Spread of Information in a Social Network Using Influential Nodes

Arpan Chaudhury, Partha Basuchowdhuri*, and Subhashis Majumder

Heritage Institute of Technology,
Department of Computer Science and Engineering,
Chowbaga Road, Anandapur, Kolkata 700107, WB, India
arpanchaudhury@gmail.com,
{parthabasu.chowdhuri,subhashis.majumder}@heritageit.edu

Abstract. Viral marketing works with a social network as its backbone, where social interactions help spreading a message from one person to another. In social networks, a node with a higher degree can reach larger number of nodes in a single hop, and hence can be considered to be more influential than a node with lesser degree. For viral marketing with limited resources, initially the seller can focus on marketing the product to a certain influential group of individuals, here mentioned as *core*. If k persons are targeted for initial marketing, then the objective is to find the initial set of k active nodes, which will facilitate the spread most efficiently. We did a degree based scaling in graphs for making the edge weights suitable for degree based spreading. Then we detect the *core* from the maximum spanning tree (MST) of the graph by finding the top k influential nodes and the paths in MST that joins them. The paths within the *core* depict the key interaction sequences that will trigger the spread within the network. Experimental results show that the set of k influential nodes found by our *core* finding method spreads information faster than the greedy k-center method for the same k value.

Keywords: spread of information, social network analysis, maximum spanning tree, k-center problem.

1 Introduction

1.1 Motivation

A social network is a graph that represents relationships and interactions between a group of individuals. It acts as a medium through which information, innovations and influence spread among its members. An idea forked up from a community or an individual can either disappear with passage of time or influence a significant number of members in the network. For industry-based market analysts, the most interesting feature about a social network is that when people start recommending a new product to their friends, the product gains popularity

* Corresponding author.

P.-N. Tan et al. (Eds.): PAKDD 2012, Part II, LNAI 7302, pp. 121–132, 2012.

very quickly. The strategy of marketing a product by targeting a small number of individuals, which trigger *brand awareness* [1] among all the members of the network through self-replicating viral diffusion of messages, is known as *viral marketing* [2,3,4]. Viral marketing can be much more cost effective than traditional methods since it employs the customers themselves to accomplish most of the promotional effort. Further, as people trust recommendations from their friends more than the manufacturing company itself, viral marketing pays off handsomely. The only challenge we face while utilizing *word-of-mouth* [5,6] advertisement is that we need to pick out a set of customers that maximizes the information flow within a network. For example, suppose we have a social network where the extent to which individuals influence one another is known and we want to endorse a new product in the network. We have a limited budget which is sufficient to convince at most k members to adopt the product. These k members of the network having the information are referred to as the *initial active node set* S and the rest of the nodes, who do not have the information yet, are called *inactive nodes*. The influence spreads from one node to another with time and the *active node set* grows (similarly *inactive node set* decreases) until further spread is not possible.

The problem mentioned above is known as the *influence maximization* problem, which was first introduced by Kempe et al. [7,8] as a discrete optimization problem. In this paper, we put forward an efficient heuristic which improves existing algorithms for influence maximization from two complementary directions. One is to propose a new heuristic that spreads the influence to maximum number of nodes within minimum amount of time and the second is to improve the greedy algorithm to further reduce its run-time. In this section we provide a brief introduction to the problem that we have solved and we also discuss some of the important works related to spread of information that is relevant to our work. In the next section, we have discussed our approach for an efficient spread of information in a network and describe our algorithm elaborately. In the third section, we have discussed about the experimental results describing the performance of our algorithm compared to pre-existing algorithms, we conclude by highlighting our contributions in the section thereafter.

1.2 Literature Review

It is a widely accepted fact that with proper choice of *influential mediators* [9] information can circulate within the network in minimum time. The optimization problem of finding such influential nodes in a social network was first introduced by Domingos and Richardson [2,3]. Motivated by its application in viral marketing, Kempe et al. [7,8] studied the *influence maximization* problem, considering two fundamental propagation models - *linear threshold model* (LT) and *independent cascade model* (IC). They showed that *influence maximization* problem is NP-hard and a simple greedy algorithm of successively selecting influential mediators approximates the optimum solution within a factor of $(1 - \frac{1}{e})$.

Later, Even-Dar and Shapira extended the study of *spread maximization set* problem where the underlying social network behaves like the *voter model*. In

their paper [10], they proposed an algorithm that gives an exact solution to the abovementioned problem, when all nodes have the same cost (cost of introducing a person to a new technology/product), and also provided a *fully polynomial time approximation scheme* for the more general case in which different nodes may have different costs. Kimura and Saito proposed *shortest path based influence cascade models* [11] and provided efficient algorithms to compute spread of influence under these models. Recently, Kimura et al. [12] proposed an alternative method for finding a good approximate solution to the *influence maximization* problem on the basis of *bond percolation* and graph theory. Using large-scale real networks including blog networks they experimentally demonstrated that the method proposed by them is much more efficient than the conventional methods.

Another well-studied problem that we refer to in this paper is the k-center problem [13,14,15,16]. It is defined as a facility location problem where the objective is to find appropriate locations for the facilities such that maximum distance from any client to its nearest facility is minimized. A close observation on k-center problem shows that it is very much similar to the *influence maximization* problem, as in both the cases we try to find a set of nodes which facilitate the service or information spread. In our paper, we show that selecting influentials based on their degrees can produce even better result than existing algorithms and that too in much less time. Knowledge of the related works mentioned in this section gives us an overview of spread of information. However, the algorithm that we have presented in this paper, approaches the problem differently from the existing models.

2 Maximizing Influence Spread

2.1 Problem Definition

Assuming each member of a social graph spreads information to its neighbors with probability 1, we aim at solving the following problem,

Problem 1: *Given a social network graph $G = (V, E)$ with a weight vector W indicating the extent to which individuals influence one another, find a set S of influential mediators of cardinality at most k, such that the objective function is defined as,*

$$r = \max_{v \in V} d(v, S) \tag{1}$$

and is minimized where,

$$d(v, S) = \min_{i=1}^{k} d(v, s_i) \tag{2}$$

and $d(a, b)$ is the shortest distance between nodes a and b.

2.2 Our Approach

In this paper, our primary objective is to find an initial set of *active* nodes in a social graph which maximizes propagation of a new innovation within the

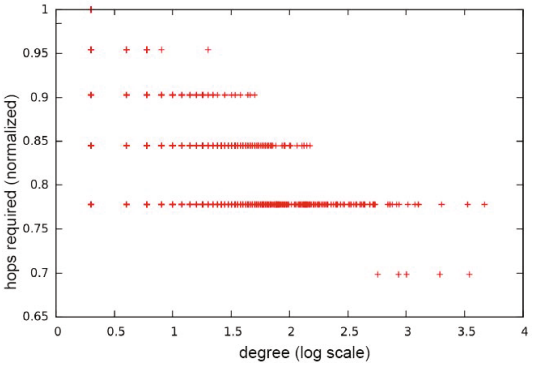


Fig. 1. Correlation between the degree of nodes in a network and the time it takes to spread the information throughout the network (AS relationship data, with 6474 nodes and 13895 edges), if the spread is simulated assuming that initial active node set consists of that node only

network in minimum time. For example, adoption of new drug within the medical profession, use of cell phone among school students, etc. For a social graph $G = (V, E)$, we consider the problem of finding a subgraph $G_c = (V_c, E_c)$, where $V_c \subseteq V$ and $E_c \subseteq E$, through which maximum information flow is likely to happen. We define this region as *core* of the graph. Initially we scale the weight of an edge $e \in E$, by the average of the degrees of the two nodes connected to e. Based on the notion that greater the degree of a node, higher the influence it imparts on the social network due to its ability to reach greater number of nodes (refer to Fig 1). It is desirable to use the edges that are incident on nodes having higher degree. Hence we use this average degree value as a multiplicative factor to the existing edge weights. In case of unweighted graphs, initial edge weights for all edges are taken to be 1 and for weighted graphs, some existing edge weights are assumed to be provided. These initial edge weights have been denoted as $weight_{old}(e_{ij})$ in equation 3. The basic idea is to include high-degree nodes within the *initial active node set*, so that reachability to other nodes within one hop is maximized from the very first step of the spread. Hence, it is also important to track the interactions or the edges between nodes with high influence. To track such edges, we define an objective function to scale the weight of each edge of the graph with the average degree of the nodes connected by that edge.

Definition 1: *Given a social network graph $G = (V, E)$, where $\forall e_{ij} \in E$, e_{ij} denotes an unordered pair of nodes (v_i, v_j), and $v_i, v_j \in V$. We denote the existing weight of e_{ij} by $weight_{old}(e_{ij})$, and then we define the revised weight of an edge to be*

$$weight(e_{ij}) = weight_{old}(e_{ij}) \times \frac{degree(v_i) + degree(v_j)}{2} \tag{3}$$

After scaling the edge weights of the graph, we aim to find the maximum cost spanning tree of the weighted graph. This problem is same as finding a minimum cost spanning tree of an isomorphic graph G_{iso} that has a one to one mapping for all the nodes and edges in G, where the edge weights are of same absolute value but with negative signs. Prim's algorithm for finding minimum cost spanning tree is quite popular and is used on G_{iso}. This minimum cost spanning tree generated from G_{iso} gives us the tree, which can be re-mapped to the labels in G and hence the maximum cost spanning tree of G can be found.

The above representation gives us edge weights based on the influence of the nodes. The function used for defining weights of the edges was motivated by the fact that finding a maximum spanning tree from the weighted graph would give us the path by which a node is connected to its neighbor with highest degree. Hence the maximum spanning tree would generate the path that is most likely to be followed if the influence starts to spread from the nodes with highest degree.

Definition 2: *The maximum spanning tree of a graph G is a connected subgraph $G_T = (V, E_T)$, where $E_T \subseteq E$ and $\forall e_{T_i} \in E_T$,*

$$\sum_{i=1}^{|E_T|} weight(e_{T_i}) \geq \sum_{i=1}^{|E_k|} weight(e_{k_i}) \tag{4}$$

for any E_k, where $E_k \subseteq E$ and $\forall e_{k_i} \in E_k$, forming a spanning tree. The edge weights here essentially denote the strength of interactions between adjacent nodes. So, we essentially scale the existing weight based on the topological structure of the graph and include the significance of the degree of vertices within the edge weights. Attributed graph may have different edge weights for the same edge based on different features. As for example, in a zonal call graph of a cellular service provider, interactions between any two users can be judged by the number of calls or the number of SMSs or some other mode of interactions between them. Strength of such interactions can be judged by the number of calls/week or number of SMSs/week basis. As long as a single composite edge weight based on some objective function can be deduced from the edge weights for each feature, we can also use this method for attributed graphs. However, the ways of finding such composite edge weights, remains out of the scope of this paper. If the edge weights are only determined by some apriori information, the effects of the graph topological structure can be ignored. Hence, in order to take into account both the externally collected information as well as the knowledge of the graph topological structure, we scale the initial edge weights to convert them into new edge weights.

It should be noted that in case of disconnected graphs, if we try to get the maximum spanning tree, not all the nodes will be included in the tree. So, it would only be meaningful to pick the largest connected component of the original graph as the graph, where the spread of information is observed and find the maximum spanning tree from it. Here, G is considered as the largest

connected component of the original graph. Usually for social graphs, largest connected components consist of around 95% (or more) of the nodes in the graph. After selecting the largest connected component and extracting the maximum spanning tree from it, we will have a unique path between any pair of nodes.

The maximum spanning tree, at this stage, consists of a subset of the edge set E using which maximum amount of information flows, but it still consists of all the nodes as in V. For social graphs with fairly large number of actors(nodes), influencing all the nodes immediately requires huge marketing expenses. So, our objective is to select a few nodes with topmost degrees of the network, target to market the product to those influential nodes so that they could spread the influence in as less number of steps as possible. Finding the *core* of the graph provides us with a trade-off between the budget and the time of spread. It does not require the product to be marketed to everyone i.e, the nodes with lower influence can be ignored. Hence, this model can work with a restricted budget. But at the same time instead of influencing everyone in one step, it takes more number of steps to reach all the nodes in the graph. The number of steps to reach all or the majority of the nodes needs to be optimized by suitably choosing the top k influential individuals. We follow some rudimentary graph coarsening techniques to reduce the number of nodes so that we can follow the behavior of the *cores* with various influence limits and become aware of their structures. In order to coarsen the graph, we pick a certain degree threshold based on the point where the degree distribution plot of the nodes in V has the least slope. If the degree threshold is denoted by d_{th} then the final graph that represents a *core* for the threshold d_{th} is denoted by $G_x = (V_x, E_x)$ where $\forall v_{x_i} \in V_x$, $degree(v_{x_i}) \geq d_{th}$. Higher the value of d_{th}, lower will be the cardinality of V_x.

In some cases, where coarsening the graph results in formation of a disconnected *core*, we introduce *bridge* nodes to join the components. Addition of *bridge* nodes in influential node set enhances the chance of knowledge propagation between influentials and hence different communities and thereby increases the spread within the network [17]. Influential nodes may exist in disjoint clusters in different parts of the network. In that case, these *bridge* nodes work as brokers of information from one of those clusters to another. For example, in ancient or medieval age epidemic break-outs stayed within a geographic location as communication between geographic regions was restricted, therefore restricting the *brokers*. But recently, during spread of swine flu, which generated from Mexico, some individuals (here *brokers*) helped its spread to even geographically distant locations like eastern Asia. Intuitively, these *brokers* should have higher edge betweenness values than other nodes in the network.

Note that, in this model we are assuming that a node, who gets activated at time-stamp t, always transmits the information to its neighbors and the inactive neighbors accept the information to become activated at time-stamp $t+1$. If acceptance of information by the neighbors becomes probabilistic, then the model becomes probabilistic too. We plan to follow-up this work with a probabilistic model of influence spread using the *core* as a seed for the spread.

2.3 Detecting the *core*

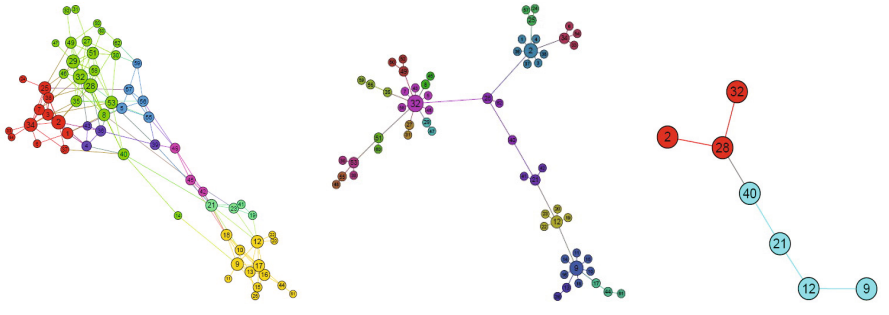
In this section, we explain the algorithm for finding the *core*, as defined in the previous section. Given the graph G and modified weight vector W we find the core G_c using this algorithm. In line 1, we use Prim's algorithm to find the maximum spanning tree and store it as G_T. The vertex and edge set of G_c are initialized in line 3. In lines 4-13, we get the nodes with degree value higher than degree threshold(d_{th}) and connect the maximum spanning tree edges between

Algorithm 1. Detecting the *core* of a graph

Input : $G(V,E)$ the social network, $weight(e_{ij}) \in W, \forall e_{ij} \in E$
Output: The *core* $G_c = (V_c, E_c)$

1 $G_T(V, E_T) \leftarrow MSTPrim(G, weight_{ij})$
2 $//MSTPrim(G, weight_{ij})$ finds the maximum spanning tree of G using Prim's algorithm
3 $V_c \leftarrow \emptyset,\ E_c \leftarrow \emptyset$
4 **for** *each vertex* $v \in V$ **do**
5 **if** $degree(v) \geq d_{th}$ **then**
6 then $V_c \leftarrow V_c \cup \{v\}$
7 **end**
8 **end**
9 **for** *each vertex* $e_{ij} \in E_T$ **do**
10 **if** $u, v \in V_c$ **then**
11 then $E_c \leftarrow E_c \cup \{e_{ij}\}$
12 **end**
13 **end**
14 // In MST, path between any pair of nodes v_i, v_j gives us a tree, defined as $G_{path(i,j)} = (V_{path(i,j)}, E_{path(i,j)})$
15 $G_{cc} \leftarrow DepthFirstSearch(G_c)$
16 // If G_c is disconnected, let $G_{cc} = \{G_{cc1} \cup G_{cc2} \cup G_{cc3} \cup ... \cup G_{ccp}\}$, where $G_{cci} = (V_{cci}, E_{cci})$
17 // Given $i, j,\ where\ i < j, v_i \in V_{cci}, v_j \in V_{ccj},\ G_{cci}, G_{ccj} \in G_{cc}$
18 **repeat**
19 **foreach** *pair* $G_{cci}, G_{ccj} \in G_{cc}$ **do**
20 **if** $\exists k, V_{cck} \subseteq V_{cc}\ and\ \exists v_l \ni v_l \in V_{cck}, V_{path(i,j)}, v_l \notin V_{cci}, V_{ccj}$ **then**
21 $V_{cci} \leftarrow V_{cci} \cup V_{ccj} \cup V_{path(i,j)} \cup V_{cck1} \cup V_{cck2} \cup ... \cup V_{cckr}$
22 $E_{cci} \leftarrow E_{cci} \cup E_{ccj} \cup E_{path(i,j)} \cup E_{cck1} \cup E_{cck2} \cup ... \cup E_{cckr}$
23 $V_{cc} \leftarrow V_{cc} - V_{ccj} - V_{cck1} - V_{cck2} - ... - V_{cckr}$
24 $E_{cc} \leftarrow E_{cc} - E_{ccj} - E_{cck1} - E_{cck2} - ... - E_{cckr}$
25 **else**
26 $V_{cci} \leftarrow V_{cci} \cup V_{ccj} \cup V_{path(i,j)}$
27 $E_{cci} \leftarrow E_{cci} \cup E_{ccj} \cup E_{path(i,j)}$
28 $V_{cc} \leftarrow V_{cc} - V_{ccj}$
29 $E_{cc} \leftarrow E_{cc} - E_{ccj}$
30 **end**
31 **end**
32 **until** G_{cc} *consists of only* G_{cc1};

them. In this way, we get the *influential* nodes but the *broker* nodes are yet to be accounted for. Also, such that the *core* at this stage may be disconnected. So we run depth first search (DFS) on G_c and store the components in G_{cc}. In essence, G_c and G_{cc} are same. If G_c is disconnected, then G_{cc} would be union of multiple disjoint graph components. In lines 18-32, we keep merging the components until one single connected component is produced. In this process, for all pair of components we select any node from each of them and try to find the path between them from E_T. While adding, any node external to V_c might be added. Note that the path between the two components may go through other components. In those cases, all these components are merged into one. The process continues until G_c becomes one connected component. Due to the use of maximum spanning tree for its generation, the final *core* turns out to be a tree (refer to Fig 2 and Fig 3). The run-time of the algorithm is dominated by the step where Prim's algorithm (using binary heap) is being called i.e. $O(|E|log|V|)$.



(a) Dolphin network with 62 vertices and 159 edges

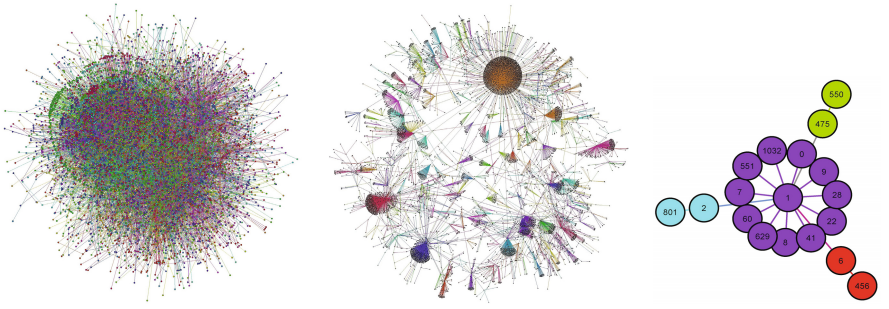
(b) Maximum spanning tree of Dolphin network

(c) *Core* of Dolphin network, with $k = 7$

Fig. 2. Diagrams of the original dolphin interaction network, its maximum spanning tree and *core* with k=7. The coloring of nodes depicts communities within the network and its purpose is to generate a better visualization.

3 Experimental Results

We have tested the quality of *influence maximization set* generated by our method on a number of social networks which have been studied by several authors [18,17,19]. We executed our algorithm for *core* finding and spread of information on a desktop PC with 2.0 GHz Intel core duo processor, 3 GB RAM and LINUX Ubuntu 10.10 OS. We also compared the accuracy of our heuristic with a popular spread maximization method. For graph visualization we used Gephi [20], an open source software for exploring and manipulating networks. All the programs developed for experiment purpose, have been written in C++ and was compiled with GNU g++ 4.6.0 compiler.



(a) AS relationship network with 6474 nodes and 13895 edges

(b) Maximum spanning tree of AS relationship network

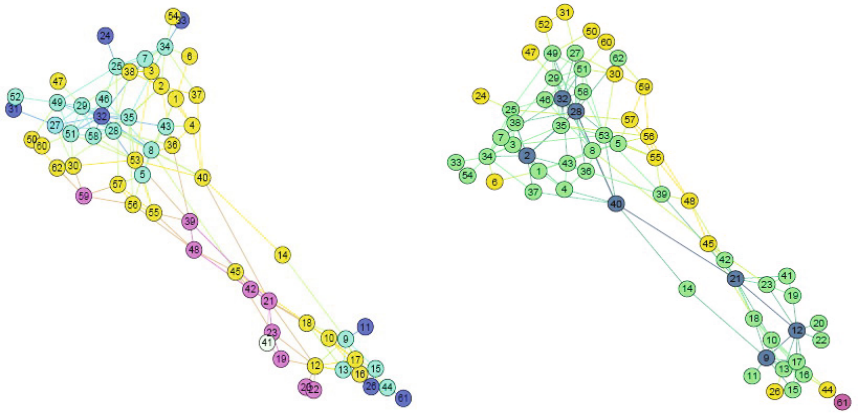
(c) *Core* of AS relationship network, with $k = 18$

Fig. 3. Diagrams of the smaller AS relationship network, its maximum spanning tree and *core* with k=18 (d_{th}=50). The coloring of nodes depicts communities within the network and its purpose is to generate a better visualization.

Table 1. *Core* finding method performs better than greedy k-center overall. Higher the number of k, better the performance of *core* finding method over k-center method.

Data	Number of k	Hops to spread information to 99% of the network nodes	
		greedy k-center	*core* finding
Zachary's Karate Club	3	2	2
Dolphin Network	7	4	3
ArXiv GrQc collaboration	9	7	7
	6	7	7
AS relationship network (small)	18	4	3
	10	4	3
	3	4	4
AS relationship network (large)	30	4	3
	12	4	4
	6	4	4

We have performed the experiments on a total of five different social network datasets of different size. The first one, is Zachary's karate club data, a social network of friendships between 34 members of a karate club at a US university in the 1970s [18]. The second one is an undirected social network of frequent associations between 62 dolphins in a community living off a coastal region of New Zealand. The third dataset, GR-QC (General Relativity and Quantum Cosmology) collaboration network, is from the e-print arXiv and covers co-author relationships between scientists, who submitted their papers to the General Relativity and Quantum Cosmology category between 1993 to 2003. It consists of 5242 nodes and 28980 edges. The other two datasets are, AS-relationship

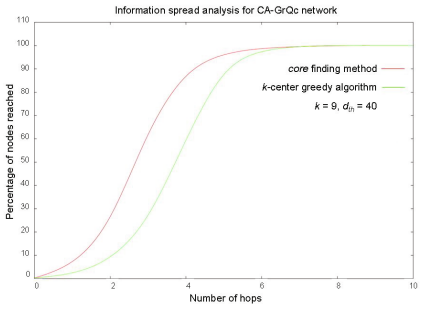


(a) Spread using k-center (b) Spread using *core*

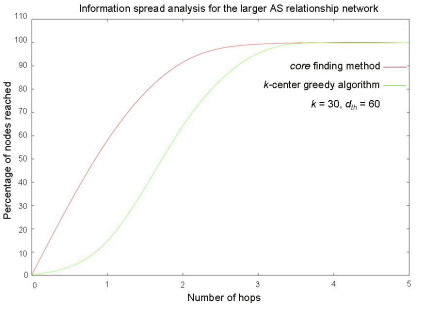
Fig. 4. Visual representation of the spread of information in the dolphin interaction network. Initial active node set is denoted by blue colored nodes. The nodes getting the information at first hop are colored green/cyan, the nodes at second hop are colored yellow, the nodes at third hop are colored pink and the nodes getting the information at the fourth and final step are colored white.

datasets from CAIDA website [19]. AS-relationships are important for routing policies and has implications on network robustness, traffic engineering, macroscopic topology measurement strategies. We use two AS-relationship datasets of different size, to observe how our algorithm performs on network of similar structure but of different size. One of these two datasets has 6474 nodes and 13895 edges and the other has 16301 nodes and 32955 edges.

We have compared our method with k-center problem, which is also a facility location problem and distributes the facilities within the network in such a way so that the maximum distance from all the nodes to its nearest facility is minimized. This is essentially another way to model the spread where the facility locations could be selected for initiating the spread. For all the instances of our experiments, we have seen that the *core* finding method works faster or at least as fast as the greedy solution for the k-center problem. In *core* finding, value of k is determined by d_{th}. From Table 1, it seems that with higher value of k, *core* finding performs better than the greedy solution for the k-center problem. Comparative performance between these two methods for some k and d_{th} combinations using all five datasets have been shown in Table 1, Fig 4 and Fig 5. An important observation from the experimental results is that, even if we increase the value of k, number of steps to reach the information to 99% of the nodes in the network does not necessarily reduce. Say, due to budget constraints, we want to choose k to be 7. Based on input value of k, say, by using the algorithm, we get the number of hops to reach every node in a network from its *core* to be 4. From another observation, we may also get to see that in that same network



(a) Rate of spread in ArXiv GrQc collaboration network



(b) Rate of spread in AS relationship network (large)

Fig. 5. Comparative study between the rates of spread of information using k-center method and *core* finding method, in CA-GrQc and AS-relationship(large) datasets

we can achieve the spread to all nodes with 4 hops for k=5 too. In that case, we need to find the lowest value of k for which the number of hops still remain the same as in case of the input k value. In such a situation, remaining within the budget constraints, no faster spread will be possible but it will be possible that not all the budget will be used up for initial marketing or creating the *initial active set* of nodes. Hence, a lower number of nodes may also be able to spread the information in same time. We want to extend our work by efficiently finding the lowest k values for all set of hops.

4 Conclusion

In this paper, we have presented an efficient method for spread of information by selecting the influential nodes based on degree. We have proposed a technique of scaling existing edge weights based on the degree of the two nodes on which the edge is incident. Using the new scaled edge weights, we have proposed a method to find an important set of nodes from the network and have named it as *core*. We have selected this *core* as the seed or the initial set of *active* nodes for the spread of information and have shown that the spread using the *core* works faster than greedy k-center method.

References

1. Kiss, C., Bichler, M.: Identification of influencers - measuring influence in customer networks. Decision Support Systems 46, 233–253 (2008)
2. Domingos, P., Richardson, M.: Mining the network value of customers. In: KDD, pp. 57–66 (2001)
3. Richardson, M., Domingos, P.: Mining knowledge-sharing sites for viral marketing. In: KDD, pp. 61–70 (2002)

4. Leskovec, J., Adamic, L.A., Huberman, B.A.: The dynamics of viral marketing. TWEB 1(1) (2007)
5. Goldenberg, J., Libai, B., Muller, E.: Talk of the network: A complex systems look at the underlying process of word-of-mouth. Marketing Letters 12, 211–223 (2001)
6. Goldenberg, J., Libai, B., Muller, E.: 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, 118 (2001)
7. Kempe, D., Kleinberg, J.M., Tardos, É.: Maximizing the spread of influence through a social network. In: KDD, pp. 137–146 (2003)
8. Kempe, D., Kleinberg, J.M., Tardos, É.: Influential Nodes in a Diffusion Model for Social Networks. In: Caires, L., Italiano, G.F., Monteiro, L., Palamidessi, C., Yung, M. (eds.) ICALP 2005. LNCS, vol. 3580, pp. 1127–1138. Springer, Heidelberg (2005)
9. Li, C.-T., Lin, S.-D., Shan, M.-K.: Finding influential mediators in social networks. In: WWW (Companion Volume), pp. 75–76 (2011)
10. Even-Dar, E., Shapira, A.: A Note on Maximizing the Spread of Influence in Social Networks. In: Deng, X., Graham, F.C. (eds.) WINE 2007. LNCS, vol. 4858, pp. 281–286. Springer, Heidelberg (2007)
11. Kimura, M., Saito, K.: Tractable Models for Information Diffusion in Social Networks. In: Fürnkranz, J., Scheffer, T., Spiliopoulou, M. (eds.) PKDD 2006. LNCS (LNAI), vol. 4213, pp. 259–271. Springer, Heidelberg (2006)
12. Kimura, M., Saito, K., Nakano, R., Motoda, H.: Extracting influential nodes on a social network for information diffusion. Data Min. Knowl. Discov. 20, 70–97 (2010)
13. Hochbaum, D.S., Shmoys, D.B.: A best possible heuristic for the k-center problem. Mathematics of Operations Research 10, 180–184 (1985)
14. Gonzalez, T.F.: Clustering to minimize the maximum intercluster distance. Theor. Comput. Sci. 38, 293–306 (1985)
15. Mihelic, J., Robic, B.: Solving the k-center problem efficiently with a dominating set algorithm. CIT 13, 225–234 (2005)
16. Berger-Wolf, T.Y., Hart, W.E., Saia, J.: Discrete sensor placement problems in distribution networks. Mathematical and Computer Modelling 42, 1385–1396 (2005)
17. Lusseau, D., Newman, M.E.J.: Identifying the role that individual animals play in their social network. Proc. R. Soc. London B 271, S477 (2004)
18. Zachary, W.W.: An information flow model for conflict and fission in small groups. Journal of Anthropological Research 33, 452–473 (1977)
19. Dimitropoulos, X., Hyun, Y., Krioukov, D., Fomenkov, M., Riley, G., Huffaker, B.: As relationships: Inference and validation. Comput. Commun. Rev. (2007)
20. Bastian, M., Heymann, S., Jacomy, M.: Gephi: An open source software for exploring and manipulating networks. In: International AAAI Conference on Weblogs and Social Media (2009)