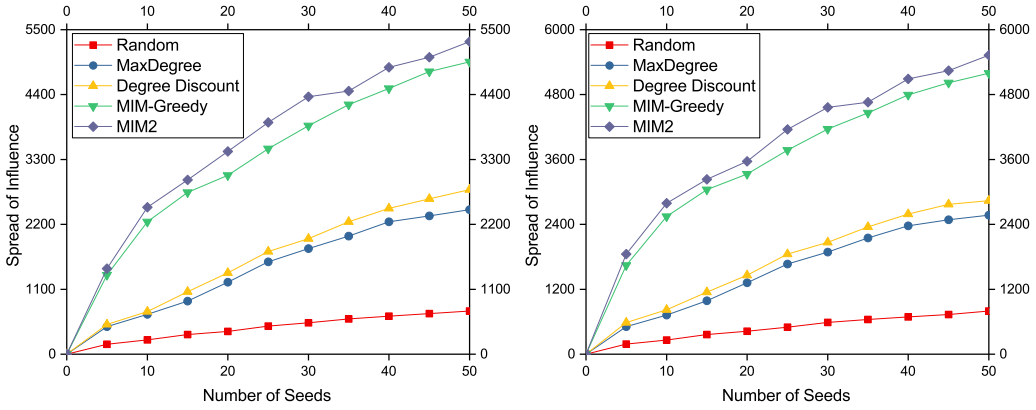
diffusion process. FIGURE 5.8 also illustrates the effect of the coupled network in co-author networks. The influence spread is very less when we consider only HT or only NS network. TABLE 5.2 and 5.3 provide the influence spread of MIM2 algorithm in coupled multiplex networks under IM2 framework at $k = 50$ for Higgs Twitter and co-author datasets respectively.

TABLE 5.4: The Influence Spread Comparison of Compared Algorithms under MIM Framework for Different m at $k = 50$

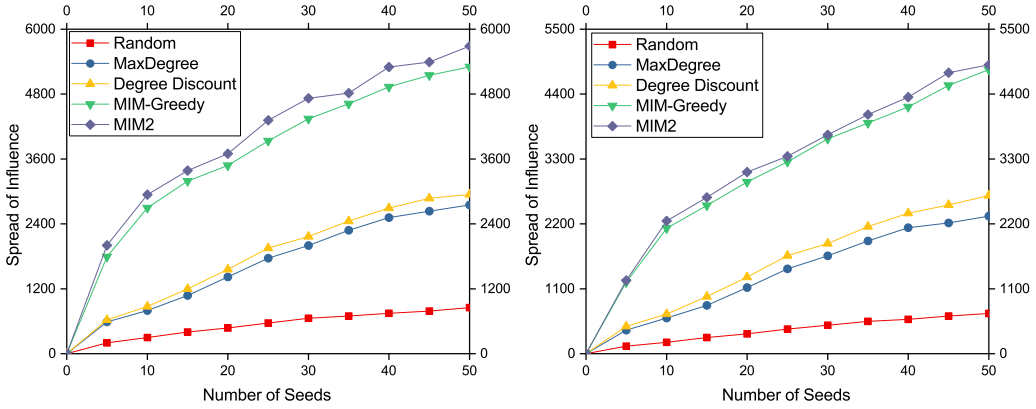
Dataset	m	Influence Spread				
		Random	MaxDegree	Degree Discount	MIM-Greedy	MIM2
Twitter	1	731	2451	2789	4954	5298
	2	798	2571	2839	5194	5528
	3	853	2751	2945	5304	5688
Co-author	1	681	2331	2679	4814	4897
	2	781	2461	2779	4974	5197
	3	831	2521	2849	5074	5297

5.4.4 Advantages of using MIM

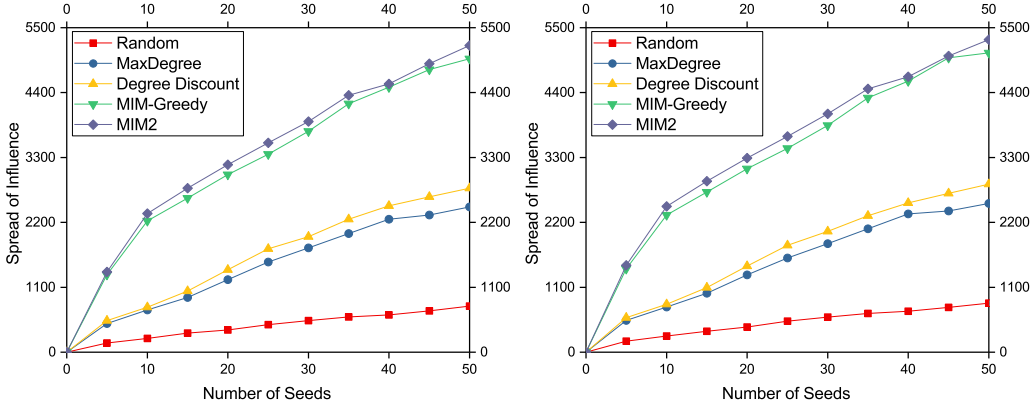
In our experiments, we assume number of products $m = 1, 2, 3$ and seed size $k = 5, 10, \ldots, 50$. FIGURE 5.9 shows the influence spread of compared algorithms for different size seed set and for different value of m under IC model. We can see that MIM2 algorithm outperforms other algorithms under MIM framework for the different number of products. This is because of Random, MaxDegree, and Degree Discount heuristic methods are only dependent on network topology and edge weight. MIM-Greedy is based on the influence matrix which considers influence



(A) Higgs dataset: $m = 1$ (B) Higgs dataset: $m = 2$



(C) Higgs dataset: $m = 3$ (D) Co-author: $m = 1$

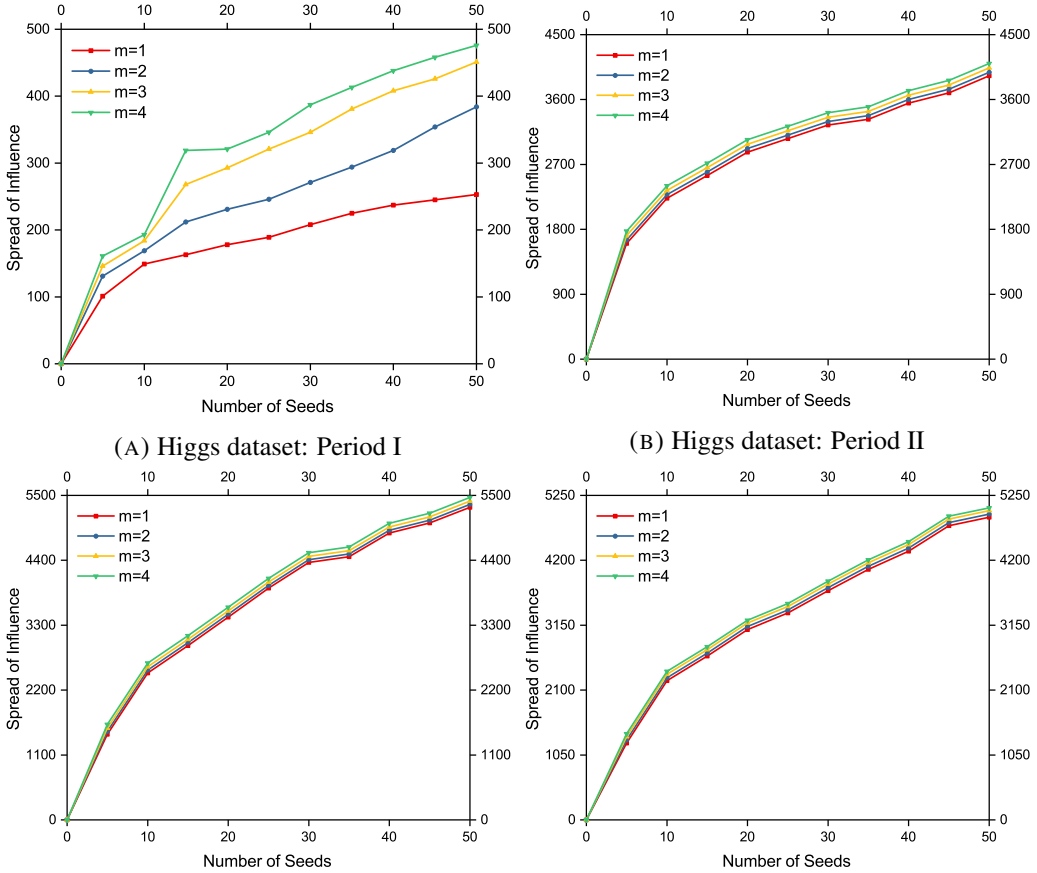


(E) Co-author: $m = 2$ (F) Co-author: $m = 3$

FIGURE 5.9: The Comparison of the Influence Spread over MIM Framework for Different Number of Product m under IC Model

up to six hope. Our MIM2 algorithm estimates the diffusion degree of each node by reverse tracing the whole network. FIGUREs 5.9a to 5.9c illustrate that the influence spread of algorithms are increases with the

increase of the number of products in Higgs dataset. Similarly, FIGUREs 5.9d to 5.9f compares the influence spread of algorithm for different value of m. From these experiments we can observe that the influence spread of MIM2 algorithm is increases with increment in number of products. This influence spread increment is highly evident until k reaches a certain threshold. This is because of the remaining nodes have less social source, in-neighbors and out-neighbors. TABLE5.4 gives the influence spread of compared algorithm under IC diffusion model for MIM framework at $k = 50$.



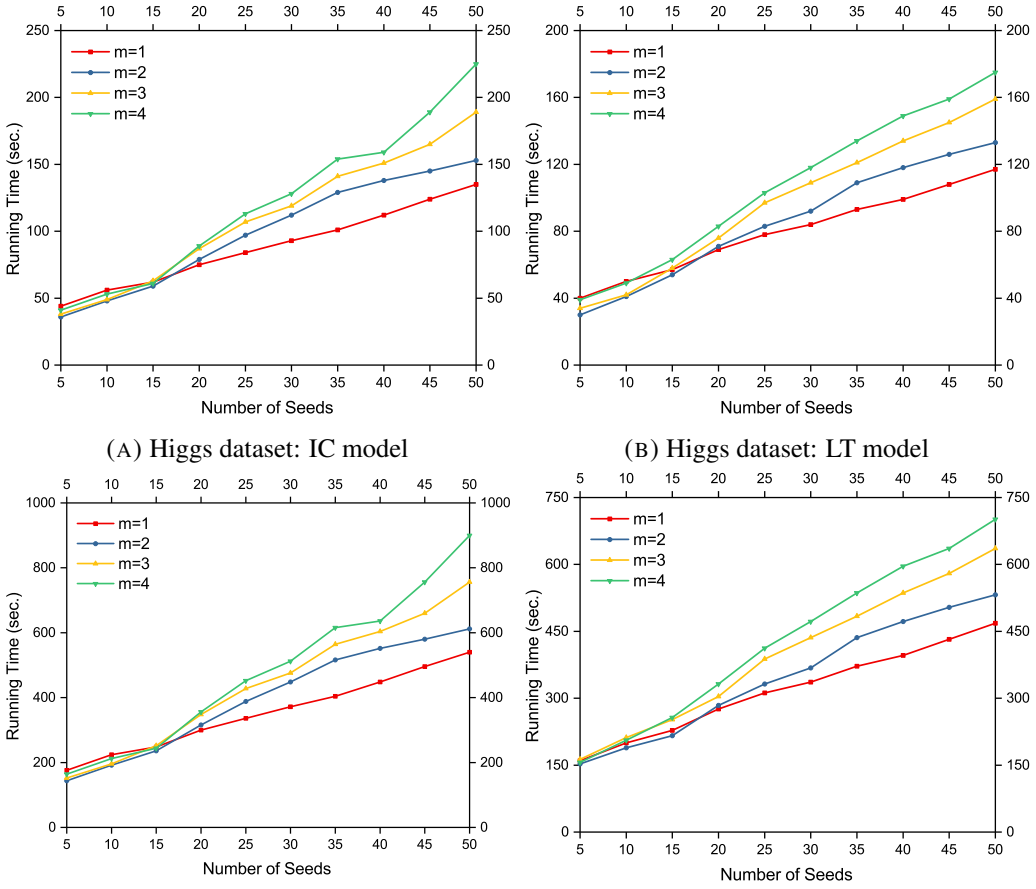
(A) Higgs dataset: Period I

(B) Higgs dataset: Period II

(C) Higgs dataset: Period III

(D) Co-author dataset

FIGURE 5.10: The Comparison of the Influence Spread of MIM2 Framework for Different Number of Product m under IC Model



(A) Higgs dataset: IC model

(B) Higgs dataset: LT model

(C) Higgs dataset: IC model

(D) Co-author: LT model

FIGURE 5.11: The Comparison of the Running Time of MIM2 Framework for Different Number of Product m

5.4.5 Advantages of using MIM2

To understand the benefit of proposed MIM2 framework, we perform experiment on influence spread and running time of proposed algorithm for different diffusion graphs under traditional diffusion models.

5.4.5.1 Influence spread

FIGURE 5.10 shows the influence spread of MIM2 algorithm for different m values under IC model. We can see that the influence spread increases with growth in seed size. It is also evident that the influence spread of MIM2 algorithm increases with increment in the number of diffusion graphs k. These experimental results show that our influence model and MIM2 framework are full of effectiveness.

5.4.5.2 Running time

Apart from the influence spread comparison, we also compare the running time of proposed algorithm under MIM2 framework for different m ranges from 1 to 4. FIGURE 5.11 shows the running time of our algorithm for both datasets. We can see that the running time increases with the growth of seed set k as well as the number of product diffusion graphs m. The running time of selecting seed under MIM2 framework takes very few minutes. This shows that MIM2 algorithm has good performance. Therefore, we can say that MIM2 algorithm has good quality solution and efficiency.

5.5 Conclusion

This study is first to introduced a novel framework for multiple influence maximization across multiple social networks. The proposed algorithm

tackles the scenario that multiple products can be promoted in a network with different channel of interactions. Advantages of MIM2 algorithm are noted below.

- The algorithm is simple and effective. Most importantly, it adopts a promotion strategy that each seed user can freely recommend several different kinds of items together and each non-seed user can consider accepting different categories promotions at the same time. Also, it allows users to propagate information across different networks.
- Direct linkage strategy has the advantage to use any existing algorithm on a single network to produce the solution in multiplex networks with the same quality.
- The influence function is submodular and the marginal influence gain can be obtained in linear time under proposed framework.
- Empirical results on real-world networks provides new insights into the MIM2 problem and the benefits of proposed framework over IM, IM2, and MIM framework.