Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 173 (2020) 315–324

Procedia Computer Science

www.elsevier.com/locate/procedia

International Conference on *Smart Sustainable Intelligent Computing and Applications* under ICITETM2020

Efficient Influence Maximization in Social-Networks Under Independent Cascade Model

Nitesh Trivedi, Anurag Singh

Department of Computer Science & Engineering, National Institute of Technology Delhi, New Delhi-110040, India

Abstract

With in a social network, users interact with each other by sharing news, picture, videos, political content or product promotion content. This makes social networks information diffusion(or marketing) platform. To realize the extent of the adoption of such ideas, it becomes important to study the dynamics of adoption underlying within the social network. However, to analyze social networks as information diffusion/marketing platform, many challenges have to be met. In this paper, we address one of the challenge, namely **Influence Maximization in Social Networks**. Our main purpose is to select k influential users which can gain maximum spread while cost of selecting k users is minimum. In this paper, a) We have proposed an degree heuristic algorithm under Independent Cascade Model (ICM) to extract top k influential users efficiently. b) A modification in ICM is proposed which derives propagation probability based on similarity metrics. The proposed work is implemented and evaluated on two network data-sets, academic collaboration is taken from the online archival database arXiv.org. The obtained results prove that degree heuristic algorithm is very efficient and has influence spread far better than many centrality based heuristics and close to benchmark greedy algorithm. On the other hand, modification in ICM has a significant stable increase in the influence achieved for implemented algorithms compared to influence achieved by the same algorithms on classic ICM.

Keywords: Influence Maximization;Diffusion Models;Social Networks;Independent Cascade Model;Centrality Heuristics

1. Introduction

The average size of the private network of an individual continues to grow. Communication and exchange of data between colleagues and social connections with the popularity of social networking websites such as Twitter and Facebook are much simpler and more common. As a result, in many areas such as economics, sociology, and

* Corresponding author.
E-mail address: anuragsg@nitdelhi.ac.in

10.1016/j.procs.2020.06.037

computer science, the analysis of network and the dynamics over it. Ii helps in information dynamics on a social network and it is a topic of interest in the past few years. Large data sets extracted from social networking websites facilitate research in this domain. Individual's decisions of buying a product, watching a movie, voting particular candidates are affected by their social network to a certain extent. According to the viral marketing theory, if a product is adopted by a certain number of individuals then we can predict further adoptions within the network due to the word-of-mouth effect [6, 3]. It has been observed that different people in the network have different influence power, i.e., some people can influence a larger group of individuals than others, which results in some people acting as the **Influencing agents** in the network. Businesses are now using these agents in the process of Viral Marketing, where a business uses social-network platforms to market their products [16], spreading attractive information about their products to advertise them. One of these strategies consists of finding agents with good online-presence, who could engage potential consumers and advocate brands. To reduce cost and maximize the rate of interest, businesses target minimum set of these agents who can influence the maximum number of consumers in the network. Influence Maximization problem is formally defined as, given a social-network $G = (V, E, W)$ with V as set of vertices, E as set of edges present in graph and W as weight on edges and the diffusion model M on G, goal is to find the k initial active vertices (seed) that can propagate the influence to maximum number of vertices in G. For any subset of vertices, $S \subseteq V$ represents the initial set of active vertices and $\Gamma(A)$ denotes the expected spread with S. We need to find the set S with k vertices such that,

$$S = max(\forall_{S \subseteq V}(\Gamma S)) \tag{1.1}$$

The problem statement can also be generalized by considering the cost associated with the selection of k initial adopters. Rest of the paper is structured as follows, in section 2, some preliminaries are discussed followed by a literature survey in the field of IM. Section 3 contains our proposed work in detail. Section 4 contains results and discussions. It is concluded in section 5 with future scope.

2. Preliminaries and Literature Survey

To understand Influence Maximization from an algorithmic perspective, first, it is needed to discuss some preliminary concepts associated with modeling of an influence maximization problem. **Social Networks** are real-world networks that are modeled corresponding to a graph $G(V, E, W)$, where V denotes the set of vertices (nodes) representing individuals in social network, E denotes the set of edges representing relationship between two individuals, also $E \subseteq V \times V$ and W represents the weights associated with particular edge. Significance and calculation of W depend on the diffusion model. **Seed Node**(say $v \in V$) is a source of information diffusion is considered as a seed node. Seed set is considered as the set of seed nodes, often denoted as S. **Active Node**(say $v \in V$) is considered as an active node under two cases, a) If v is seed node. b) If v receives information due to information propagation dynamics under the diffusion model M from a node belonging to set S. Once v is become activated then it is added to the set of active nodes A_t, where A_t is the set of active nodes at t^th time step. **Spread** **($\Gamma(S)$)** of a seed set S under diffusion model M, is the total number of active nodes at the end of random cascade process that originated from S. Mathematically, $\Gamma(S) = |A_t|$ at any time t. Algorithm 1 represents general spread calculation algorithm [2].

Algorithm 1 General Spread Function

```
Input: $G(V, E, W)$,seed set $S_0$, diffusion model $M$
Output: $A_t$ after some $t$ time steps.
$t \leftarrow 0$
do
$t \leftarrow t + 1$
$A_t \leftarrow$ newly activated nodes at $t^{th}$ time step under $M$
$S_t \leftarrow S_{t\text{-}1} \cup A_t$
Repeat unless $S_t - S_{t\text{-}1} = \emptyset$
$A_v \leftarrow S_t$
return $A_t$
```

There are two major **diffusion models** that are frequently used in the field of influence maximization. The Cascade model and the Threshold model, are two basic models that are used to calculate spread within the social network. In [17], Kempe et al. defined stated two models in their general form along with two special cases known as Linear Threshold Model(LTM) and Independent Cascade Model(ICM). Most diffusion models used currently are an extension of the stated models. Granovetter et al. [12] was first to propose the **threshold model** to study the dynamic process of information propagation within social networks. In LTM, every node within network posses a randomly generated activation threshold θ_v, where θ_v lies in interval $[0, 1]$. In LTM, sum of all incoming edge weights is at most 1 that is $\sum_{\forall u \in In(v)} W(u, v) \leq 1$. A node v gets activated only if the sum of weights present on incoming edges originated from set of active nodes exceeds the activation threshold (θ_v) [12]. Goldenberg et al. [2] investigated the **ICM** to study marketing network. They first start with nodes in the set A_0(Active Set), which is the set of initial active nodes and then the process unfolds in discrete time steps following random behaviour. Each node in set A_0 is considered active at time t. At $t + 1$, assume a node $u \in A_0$ has single chance to activate all its neighbours $v(v \in N(u))$; node u succeeds with a probability p_{uv} (referred as propagation probability), an independent parameter. For many neighbors of u, neighbors get activated in arbitrary sequence. If u successfully activates v, then v will be considered as an active node at time $t + 1$ and will be added in A_{t+1}. The process stops once no further activation occurs. There are different of edge-weight assignment techniques for each available diffusion model. Edge-weights have different interpretations concerning a particular diffusion model. In ICM edge-weight represents the propagation probability while in LTM edge-weight on a directed edge $\vec{uv}$ is the influence node u has on node v. Therefore, we discuss some models which bear different techniques to assign edge-weights under ICM and LTM. LTM edge-weight assignment models include Uniform, Random and Parallel Edges. **Uniform** model assigns edge-weight as $\frac{1}{|In(v)|}$,where $|In(v)|$ denotes in-degree of node v. This model states that each incoming neighbor of v exerts equal influence on v [2]. **Random** model assigns edge-weights randomly between interval $[0, 1]$, since the sum of weights of all incoming edges from active nodes should not be greater than one [18, 19]. **Parallel Edges** model assigns edge-weights to multigraphs. Multigraph is a graph that has parallel edges. Nodes representing individuals in social network graphs often communicate for more than once to other nodes. Under this scenario, weights are assigned based on this particular model. This model is considered as general uniform model for multi-graphs [9]. On the other hand, ICM edge-weight assignment models include Constant, weighted cascade and tri-valency model. In **Constant** model,edge-weights are constant probability value. Each edge has the same probability value. The most stable probability value is.01 whereas other values are possible depending on network [12, 9, 15]. **Weighted Cascade** edge-weight assignment is similar to uniform model in LTM. Again weights are assigned as $\frac{1}{|In(v)|}$,where $|In(v)|$ denotes in-degree of node v. This model states that each incoming neighbor of v exerts equal influence on v hence it is easier to influence nodes with fewer degree [18, 19]. In **Tri-valency Model**, edge-weights are randomly chosen from the set of probability $[0.001, 0.01, 0.1]$ [10, 7]. Since stochastic diffusion models are exploited to calculate the spread given a set of initially active nodes, calculation of the exact spread is not possible. Hence spread is approximated using large Monte-Carlo simulations. The accuracy of the spread achieved is directly proportional to the number of Monte-Carlo simulations taken.

2.1. Evolution in the field of Influence Maximization(IM))

Influence in networks was first explored by Richardson et al. [17] in the framework of a viral advertising strategy to select the finest viral marketing strategy by excavating the networks from data and constructing probabilistic models. The Network was built on data collected from knowledge-sharing websites where customers reviewed products and advised others to buy or not to buy a particular product. They claimed about the optimized cost for each customer, rather than just considering any specific customer's binary occurrence of advertising or not-advertising to that customer. This work is regarded as the initial step towards evolution in the IM field, from then onward the practice of studying relationships between customers using network modeling to estimate influence prevailed. Later on, researchers from many fields applied the concept of influence maximization by formulating it according to their domain. Following Richardson et al.,Kempe et al., [12] were first to prove that the IM problem is NP-hard by relating it to hitting subset problems. They framed the IM problem as an optimization problem considering the influence propagation in social networks under ICM and LTM. They also proved that the spread function $(\Gamma(.))$ for the set of nodes along with its expected value $(E\Gamma(.))$ is sub-modular and monotone. Using monotonic property of spread they proposed a greedy algorithm that can provide solution near to the optimal solution of the IM problem. The Greedy algorithm runs of k (the number of elements in seed nodes to be selected) iterations. The algorithm in its first iteration calculates the spread of each node in the social network under a given diffusion model M and selects the node with maximum influence. For the subsequent iterations, the algorithm calculates the combined influence with nodes in the network (not in seed set) and nodes included in the seed set during previous iterations and selects one node per iteration having maximum marginal influence. Marginal Influence of a node u can be defined mathematically,

$$M_{\text{inf}}(u) = |\Gamma(S \cup u) - \Gamma(S)| \tag{2.1}$$

where, $M_{\text{inf}}(u)$ denotes marginal influence of a node u. Algorithm 2 represents the pseudo-code of the greedy algorithm. As discussed earlier, Influence is a random process, hence it is difficult to calculate exact influence. Therefore

Algorithm 2 The Greedy Algorithm

Input: G as social network graph, k being number of top influential nodes, model of diffusion M
Output: Set of seed nodes S

```
$initialize : S = \emptyset$
for $i \leftarrow 1$ to $k$ do
    for each $u \in V \setminus S$ do
        $Inf_u = 0$
        for $i \leftarrow 1$ to $R$ do
            $Inf_u += |\Gamma(S \cup u) - \Gamma(S)|$ on $M$
        end for
        $Inf_u = Inf_u / R$
    end for
    $S = S \cup \{argmax_{u \in V \setminus S} Inf_u\}$
end for
```

Algorithm 2 uses R as the counter for number of Monte-Carlo simulations which approximate achieved influence. Kempe et al. proved that greedy algorithm approximates spread not less than 63% for the optimal solution. Influence approximation achieved by the greedy algorithm has not been challenged yet hence, this algorithm is considered as the state-of-art work. But the algorithm suffers from one drawback of high time complexity [5]. The greedy algorithm takes a large time to execute on modern systems for medium-sized graphs (graphs with nodes more than 1500). Following the greedy algorithm, many researchers addressed the complexity issues of the greedy algorithm. Out of these one widely appreciated work is of Leskovec et al. [13]. They exploited the sub-modularity property of the greedy algorithm to reduce the complexity of the algorithm. They named it as CELF (Cost Effective Lazy Forward) approach and their results prove that their algorithm runs 700 times faster compared to the general greedy algorithm. After the CELF algorithm, Goyal et al. [8] proposed an update in CELF++ by minimizing the recursive Monte-Carlo simulation being computed in the CELF algorithm. Their method is known as CELF++. They empirically proved that their algorithm is 35-55% faster as compared to the CELF algorithm. Goyal et al. also proposed SIMPATH algorithm [9]. Their model works for LTM and they calculate spread by summing probability values locally in sub-graph of nodes instead of calculating spread over random models. Chen et al. [20] proposed an approach that calculates the maximum influence path between two nodes using the Dijkstra algorithm to estimate the spread. They first created an arborescence tree (a tree in which nodes are directed from the source to sink or vice-versa), later the Dijkstra algorithm calculates the maximum influence path which helps to estimate the influence spread within the network. So far, we have discussed much work that followed the greedy algorithm to address its efficiency issues. The degree is one of the important measures to select influential nodes. Experimental results represent that selecting nodes with order of highest degrees as seeds result in significant influence spread compared to any other heuristics [12], still influence is achieved very less compared to the influence achieved by the greedy algorithm. Chen et al. [4] proposed the degree discount heuristic which is very much efficient as compared to methods available in the greedy domain. Alshahrani et al. [1] proposed an algorithm which first pre-processes the network based on the degree of nodes, they claim that pre-processing generates a more accurate graph. Then they calculate the Katz centrality of nodes in the pre-processed network and select k nodes based on sorted Katz centrality values. They named the algorithm as **PrKatz**. We have re-stimulated their work and we found that their approach lacks in the total influence achieved. We believe that it happens due to the following reasons: a) Their pre-processing step divides the network into many disconnected components which leads to incorrect Katz centrality values of respective nodes. b) They use Katz-centrality to select k influential nodes, while our results prove that degree centrality performs better than Katz centrality. Zhan et al. [22] proposed an algorithm that selects k influential users considering the topological structure of the network and performs seed selection based on certain predefined constraints. Their work lacks to explain the proper derivation and significance of the pre-defined threshold. There have been significant researches addressing the efficiency issues of the greedy algorithm [14, 11], we discussed many of them as our literature review. To address problems of efficiency and accuracy in the field of IM, we propose our degree heuristic algorithm for ICM and to imprint ICM with real-world diffusion process we propose modified ICM.

3. Proposed Method

Proposed work follows two directions in the field of influence maximization: In one direction a degree heuristic approach is discussed, on the other hand, ICM is altered to capture real-world information propagation dynamics.

3.1. Degree Heuristic for Independent Cascade Model(DHICM)

The efficiency issue involved in the greedy approach can be tackled if we turn our way towards centrality heuristics. Degree Centrality, including other available centrality based heuristics, are very common in the field of sociology and literature for detecting influential nodes within the social networks [21]. Degree Centrality has been prevalent than other available centrality heuristics. Results in [12], as well as our results, show that selecting vertices with maximum degree produces higher influence than any other centrality measures. DHICM exploits degree centrality to produce a better result under ICM. The proposed algorithm in general performs far better than the classic degree method and converges towards the influence achieved by the greedy algorithm. On the other side, degree heuristic takes seconds to execute on large networks, which is much faster than the greedy algorithm. The general idea is derived from the single degree discount discussed in [4]. Consider a sub-graph $\mathcal{G} \subseteq G$, having only vertex u and its first neighbours connected to u(see Figure 3.1). Assume $v_i \in N(u)$, where $N(u)$ and d_{vi} represents set of first neighbours of u and degree of vertex v_i respectively. If we assume u as the maximum degree vertex of graph G, then u will be chosen as an influential node if we select nodes based on the maximum degree. Now if we look at Figure 3.1, once vertex u is included in the seed set of influential nodes, each neighbor of u will be subtracted by 1 from its degree. If u is selected as a seed node then mathematically this can be expressed as:

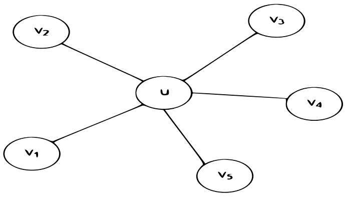


Fig. 3.1: Subgraph $\mathcal{G}$

$$d_{v_i} = \{d_{v_i} - 1, \text{for each } v_i \in N(u)\} \tag{3.1}$$

The degree of v_i is decremented by 1 since any $v_i \in N(u)$ cannot influence u as it has been already chosen into the seed set. This is known as a single degree discount. In our degree heuristic method, we extend this single discount further. We combine the single degree discount with the expected spread that would be caused by node u if particular v_i influences node u. Embedding our method in Eq. 3.2,

$$d_{v_i} = \{d_{v_i} - 1 - (d_u - 1)p, \forall_i v_i \in N(u)\} \tag{3.2}$$

where p denotes propagation probability in ICM. In equation 4.2 we further subtract expected influence of u from the degree of v_i consider the following case: If any v_i selected in the seed set after vertex u. v_i cannot influence u as it is already in seed set as well as the expected influence that can be gained by influencing u is lost for v_i hence we subtract term $(d_u - 1)p$ as the expected influence of node u. Our proposed DHICM algorithm is derived from the degree heuristic algorithm [4]. Chen et al. [4] proposed a degree heuristic that also considers the number of active neighbors of a node, yet they lack to provide a strong reason behind considering active neighbors of the node while the degree of neighbors is subtracted by 1. Algorithm 3 implements the proposed degree heuristic method. Our proposed algorithm runs very efficiently with the time complexity of $O(n + k \log n + m)$, which is much lower as compared to the running time complexity of the greedy algorithm ($O(knRm)$, where R is the total number of Monte-Carlo simulations). It happens because the proposed algorithm works on the degree of nodes to extract influential nodes and calculating degree for each node takes $O(n)$. The efficiency of the discussed algorithm is proved in the results section by comparing it with several other algorithms including the CELF algorithm [13]. We have also compared the influence spread of our algorithm with many others including CELF algorithm [13]. The results prove that the proposed algorithm out performs the classic degree centrality and others while the spread achieved converges towards the influence achieved by the greedy algorithm. The main advantage of the degree heuristic is its efficiency. The algorithm is scalable to large size graphs.

Algorithm 3 DHICM

Input: G as social network graph, k is the number of top influential nodes, model of diffusion M
Output: Set S of seed nodes

```
initialize : $S = \emptyset$
for every $v \in V$ do
    compute the degree of $v$ as $d_v$
    $dd_v = d_v$
end for
for $i \leftarrow 1$ to $k$ do
    $u = argmax_v\{dd_v | v \in V \setminus S\}$
    $S \cup \{u\}$
    for each neighbour $v$ of $u$ and $v \in V \setminus S$ do
        $dd_v = d_v - 1 - (d_u - 1)p$
    end for
end for
Output $S$
```

3.2. Modified ICM

In the ICM it has been reported that even for lower propagation probability (p) as 0.1 [17], there exists a giant component in the network even after the removal of each edge with probability $1 - p$. It makes higher values of p unstable for different influence maximization algorithms. Results in [17] depict that for $p = 0.1$,maximum influence value is reached for very small value of k, for almost every compared algorithm. For $p = 0.1$, increment in marginal influence saturates for small k values. Therefore, a very small probability value(such as $p = 0.01$) is selected to have stable results. But the selection of very small value of p(such as $p = 0.01$) leaves ICM to have very little influence as compared to other available cascade models. Also, with very small p value on each edge of the network does not favor real-world information diffusion dynamics. To address this problem we modify ICM to exhibit similarity property of nodes to define the strength of an edge. Two measures that can locally define the strength of an edge between two nodes are the degree of nodes and common neighbors of nodes. We use both to define the strength of propagation probability. We assume that only network topology is known to us while no information regarding the attributes of nodes is present. We define propagation probability between any two nodes i and j based on Eq. 3.3.

$$p_{\text{ij}} = .01 + \frac{d_i + d_j}{n} + \frac{CN(i, j)}{n} \tag{3.3}$$

where d_i represents the degree of the vertex i, $CN(u, v)$ is the number of common neighbors between nodes u and v and n is the total number of nodes in the network. From right-hand side of equation 3.3, the first term denotes the actual propagation probability, second term added derives the strength of edge (u, v) based on the sum of degree of node u and node v, and it is normalized by dividing it by n (the number of nodes in the network), and the third term, which accounts for the similarity between u and v and hence, it is an important measure to derive the propagation probability on an connecting edge between two nodes. It calculates the similarity by evaluating the number of common neighbors. Deriving strength of each edge based on the above formula gives intuition which relates well with real-world information propagation dynamics. We have implemented various influence maximization algorithms on this proposed modified model. Our results show an increase in influence spread for each compared algorithm and are stable while we increase the value of k. We believe that stability remains because terms added to original propagation probability are normalized, hence accounts for a very little variation in propagation probability of edge. But variation is significant as influence is increased. Spread achieved on MICM is more specific since complete randomness in ICM is handled here with the help of network properties like the degree of node and similarity metric.

4. Results and Discussion

The proposed work is implemented on two large academic collaboration networks [20]. In this network data-set an author is represented as a node and the paper collaborated between two authors is represented by an edge. Both the networks are undirected multi-graphs but for simplicity, we have considered it as an undirected simple graph. We have used a 2.10 GHz Xeon Intel Processor with 6 cores and 12 logical processors with 16GB RAM. Table 4.1 contains information about data sets used to generate the result. Notations n and m represents the number of nodes and edges in network respectively.

Table 4.1: Dataset Used

Network	n	m
NetHept	15,233	58,891
NetPhy	37,154	231,584

4.1. DHICM Results

The proposed algorithms are implemented on ICM. The results of the simulations show that our algorithm DHICM achieves better than the degree centrality and has an influence close to the CELF algorithm for both considered data-sets. **On NetHept data-set** (refer to Figure 4.1), for $k = 10$ and $k = 30$, the proposed algorithm is just 12.5% and 7.9% lower than the CELF algorithm respectively. The proposed algorithm outperforms degree centrality for every value of k, influence for $k = 30$ and $k = 80$ are respectively 16% and 10.4% greater than degree centrality. On the other hand, all other available algorithms perform badly including the PrKatz algorithm proposed in [22]. Katz Centrality based approach achieves spread approximately equivalent to degree centrality for initial k values but spread gradually decreases as k increases. **On NetPhy data-set**(refer to Figure 4.1), proposed algorithm and degree centrality both performs better on NetPhy compared to when implemented on NetHept. It happens due to the underlying network structure. Whereas, the proposed algorithm performs better than the degree centrality even on NetPhy. For $k = 50$, the algorithm achieves 10% better than degree centrality and for $k = 30$, The proposed algorithm achieves influence which is just 2.5% less than the influence achieved by the CELF algorithm. If we focus on the performance of the PrKatz algorithm for both data-sets, it can be deduced that on the NetHept data-set, for some k values-the algorithm performs worst than the Random algorithm while on NetPhy data-set, it throughout performs better than the Random Algorithm. This is because Prkatz performance depends on network structure while Random performs irrespective of the network structure. The execution time ids also compared for all stated algorithms on both models. The execution time is taken in seconds. Proposed algorithm is very much efficient than the CELF algorithm [13]. It just takes a very less time to execute while CELF algorithm [13] takes very large time to execute on both data-sets. It is certainly because of the proposed algorithm works on a degree while the CELF algorithm is an improvement of the greedy

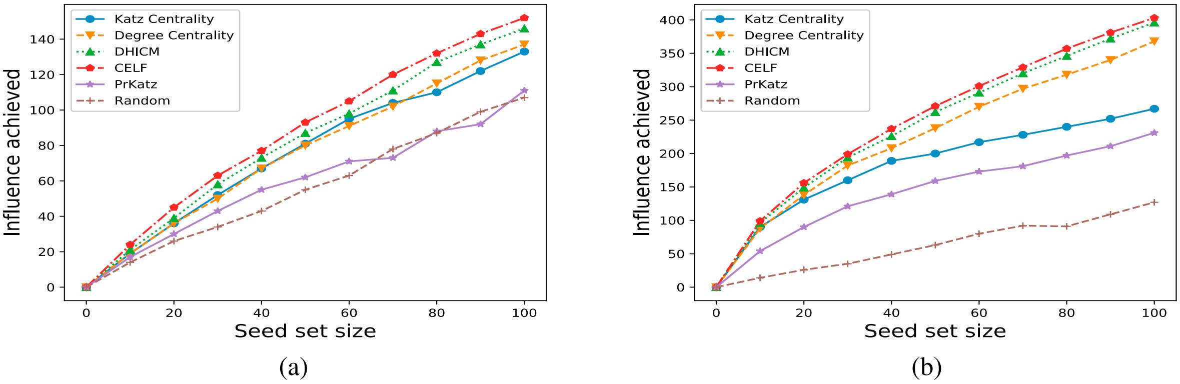


Fig. 4.1: Influence obtained by compared algorithms in **(a)** Comparison of algorithms on NetHept data-set under ICM with $p = .01$ **(b)** Comparison of algorithms on NetPhy data-set under ICM with $p = .01$.

algorithm with the same time complexity as of greedy algorithm. DHICM works far better than the PrKatz algorithm, while it takes a few more seconds to execute compared to degree centrality and Katz centrality.

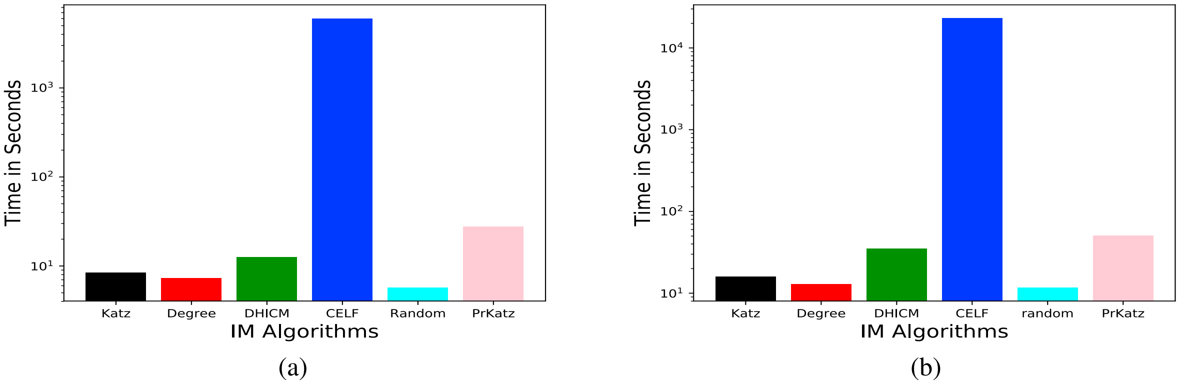
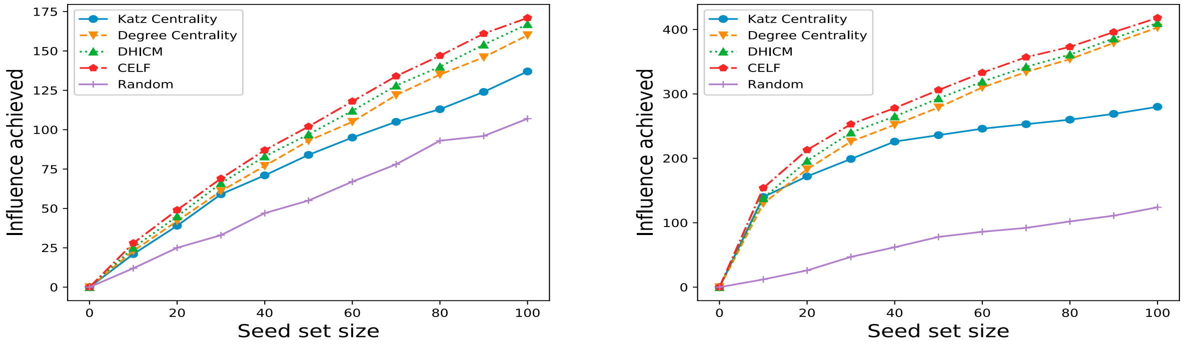


Fig. 4.2: Shows the time(in secs) taken by compared algorithms in **(a)** Time taken by algorithms on NetHept data set under ICM with $p = .01$ **(b)** Time taken by algorithms on NetPhy data set under ICM with $p = .01$.

4.2. Modified ICM Results

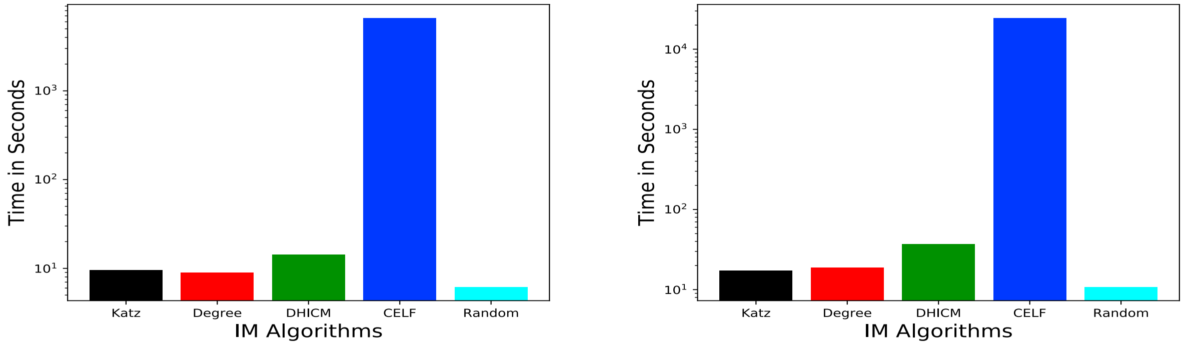
Here, the discussed algorithm is implemented in the proposed modified ICM. We omit the PrKatz algorithm from comparison since it does not perform well. There has been an increment in influence spread for each algorithm on MICM as compared to influence achieved on ICM. On **NetHept data set**(refer to Figure 4.3), Katz Centrality algorithm has 5.7% increase in the influence spread for $k = 30$. Degree Centrality algorithm has an 8.92% increment in the influence spread for $k = 70$. DHICM and CELF algorithms have 6.8% and 12.5% increment for $k = 40$ and $k = 70$ respectively. On **NetPhy data set**(refer to Figure 4.3), Katz Centrality has 31.29% of increment in the influence spread for $k = 20$. Degree Centrality has 32.60% of increment in the influence spread for $k = 20$. DHICM and CELF algorithms have increment of 23.7% and 55.55% increment for $k = 30$ and $k = 10$ respectively. The obtained results reveal that MICM achieves stable influence spread for each algorithm and influence spread is increased for each algorithm. The execution time of the algorithms on MICM is compared. It can be observed in Figure 4.4 that each algorithm takes little more time compared to the time taken on ICM. We believe this happens due to the increase in the influence achieved. For each active node, it takes $O(m)$ time to estimate the influence spread,



(a) (b)

Fig. 4.3: Influence obtained by compared algorithms in **(a)** Comparison of algorithms on NetHept data-set under MICM. **(b)** Comparison of algorithms on NetPhy data-set under MICM.

where *m* represents the number of edges in network. Time for the CELF algorithm increased by 9.4% and 6.5% respectively for NetHept and NetPhy data sets on MICM compared to the time taken by CELF on ICM. Time for our DHICM algorithm has increased by 12.6% and 8.6% for NetHept and NetPhy respectively. While there is no significant increase in time of degree centrality and Katz centrality. We believe that our model takes more time while executing these algorithms is due to the increase in influence spread. The time complexity for each algorithm remains the same under both models. Our comparison results prove that our proposed DHICM algorithm is efficient as well as has significant influence spread, which certainly means that our algorithm addresses both issues associated with influence maximization problem under ICM. Our algorithm can be applied to large networks since it's time complexity is $O(n + k \log n + m)$ which is much faster compared to the CELF algorithm with time complexity $O(nkRm)$, where R is the number of iterations in Monte-Carlo simulation. On the other hand, the proposed modification in ICM achieves increased influence spread for both models and supports the real-world dynamics of information propagation.



(a) (b)

Fig. 4.4: Running time (in secs) for algorithms on both the data sets **(a)** Time (in secs) for algorithms on NetHept data set under MICM. **(b)** Time(in secs) taken by algorithms on NetPhy data set under MICM.

5. Conclusion

We proposed a degree heuristic algorithm under ICM and our results proved that our algorithm is efficient as well as spread close to the greedy algorithm. On the other hand, results also prove that modified ICM has better influence spread than original ICM. Hence, the proposed work has a conclusion with respect to two independent directions. First, we believe instead of making efficient time complexity of the greedy algorithm we need to work in the direction of improving centrality based heuristics, since they are very fast as compared to available methods in the greedy domain. The proposed DHICM algorithm works well for ICM while its performance needs to be tested on other cascade models. Second, we stress on deriving the strength of propagation probability based on available node information or network properties. Though if we have node attributes available we can derive more reasonable propagation probabilities which will reflect actual information propagation dynamics. We can reduce the randomness of diffusion models with external knowledge about node attributes.

References

[1] Mohammad Alshahrani, Zhu Fuxi, Ahmed Sameh, Soufiana Mekouar, and Sheng Huang. Top-k influential users selection based on combined katz centrality and propagation probability. In *2018 IEEE 3rd International Conference on Cloud Computing and Big Data Analysis (ICCCBDA)*, pages 52–56. IEEE, 2018.

[2] Akhil Arora, Sainyam Galhotra, and Sayan Ranu. Debunking the myths of influence maximization: An in-depth benchmarking study. In *Proceedings of the 2017 ACM International Conference on Management of Data*, pages 651–666. ACM, 2017.

[3] Jacqueline Johnson Brown and Peter H Reingen. Social ties and word-of-mouth referral behavior. *Journal of Consumer research*, 14(3):350–362, 1987.

[4] Wei Chen, Yajun Wang, and Siyu Yang. Efficient influence maximization in social networks. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 199–208. ACM, 2009.

[5] Matteo Fischetti and David P Williamson. *Integer Programming and Combinatorial Optimization: 12th International IPCO Conference, Ithaca, NY, USA, June 25-27, 2007, Proceedings*, volume 4513. Springer, 2007.

[6] Jacob Goldenberg, Barak Libai, and Eitan Muller. Talk of the network: A complex systems look at the underlying process of word-of-mouth. *Marketing letters*, 12(3):211–223, 2001.

[7] Jacob Goldenberg, Barak Libai, and Eitan Muller. Using complex systems analysis to advance marketing theory development: Modeling heterogeneity effects on new product growth through stochastic cellular automata. *Academy of Marketing Science Review*, 9(3):1–18, 2001.

[8] Amit Goyal, Wei Lu, and Laks VS Lakshmanan. Celf++: optimizing the greedy algorithm for influence maximization in social networks. In *Proceedings of the 20th international conference companion on World wide web*, pages 47–48. ACM, 2011.

[9] Amit Goyal, Wei Lu, and Laks VS Lakshmanan. Simpath: An efficient algorithm for influence maximization under the linear threshold model. In *2011 IEEE 11th international conference on data mining*, pages 211–220. IEEE, 2011.

[10] Mark Granovetter. Threshold models of collective behavior. *American journal of sociology*, 83(6):1420–1443, 1978.

[11] Meng Han, Mingyuan Yan, Zhipeng Cai, Yingshu Li, Xingquan Cai, and Jiguo Yu. Influence maximization by probing partial communities in dynamic online social networks. *Transactions on Emerging Telecommunications Technologies*, 28(4):e3054, 2017.

[12] David Kempe, Jon Kleinberg, and Éva Tardos. Maximizing the spread of influence through a social network. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 137–146. ACM, 2003.

[13] Jure Leskovec, Andreas Krause, Carlos Guestrin, Christos Faloutsos, Jeanne VanBriesen, and Natalie Glance. Cost-effective outbreak detection in networks. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 420–429. ACM, 2007.

[14] Yanhua Li, Wei Chen, Yajun Wang, and Zhi-Li Zhang. Influence diffusion dynamics and influence maximization in social networks with friend and foe relationships. In *Proceedings of the sixth ACM international conference on Web search and data mining*, pages 657–666, 2013.

[15] Mark Newman. *Networks: an introduction*. Oxford university press, 2010.

[16] Evelien Otte and Ronald Rousseau. Social network analysis: a powerful strategy, also for the information sciences. *Journal of information Science*, 28(6):441–453, 2002.

[17] Matthew Richardson and Pedro Domingos. Mining knowledge-sharing sites for viral marketing. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 61–70. ACM, 2002.

[18] Youze Tang, Yanchen Shi, and Xiaokui Xiao. Influence maximization in near-linear time: A martingale approach. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*, pages 1539–1554. ACM, 2015.

[19] Youze Tang, Xiaokui Xiao, and Yanchen Shi. Influence maximization: Near-optimal time complexity meets practical efficiency. In *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*, pages 75–86. ACM, 2014.

[20] Chi Wang, Wei Chen, and Yajun Wang. Scalable influence maximization for independent cascade model in large-scale social networks. *Data Mining and Knowledge Discovery*, 25(3):545–576, 2012.

[21] Stanley Wasserman and Katherine Faust. *Social network analysis: Methods and applications*, volume 8. Cambridge university press, 1994.

[22] Justin Zhan, Sweta Gurung, and Sai Phani Krishna Parsa. Identification of top-k nodes in large networks using katz centrality. *Journal of Big Data*, 4(1):16, 2017.