Chapter 4

SIM: Similarity based community model to find influential nodes in multilayer networks

One of the disadvantages of CIM is, its insensitivity to the community structure. This chapter[1] proposes SIM, Similarity-based community influence maximization. SIM overcomes the disadvantages of CIM. SIM is an incremental community based algorithm for influence maximization. Generally, individuals in a community interact frequently and are more likely to influence each other. SIM aims to find k seed nodes in a multilayer network. SIM model has four stages, In the first stage, SIM assigns weighting index (WI) to nodes based on sum of edge weights. In the second stage, we find the initial communities using WI and jaccard similarity. In the third stage, SIM combines some of the small communities which were generated in the second stage. In the fourth stage, SIM selects the k seed nodes from the generated communities based on their degrees. A detailed study of different influence maximization algorithms has been carried out with our SIM algorithm. From our experiments, we show that SIM out performs CIM on real time datasets.

Most of the previous influence maximization studies focus on heuristic approaches, approximation algorithms, and sampling-based algorithms. These models ignore the effect of community structure in influence distribution. In traditional methods, data distribution is over the network's partial area; those in another area of the network never get an opportunity to re-

[1] Under review at Journal of social network analysis and mining, Springer

ceive the information when the seed set k is limited. In practice, we need to avoid the imbalance distribution, i.e., make the influence distribution evenly over the network. Let us discuss the possible problems as an example. There are general elections every five years to elect the union government in India. In order to win with a huge majority, each camp will promote its prime ministerial candidate. However, it is not enough to pursue the maximum total influence because each state has voting rights. Thus, we need to spread the advantage to support the prime ministerial candidate in every state. However, some less populated or less connected states may be ignored in the traditional IM models, which is not advisable.

To the best our knowledge, community-based influence maximization in multilayer networks is not explored. This is the first attempt to perform IM in multilayer networks using a community structure. The significant contributions of this chapter are given as follows. Proposed work has been briefed in Section 4.2.

- We propose weighting index (WI) to nodes based on the edge weights.
- We propose the merging index and cut similarity to merge some small communities.
- We compare the performance of the proposed algorithm with the State of the art algorithms under the IC model.

Table 4.1: Notations used in this chapter

Notation	Description
V	Number of nodes in the multilayer network
K	Number of seed nodes
p	Propagation probability in IC model
G_i	i layer graph in multilayer network
α	attenuation factor
d_j^l	Degree of node j in l^{th} layer
$nbrs(u^l)$	Neighbours of node u in l^{th} layer
θ	Merging index
CS_{ic}	Initial communities
hl	hop length

4.1 Model of Multilayer networks

A multilayer network can be represented as $M = \{G^1, G^2, ..G^{|L|}\}$ [65]. A multilayer network i.e $M = \{G^i; i = 1, 2, ..|L|\}$. In multilayer networks, clones are themselves in other layers i.e.

$V^1 = V^2 = ...V^L = V$. In this, edges are allowed between a node and its counterpart in other layers. A Single layer network represents as $G^i(V, E^i)$, where V set of nodes and E^i is set of edges in layer i. Table 4.1 describes the notations used in this chapter.

Figure 4.1 explains the procedure of our SIM through graph network, network consists of two layers i.e. layer 1 and layer 2. Each layer comprises of 12 nodes. Initially, we find the communities from the two layer networks. In layer 1, it has 3 communities. Community C1 in layer 1 has nodes 1, 2, 3, 4, 5, 6, community C2 in layer 1 has nodes 9, 10, 11 and 12 and community C3 in layer 1 has nodes 7, 8, and 12. Community C1 in layer 2 has node 1, 2, 3, 5, and 6 and community C2 in layer 2 has 4, 6, 7, 8, 9, 10, 11, and 12. Then, we select the seed nodes from the identified communities. Nodes 3 and 12 has selected as seed nodes from the communities in layer 1 and nodes 4 and 5 selected as seed nodes from communities in layer 2. After influence maximization, nodes 3, 5, 9, 10 and 12 are become active in layer 1 and nodes 2, 4, 5, 6, 12 become active in layer 2.

Algorithm 2 SIM: Similarity-based Influence Maximization in multilayer networks

1: Multilayer network $M = \{G^1, G^2, ..G^{|L|}\}$

2: Detect the initial community structures

$$CS_{ic} \leftarrow DIC(M)$$

3: Merge the small communities in CS_{ic}

$$FC \leftarrow MSC(CS_{ic}, \theta)$$

4: Select the seed nodes from the communities based on the size of the community.

4.2 SIM: Similarity based community method to find influential nodes in multilayer networks

SIM is explained in Algorithm 2.

4.2.1 Stage 1: Finding the Weighting Index (WI)

This section presents the first stage of the SIM. SIM propose weighting index and WI is computed based on the edge weights. WI is similar to Katz's centrality. Katz centrality considers the distance. If the distance between node i^l and node j^l is far, then Katz centrality penalizes the edge's weight based on the distance between the nodes. Similarly, WI also penalizes the edge

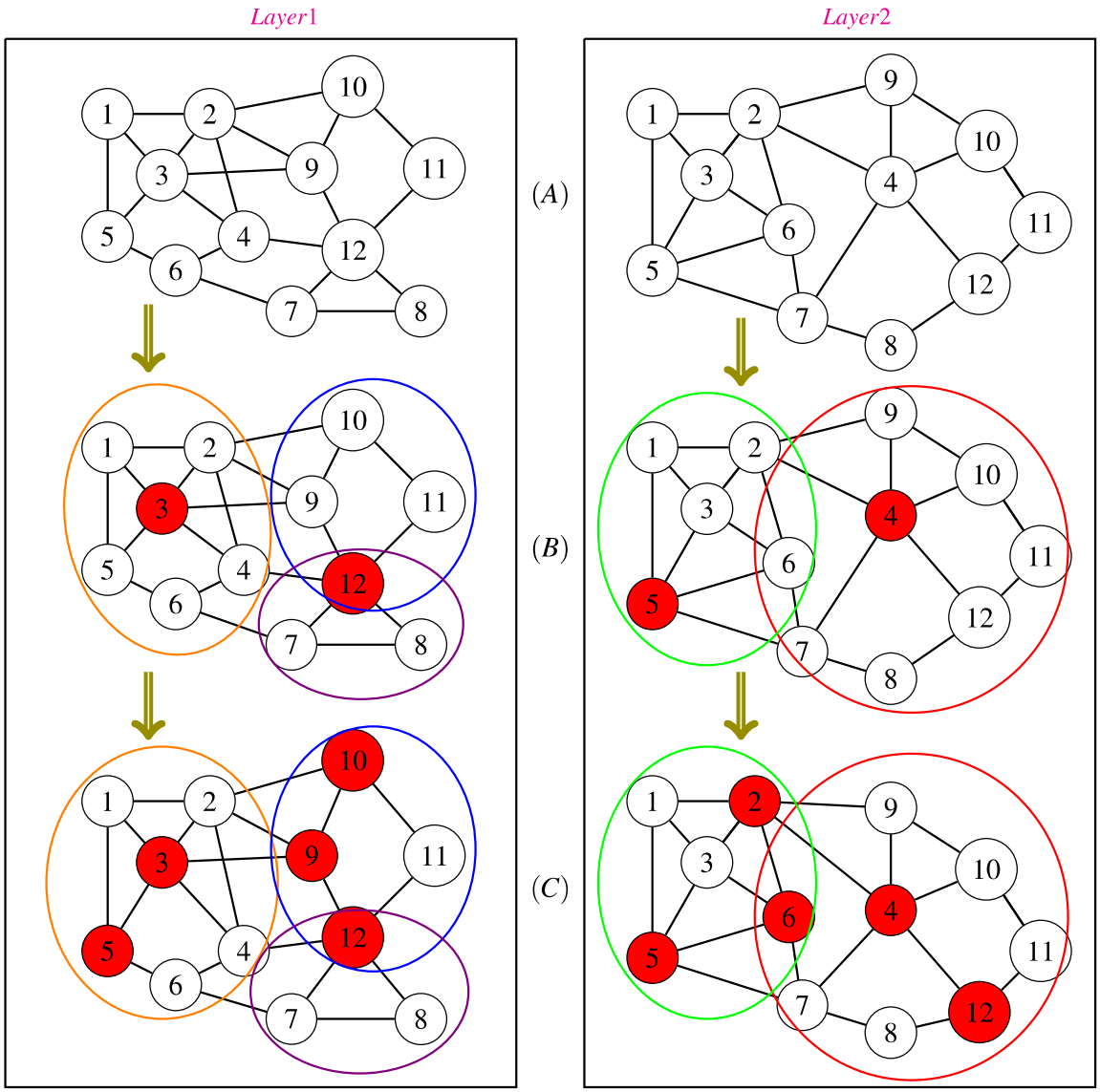


Figure 4.1: (A) Shows the two graphs in layer 1 and layer 2. (B) Shows the different identified communities in layer and layer 2, red nodes are seed nodes. (C) Red nodes are influenced nodes i.e. active nodes and remaining are inactive nodes.

weights based on the distance, but WI rewards the edge's weight based on the degree of the node. WI uses the hop length *hl* to balance the edge weight and reward points. WI is computed as given in Equation 4.1.

$$WI_i^l = \Sigma_{hl=1}^{\infty} \Sigma_{j=1}^{n} [\alpha^{hl} * d_j^l] * (A^{hl})_{ij}] \tag{4.1}$$

WI_i^l is the weighting index for the node i in layer l, A is the adjacency matrix i.e. $n \times n$ matrix, n is the number of nodes in community (C_i^l). $(A^{hl})_{ji}$ shows the total number of paths with length hl from the node i to the node j. α is the attenuation factor, and empirically fix it as 0.1.

4.2.2 Stage 2: Detecting the initial communities

We use WI and Jaccard similarity to detect the initial set of communities. SIM picks the node with highest WI and form it as a new community, then it selects the most similar node to the new community. After that, from the remaining network it chooses the next highest WI node(x) and then finds the node with most similar node (y) to node x. If node y is already available in any of the existing communities, then SIM adds node x to community of y. Otherwise, SIM forms a new community with x and y as its members. This process is continued till all the nodes are placed in some community. Algorithm 3 gives the procedure to detect the initial set of communities.

$$Sim(u^l, v^l) = \frac{(nbrs(u^l) \cap nbrs(v^l)}{nbrs(u^l) \cup nbrs(v^l)} \tag{4.2}$$

4.2.3 Stage 3: Combining small communities

The initial communities generated in stage 2 are small. Selecting seed nodes from these small communities will not give impressive results, as it violates the fundamental characteristic of a community. So it is advantages to merge every small community with their similar communities to form the final community set. Here, we present the process of merging small communities.

Algorithm 3 Detecting initial communities

1: Initialize the two variables i.e. NoS to save the node set and CS_{ic} for initial community set.

$$NoS \leftarrow V, CS_{ic} \leftarrow \Phi$$

2: Caliculate the WI for all the nodes in multilayer network M and record it.
3: Select the highest WI node v^l from multilayer network.
4: Find the most similar node (s^l) of the v^l using Jaccard similarity.
5: **If** the similar node s^l is not available in any of the existing community **then**
6: Form a new community C^l_{N+1} and assign v^l and sn^l to it.

$$N \leftarrow |CS_{ic}|; C^l_{N+1} \leftarrow \{v^l, s^l\}$$

7: Add the new community C^l_{N+1} into community structure CS_{ic}.

$$CS_{ic} = CS_{ic} + C^l_{N+1}$$

8: Remove the two nodes i.e. v^l and s^l from the node set NoS.

$$NoS \leftarrow NoS - \{v^l, s^l\}$$

9: **else**
10: Locate the community where s^l exists to and denote it as C^l_b
11: Add the node v^l into community C^l_b and remove the node v^l from node set NoS

$$C^l_b \leftarrow C^l_b \cup \{v^l\} \; and \; NoS \leftarrow NoS - \{v^l\}$$

12: Repeat the same steps from 3 to 10, until $NoS = nill$
13: return CS_{ic}

SIM identifies the communities that are to be merged. Then it identifies the communities with which the above communities can be merged. We propose the merging Index; the merging Index is measure of average number of nodes for each community.

Definition 3 (Merging Index (Θ))**.** It finds the average number of nodes for a community.

$$\Theta = \frac{|V|}{|C|} \tag{4.3}$$

Where $|V|$ is the total number of nodes in a multilayer network and $|C|$ is the total number of communities generated in the second stage, i.e., detecting initial communities.

All the communities with nodes less than Θ will to be merged and they are listed as small communities (SC). Every community in SC will be merged with a community with which its having the highest similarity (Cut similarity). Algorithm 4 describes the procedure to merge

small communities.

Algorithm 4 Combining Small Communities

1: Initialize Final Communites (FC)

$$FC \leftarrow CS_{ic}$$

2: Find the merging index (Θ) for each community in CS_{ic}.

$$\Theta = \frac{|V|}{|C|}$$

3: Select the communities whose size is lesser than the merging index (Θ) and initialize them as small communites (SC).

4: Sort the small communities SC in ascending order.

5: Select the smallest community C_i^l from the SC and find the most similar community C_j^l to C_i^l.

6: Merge two C_i^l, C_j^l communities and form a new community (C_n^l).

$$C_n^l = C_i^l \cup C_j^l$$

7: Replace two communites C_i^l and C_j^l with new community C_n^l in final communities set (FC)

8: Repeat the merging process from 5 to 8 until $|SC| = 0$

9: return FC

Definition 4 (Cut similarity community)**.** Cut similarity between C_j^l to community C_i^l is computed as given below.

$$CSC_{C_i^l, C_j^l} = \frac{\Sigma_{u \in C_i^l} |nbrs(u) \cap C_j^l|}{|C_i^l| + |C_j^l|}$$

where $|C_i^l|$ is number of nodes in community C_i^l. $|C_j^l|$ is number of nodes in community C_j^l. $nbrs(u)$ is neighbors to node u^l. Community C_i^l need to be merged with community C_j^l.

4.2.4 Stage 4: Finding seed nodes

Seed nodes are selected from the communities by considering degrees of the nodes. The number of seed nodes from a community is computed as given in Equation 4.4. A nodes in a community are sorted in based on degree. Top $CS(C_i^l)$ nodes are selected from community C_i^l.

$$CS(C_i^l) = |S| * \frac{n_i}{|V|} \tag{4.4}$$

Where $|V|$ is the number of nodes in the network, n_i is the number of nodes in community C_i^l. $|S|$ is the total number of nodes in the seed set. The sum total of all seed nodes from various communities forms the final seed node set (SN).

4.3 Evaluation

We used the same datasets as discussed in Chapter 3.

4.3.1 Baseline models

We compare with two models, that are SPINs [56] and KSN [42] introduced in Chapter 3. We also compare with two more models as given below.

- **IMCS [10]**: Influence Maximization on Community Structure (IMCS) is an IM problem. Based on community division analysis, the most significant degree node is selected as the communities' seed node.
- **CIM [36]**: Clique-based influence maximization (CIM) is a heuristics-based algorithm, it consists of two steps. The first step is to find all the available maximal cliques in a multilayer network. Seed nodes will be selected from the maximal cliques based on their size in the second step.

4.4 Results

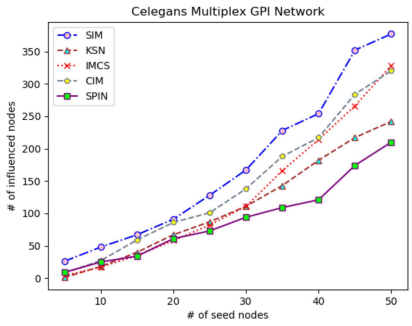
In this section, SIM algorithm is compared with other algorithms in Section 4.3.1 on the real datasets, which are listed in Section 3.2. We analyze and compare the influence spread of different algorithms for finding the seed nodes. We also presented the results for various algorithms on different real-time datasets by fixing the seed node-set size k as 50.

We presented the results in two scenarios. In the first scenario, we discuss the performance

of influence maximization for the proposed and baseline algorithms using the IC model. The second scenario discusses the execution time to identify seed nodes and influence spread on SIM and baseline algorithms using the IC model.

4.4.1 Influence spread evaluation of algorithms on IC model

Here, we present the influence spread of various algorithms on various datasets using the IC model. Figure 4.2a shows IM on Celegans Multiplex GPI Network using the IC model. SIM produced the best results among all the algorithms under different test conditions. CIM and IMCS have influenced the same number of nodes; SPIN has influenced more number of nodes than KSN. Figure 4.3a shows the IM on Arxiv net science multiplex networks. KSN performed better than SPIN; SIM produced the best results among all the algorithms. Figure 4.3b shows the IM on Homo multiplex GPI network. CIM has influenced more number of nodes than KSN and SPIN; SIM has performed better than all other algorithms. Figure 4.4a presents the influence maximization on Moscow athletics 2013, IMCS performed better than CIM, KSN and SPIN. KSN has shown the least performance, and SIM has performed better than other algorithms. Figure 4.4b shows the influence maximization on Newyork climate change 2014. SIM has performed better than all other algorithms and marginally more than IMCS. CIM has influenced more number of nodes than KSN and SPIN.

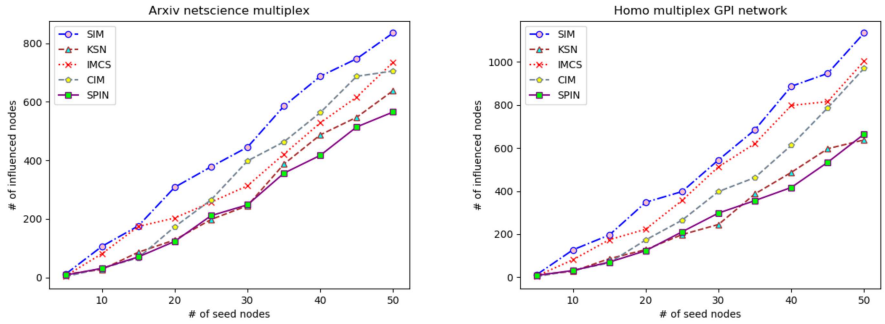


(a)

Figure 4.2: (a) Number of influenced nodes for various algorithms using IC model on Celegans Multiplex GPI network dataset.

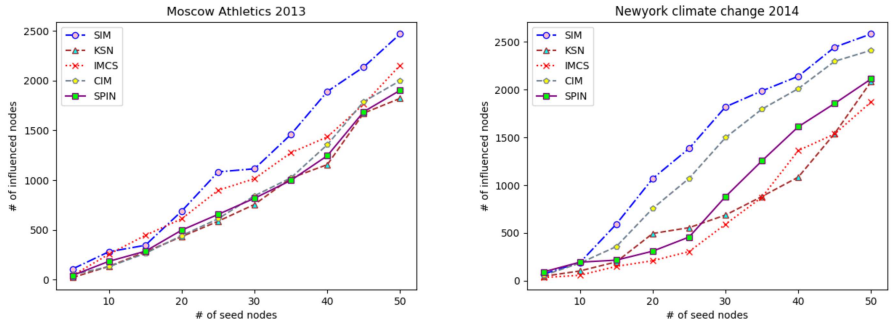
4.4.2 Running time for various algorithms on real time networks

This section presents two scenarios on mentioned datasets, and the first scenario is to find the $k = 50$ seed nodes using different algorithms. The Second scenario is to find the influence



(a) (b)

Figure 4.3: (a) Number of influenced nodes for various algorithms using IC model on Arxiv netscience multiplex dataset. (b) Number of influenced nodes for various algorithms using IC model on Homo multiplex GPI network dataset.

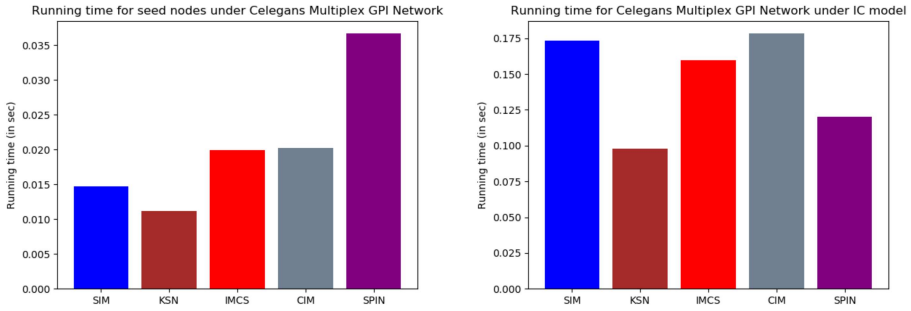


(a) (b)

Figure 4.4: (a) Number of influenced nodes for various algorithms using IC model on Moscow Athletics 2013 dataset. (b) Number of influenced nodes for various algorithms using IC model on Newyork climate change 2014 dataset.

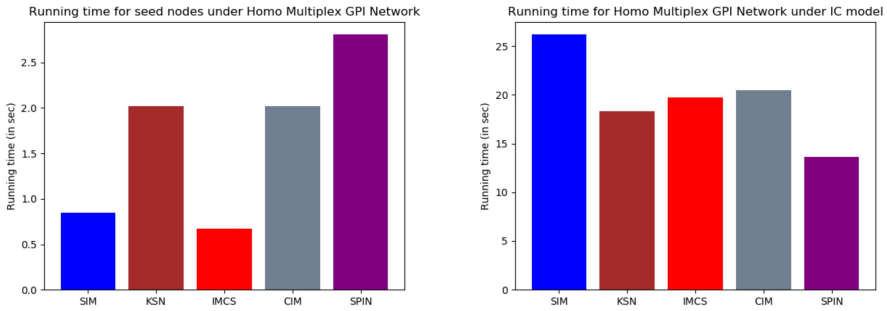
spread on various datasets using the IC model. SIM is more efficient than all other algorithms for finding seed nodes in most cases. On the other hand, SIM sometimes takes more time than other algorithms for influence spreading as it influences more nodes. Figure 4.5a shows the execution time for finding seed nodes on Celegans multiplex GPI network, and SPIN takes more time than all other algorithms. KSN is more efficient than all other algorithms. IMCS and CIM take almost the same time. Figure 4.5b shows the execution time for information spread using the IC model on the Celegans multiplex GPI network dataset. KSN is more efficient than all other algorithms because it influences less number of nodes than all other algorithms. CIM takes more time than other algorithms, and SPIN is more efficient than IMCS, CIM, and SIM.

Figure 4.6a shows the execution time for finding seed nodes on the Homo multiplex GPI network. IMCS and CIM take the same time, and they are more efficient than other algorithms. SPIN takes more time than all other algorithms. CIM and KSN take almost the same time.



(a) (b)

Figure 4.5: (a) Running time of various algorithms for identifying 50 seed nodes on Celegans Multiplex GPI network. (b) Running time of various algorithms for information spread on Celegans Multiplex GPI network using IC model.



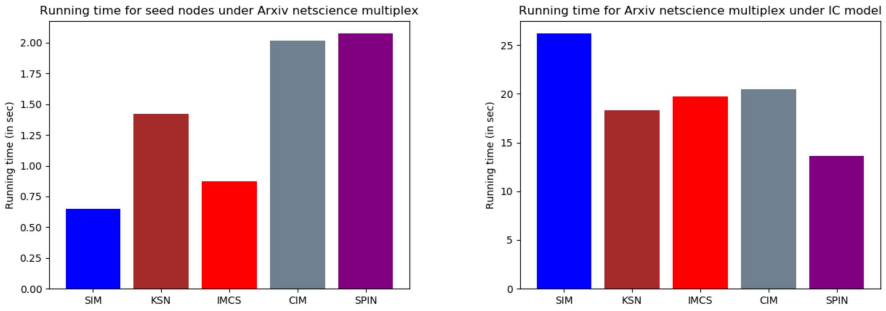
(a) (b)

Figure 4.6: (a) Running time of various algorithms for identifying 50 seed nodes on Homo multiplex GPI network dataset. (b) Running time of various algorithms for information spread on Homo multiplex GPI network dataset using IC model.

Figure 4.6b shows the execution time for information spread using the IC model on the Homo multiplex GPI network. SIM takes more time than other algorithms because it influences more number of nodes than other algorithms. SPIN is more efficient than all other algorithms because it influences less number of nodes than other algorithms. KSN is more efficient than IMCS, CIM, and SIM.

Figure 4.7a shows the execution time for finding seed nodes on the Arxiv nescience multiplex network. SIM is more efficient than all other algorithms. IMCS is more efficient than KSN, CIM, and SPIN. SPIN takes more time than other algorithms. Figure 4.7b shows the execution time for information spread using the IC model on the Arxiv nescience multiplex network. SIM takes more time than all other algorithms, and SPIN is more efficient than all other algorithms. CIM performs better than IMCS, CIM, and SIM.

Figure 4.8a shows the execution time for finding seed nodes on Moscow athletics 2013.

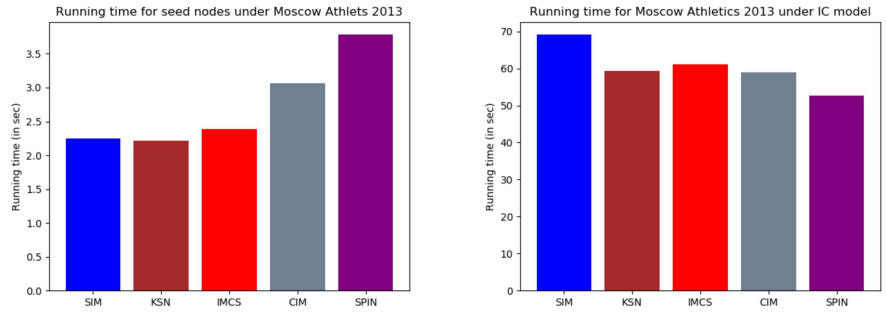


(a) (b)

Figure 4.7: (a) Running time of various algorithms for identifying 50 seed nodes on Arxiv netscience multiplex dataset. (b) Running time of various algorithms for information spread on Arxiv netscience multiplex dataset using IC model.

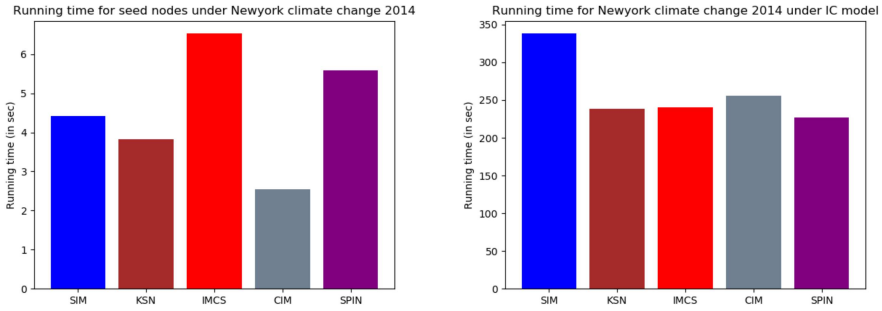
SIM and KSN take almost the same time; IMCS takes marginally more time than SIM and KSN. SPIN takes more time than all other algorithms. CIM is more efficient than SPIN and takes more time than SIM, KSN, and IMCS. Figure 4.8b shows the execution time for information spread using the IC model on Moscow athletics 2013. SIM takes more time than all other algorithms. SPIN is more efficient than other algorithms. IMCS takes more time than SPIN, CIM, and KSN.

Figure 4.9a shows the execution time for finding seed nodes on Newyork climate change 2014. CIM is more efficient than other algorithms, and IMCS takes more time than all algorithms. SIM is more efficient than IMCS and SPIN, and SIM takes more time than CIM and KSN. Figure 4.9b shows the execution time for influence spread using the IC model on Newyork climate change 2014. SIM takes more time than other algorithms, and SPIN is more efficient than other algorithms. KSN and IMCS take almost the same time; CIM takes marginally more time than KSN and IMCS.



(a) (b)

Figure 4.8: (a) Running time of various algorithms for identifying 50 seed nodes on Moscow Athletics 2013 dataset. (b) Running time of various algorithms for information spread on Moscow Athletics 2013 dataset using IC model.

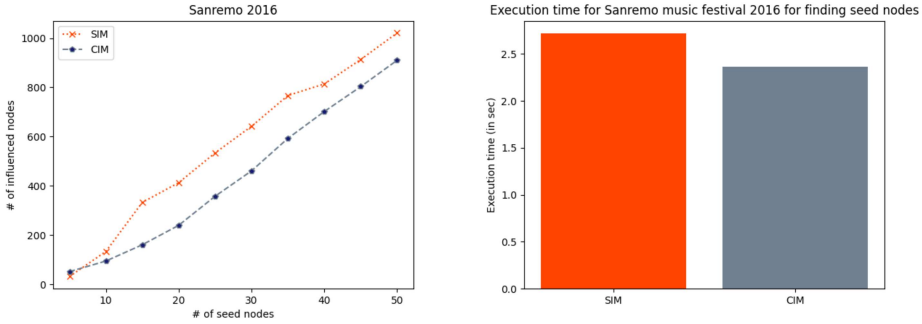


(a) (b)

Figure 4.9: (a) Running time of various algorithms for identifying 50 seed nodes on Newyork climate change 2014 dataset. (b) Running time of various algorithms for information spread on Newyork climate change 2014 dataset using IC model.

4.4.3 Comparision with CIM

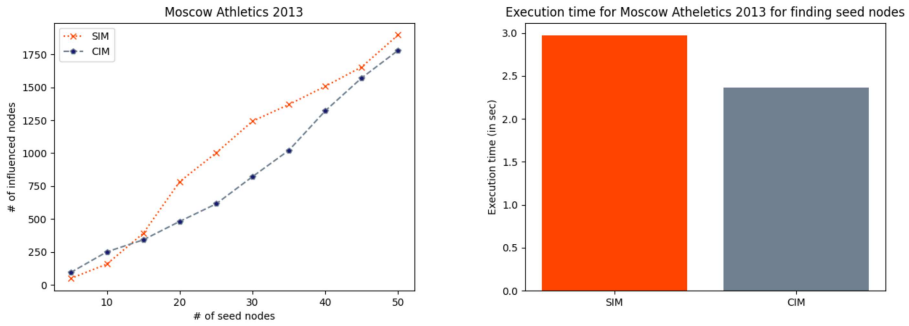
This section compares and analyzes the results of SIM and CIM. Results are in two scenarios, i.e., the influence spread and execution time for finding seed nodes on real-time datasets, which are listed in Section 3.2. Here, we discuss the influence spread and running time to find seed nodes on various datasets. Figure 4.10a shows the influence spread on Sanremo music festival 2016 datasets. SIM outperforms CIM for information spreading. Figure 4.10b shows the running time for finding seed nodes on the Sanremo music festival 2016 dataset. SIM takes marginally more time than CIM.



(a) (b)

Figure 4.10: (a) Number of influenced people for 50 seed users for various algorithms using IC model on Sanremo music festival 2016 dataset. (b) Execution time for various algorithms to find 50 seed users on Sanremo music festival 2016 dataset.

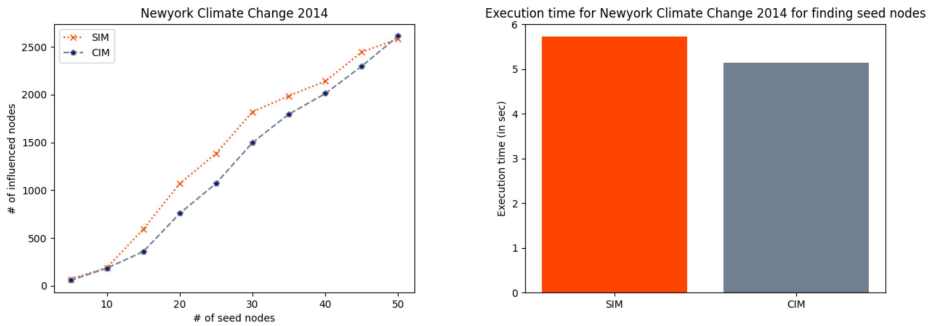
Figure 4.11a shows the influence spread on Moscow Athletics 2013 dataset. The influence spread of SIM is more than CIM. Figure 4.11b shows the running time to find the seed nodes



(a) (b)

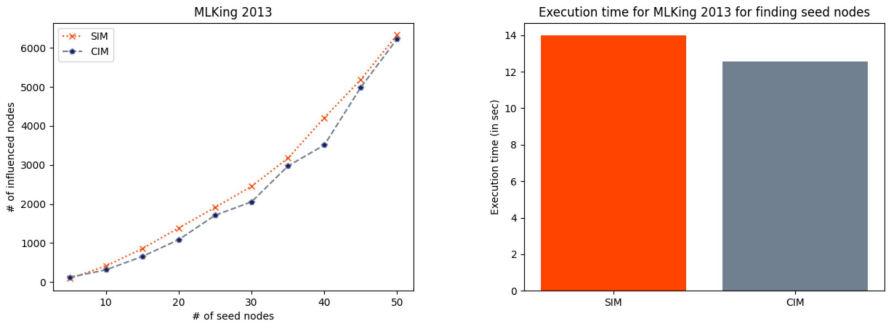
Figure 4.11: (a) Number of influenced people for 50 seed users for various algorithms using IC model on Moscow Athletics 2013 dataset. (b) Execution time for various algorithms to find 50 seed users on Moscow athletics 2013 dataset.

on Moscow Athletics 2013 dataset. Figure 4.12a shows the influence spread on the Newyork climate change 2014 dataset. SIM influence spread is more than CIM. Figure 4.12b shows the running time to find the seed nodes on the Newyork climate change 2014 dataset. Figure 4.13a shows the information spread on Martin Luther King's 50th anniversary 2013 dataset. Figure 4.13b shows the running time to find seed nodes on Martin Luther King's 50th anniversary 2013 dataset. Figure 4.14a shows the influence spread on the Cannes film festival 2013 dataset. Figure 4.14b shows the running time to find the seed nodes on the Cannes film festival 2013 dataset. In both the Cannes film festival and Martin Luther kings speech, the influence spread of SIM is more than CIM.



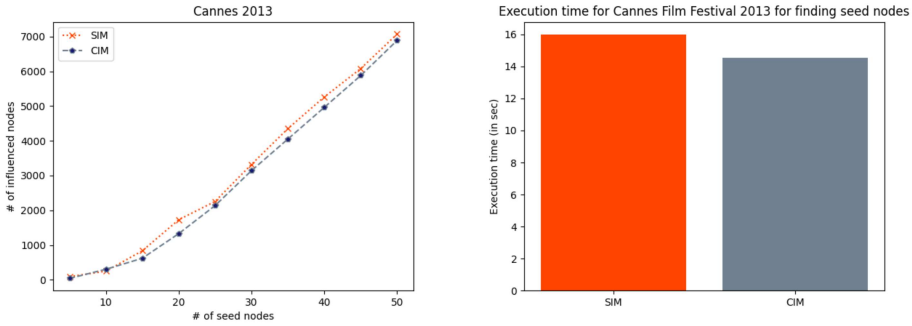
(a) (b)

Figure 4.12: (a) Number of influenced people for 50 seed users for various algorithms using IC model on Newyork climate change 2014 dataset. (b) Execution time for various algorithms to find 50 seed nodes on Newyork climate change 2014 dataset.



(a) (b)

Figure 4.13: (a) Number of influenced people for 50 seed users for various algorithms using IC model on Martin Luther King's 50th anniversary 2013 dataset. (b) Execution time for various algorithms to find 50 seed users on Martin Luther King's 50th anniversary 2013 dataset.



(a) (b)

Figure 4.14: (a) Number of influenced people for 50 seed users for various algorithms using IC model on Cannes film festival 2013 dataset. (b) Execution time for various algorithms to find 50 seed users on Cannes film festival 2013 dataset.

4.5 Summary

In this chapter, we propose the Similarity-based community method influence maximization (SIM) using the IC model in multilayer networks. SIM selects seed nodes that influence more nodes than the competing algorithms. As edge density is more in communities, individuals are likely to have frequent interactions and influence each other. Therefore, SIM has better performance than traditional algorithms.