Influence Maximization Algorithm Using Markov Clustering

Chungrim Kim, Sangkeun Lee, Sungchan Park, and Sang-goo Lee

School of Computer Science & Engineering
Seoul National University, Seoul 151-742, Korea
{merripu,liza183,baksalchan,sglee}@europa.snu.ac.kr

Abstract. Social Network Services are known as a effective marketing platform in that the customers trust the advertisement provided by their friends and neighbors. Viral Marketing is a marketing technique that uses the pre-constructed social networks to perform maketing with small cost while maximizing the spread. Therefore, which seed user to select is the primary concern in viral marketing. Influence maximization problem is a well known problem to find the top-k seed users who can maximize the spread of information in a social network.

Since obtaining the global optimal solution for the influence maximization problem is proven to be NP-Hard, many greedy as well as heuristic approach has been researched. However, greedy approaches take to much time to obtain the seed node, whereas the heuristic approaches show poor performance. To remedy such problems, we exploit the community structures in the social network to enhance the performance of the heuristic approaches. We perform markov clustering to find the natural communities in the social network and consider the most influential user in the community as the candidate for the top-k seeds. Also, we propose a novel attractor identification algorithm that finds the influential nodes in the community with reduced runtime, and 3 new hybrid approaches for influence maximization problem. Experiments show that the proposed algorithms are more scalable than the greedy approaches, whereas the influence spread obtained by those outperforms the heuristic approaches.

Keywords: Influence Maximization, Markov Clustering.

1 Introduction

Recently, Social Network Services(SNS) such as Facebook[1] or Twitter[2]are arising and its users are constantly increasing. Users of SNS services can interact with other users and form a community disregardin any temporal or geological constraints. SNS services serve as a medium to spread information, influence and ideas throughout the users, and therefore acknowled as a successful advertisement platform. With such tendency, viral marketing using pre-constructed social networks became a prominent figure.

[1] http://www.facebook.com
[2] http://www.twitter.com

Viral marketing is a marketing methodology that uses the word-of mouth effect among the SNS users to perform advertisement about a specific product. For example, Hotmail[3] included an advertisment phrase saying "Get your private, free email at http://www.hotmail.com" in the e-mail of Hotmail users. With such advertisement, Hotmail could naturally spread the advertisement among the preconstructed e-mail network. As the result of the viral marketing, Hotmail has gathered 1,200 users in 2 years[16]. Nowadays, social commerce companies such as groupon[4] use viral marketing to attract customers. Viral marketing aims to achieve maximum advertisement effect within a given budget. For efficient marketing outcome, it is needed to carefully select the seed users who will initiate the marketing process.

To resolve the problem, Influence Maximization Problem aims to find the k people who will maximize the marketing outcome when selected as seed users. Many researches until recently have proposed numerous algorithms for solving the Influence Maximization Problem, either by improving the greedy algorithm or proposing new heuristics. However, there are shortcommings in both approaches that the greedy algorithms' runtime are too large and the heuristics do not take the prominent community structures of the social network. Our contributions in this paper are in threefolds. First, we propose a novel heuristic approach that takes the community structures into account. Second, to further improve the runtime, we propose a novel algorithm to detect the influential node of each community without executing the whole graph clustering algorithm. Last, we propose two hybrid algorithms that combines the attractor detection algorithm with the existing greedy algorithm and the heuristics.

2 Problem Definition

2.1 Social Network Graph

The social network is represented as a weighted directed graph where nodes represent members of the social network and the edges represent relationships or interactions among them.

A weighted directed graph $G = (V,\, E)$ is comprised of tuples between the set of nodes, V, and the set of edges, E. An edge $e \in E$ can be represented as a pair of two nodes $u,\, v \in V$ $e = (u,\, v)$, and the direction from u to v. $e = (u,\, v)$ has $c_{u,\, v}$, the number of interaction between two nodes as weight. The set of neighbors of $u \in V$, $N_G(u)$, is defined as follows.

$$N_G(u) = \{\, v \; inV \mid \exists (u,\, v) \in E \,\} \tag{1}$$

The weight of each edge is normalized by dividing each edge weights by the sum of weights.

$$Pr(v \| u) = \frac{(u,\, v).weight}{\sum_{w \in N_G(u)} (u,\, v).weight} \tag{2}$$

[3] http://www.hotmail.com
[4] http://www.groupon.com/

2.2 Information Spread Model

There are numerous information diffusion models to simulate the spread of information among a network. [1] [4] [10] [9] [13] discuss the "word-of-mouth effect" in the real world. Two of the most basic and widely-studied models will be considered in this paper. Firstly, [10] [9] [15] proposes the Independent Cascade(IC) Model. Each edge in the social network has same probability to influence the target node. The information diffusion under the IC model is simulated as follows.

Definition 1. *Every node can be either active or inactive. An active node represents an influenced user in the social network. The seed set of active nodes is defined as A_0. Newly activated nodes in the ith iteration are defined as A_i. in the $i + 1th$ iteration, a node u in A_i tries to activate its inactive neighbor v with probabiltiy of $p_{u,\,v}$. when u successfully influences v, v is added to A_{i+1} and becomes active. Such iteration is repeated until $A_{i+1} = \emptyset$. The probability of u influencing v, $p_{u,\,v}$, is defined as follows.*

$$p_{u,\,v} = 1 - (1 - p)^{c_{u,\,v}} \tag{3}$$

p in the above formula represents the propagation probability, which is the probabilty of u influencing v with one interaction.

In the IC Model, nodes with high degree have high probability both to influence its neighbor and to be influenced by them. But in some application, nodes with high degree can be less influenced by its neighbor. For example, a person with 100 friends is not easily influenced by one of his friends. However, a person with only one friend can be easily influenced by his only friend. With such intuition, [11] proposed the Weighted Cascade(WC) Model.

Definition 2. *In the WC model, the probability of u influencing v, $p_{u,\,v}$, is defined as follows.*

$$p_{u,\,v} = \frac{c_{u,\,v}}{\sum_{i \in N_G(v)} c_{i,\,v}} \tag{4}$$

2.3 Influence Maximization Problem

Domingos and Richardson[7][18] were the first to define the Influence Maximization Problem as a probablistic algorithm problem. Kempe[11] defined the Influence Maximization Problem as an optimization problem, and proved that such problem is NP-Hard under the IC Model and the WC Model. The Influence Maximization Problem defined by Kempe[11] is as follows.

Definition 3. *Given a graph $G = (V, E)$ and the weights for each $(u,\, v) \in E$ representing the probability of u influencing v, the Influence Maximization Problem finds a set of nodes $S \subseteq V$ that maximizes the influence function $f(S)$ and $\|S\| = k$.*

3 Related Work

3.1 Greedy Approach

The greedy approach proposed by Kempe calculates the expected influence of each nodes using the Monte-Carlo simulation and adds the node with the highest expected value to the seed set, S. The result of the greedy algorithm guarantees the influence spread within (1 - 1 / e) of the optimal solution under IC and WC models[11]. Algorithm 1 describes the greedy alogithm.

Algorithm 1. `GeneralGreedy`(G, k)

```
Require: graph $G = (V, E)$, $k$ for the number of seeds to be selected
Ensure: the set $S$ that maximizes the influence spread
 1: $S = \emptyset$
 2: for i = 1 to $k$ do
 3:     for each vertex $v \in V \backslash S$ do
 4:         $s_v = 0$
 5:         for i = 1 to $R$ do
 6:             $s_v$ += $|f(\mathcal{S} \cup \{v\})|$
 7:         end for
 8:         $s_v = s_v$ / $R$
 9:     end for
10:     $S = S \cup \{\arg\max_{v \in V \backslash S} \{s_v\}\}$
11: end for
12: return $S$
```

However, the greedy approach's runtime is slow because the expected influence for each nodes are calculated using Monte-Carlo simulation. Since the process of influence spread is defined with probabilistic models, the influence of a node can only be measured by simulating the influence spread multiple times and obtaining the approximate value. More runs of Monte-Carlo simulation can improves the approximation, but also takes more time to complete the Monte-Carlo simulations.

To remedy such shortcoming, [14] proposed a method to reduce the runtime by minizing the calculation of influence spread $f(u)$ of a node u. [14] uses a priority queue to recalculate the influnce spread of influential nodes (Cost-Effective Lazy Forward). Only the influence of the node with the highest influence is recalculated. When the recalculated influence spread is still greater than other nodes' influence spread, that node is added to the seed set. However, in the first iteration, influence spread of all nodes still need to be calculated. Therefore it is still inefficient in a large social network graph. [5] also proposed an algorithm that pre-eliminates the edges to calulate the influence spread of all nodes proportional to the node size for one Monte-Carlo Simulation. However, multiple Monte-Carlo Simulation need to be run to obtain more approximate influence spreads. The runtime increases as the number of Monte-Carlo Simulation increases.

3.2 Heuristic Approach

As the greedy algorithms' running time is still large despite the improvements introduced in previous sections, they may not be suitable for large social network graphs. Heuristic Approaches prove to be efficient alternatives for the Influence Maximization Problem.

The most basic approach is the degree centrality heuristic[19]. Intuitively, a user with many friends are influential in a social network. Using such intuition, the degree centrality heuristic selects k nodes that have the highest degree. This heuristic is frequently used in sociology to mine the most influential individual in a social network. [12] expertimentally shows that the degree centrality heuristic outperforms other heuristics in the Influence Maximization problem. But the nodes selected by the degree centrality heuristic only consider its neighbors and therefore cannot be guaranteed to select the optimal seed set. When nodes with high degree are positioned nearly, the influence spread only affects nodes within a certain region.

To remedy such shortcomming, [5] proposes the degree discount heuristic. When a node u is selected as a member of the seed set, the degree of the nodes in $N_G(u)$ are discounted by one. In the next iteration, the node with the highest degree after the discount is selected as a member of the seed set. k iterations are performed to select k nodes with the highest degree. Although degree discount heuristic finds node with larger influence spread than the conventional degree centrality heuristic, it still disregards the community structure of the social network and therefore is apt to select nearby placed nodes. Lastly, [6] uses eigenvector centrality heuristic to select the k influential nodes. When a social network is represented as a transitional matrix, PageRank values for each node are calculated, and the k nodes with the highest PageRank values are selected as seed set.

Heuristic approaches are faster than the greedy approaches, but the seed set tend to be less influential, meaning that the influence spread of the obtained seed set is lower than those of the greedy approaches. Therefore we aim to improve the performance of the heuristic approaches.

4 Influence Maximization Using Markov Clustering

4.1 Markov Clustering

[8] first proposed the Markov Clustering which is a frequently used graph clustering algorithm in Bioinformatics. Markov Clustering divides the graph with a simple intuition. Assume that there exist multiple communities in a social network. When a k-step random walk is performed from a node in the graph, it is usual for the random walker to stop at one of the nodes within the community where u belongs to rather than nodes outside the community. Markov Clustering(MCL) uses such intuition and clusters the nodes whose random walker stops in the same node.

MCL performs two iterative operations repeatedly to find clusters of a graph. Each operations are named as Expansion and Inflation. One successive expansion and inflation operation is called as one iteration. MCL calculates the probability of a random walker stopping at a certain node using the expansion operation. The expansion operation multiplies the transition matrix of a social network graph with itself to calculate the transition probability with twice the random walk step as before. After the expansion operation, MCL uses inflation opration to speed up the conversion. Inflation operation increases the transitional probability of an edge with high weight, whereas decreases the transitional probability of an edge with low weight. Inflation operation modifies the transitional probability by firstly computing the transitional probability to the power of inflation rate. If the newly calculated value is below a certain threshold, that edge is removed and the whole transitional matrix is re-normalized. Expansion and Inflation operations are repeated until the transitional matrix becomes doubly idempotent, in other words, until the transitional matrix of ith iteration and i+1th iteration becomes identical.

The resulting transitional matrix contains the information about attractors and the nodes that are attached to the attractors. The column of the resulting transitional matrix are the starting nodes, whereas the rows are the result nodes. If a node u is attached to the attractor v, the value in the transitional matrix M[u][v] has a value larger than 0. Therefore, a row that contains more than one non-zero values is an attractor, and the number of nodes that are attached to the attractor is the size of the cluster.

4.2 Attractor Detection Using MCL

MCL's main aim is to divide a graph in multiple clusters. However in the Influence Maximization Problem, it is more important to obtain the attractors fastly, as the attractors are the most influential node in the cluster. Therefore we propose a novel algorithm that uses MCL to obtain the attractors fastly. MCL uses matrix multiplication for the expansion operation and therefore has a time complexity of $O(n^3)$ for each iteration. Furthermore, MCL has to perform multiple iterations repeatedly until convergence. When investigating the MCL process, the attractors are moslty identified in the early iterations, but has to complete the remaining iterations until convergence. As explained before, obtaining the attractors is more important in the Influence Maximization Prolem, the remaining iterations can be skipped when most of the attractors are already identified. Algorithm 2 shows the pseudo code for the attractor detection algorithm.

The attractor detection algorithm runs similar to the conventional MCL algorithm. It repeats Expansion and Inflation operations to find the attractors of each clusters. The MCL algorithm can be divided into two phases, namely the growing phase and the shrinking phase. The growing phase of the MCL algorithm is when the number of non-zero values of the transitional matrix increases, and the shrinking phase is when the number of non-zero values decreases. As the expansion operation performs random walks, the transitional probability can become from zero to non-zero when a node becomes reachable after additional

Algorithm 2. `AttractorDetection`(G, r)

```
Require: normalized graph $G = (V, E)$, inflation parameter r
 1: $M = M(G)$
 2: $GrowingPhase = true$
 3: repeat
 4:   $prevNNZ = nnz(M)$
 5:   $M = M^2$
 6:   for $i \in V$ do
 7:     for $j \in V$ do
 8:       $M[i][j] = M[i][j]^r$
 9:     end for
10:     for  do
11:       if $M[i][j] < \theta$ then
12:         $M[i][j] = 0$
13:       end if
14:     end for
15:     for $j \in V$ do
16:       $M[i][j] = \frac{M[i][j]}{\sum_{k \in V} M[i][k]}$
17:     end for
18:   end for
19:   if nnz(M) < prevNNZ then
20:     $GrowingPhase = false$
21:     $AC \leftarrow diag(M)$
22:   end if
23: until $GrowingPhase == true$
24: $AC = \emptyset$
25: for $i = 0 to M.length$ do
26:   if M[i][i] > AC[i] then
27:     $AC \cup i$
28:   end if
29: end for
30: return $AC$
```

random walk steps. On the contrary, the Inflation operation eliminates non-zero values if it does not exceed the given threshold. When the number of the newly created edges exceeds the number of the eliminated edges, MCL algorithm is in the growing phase, and otherwise in the shrinking phase.

In the shrinking phase, the transitional probability towards the atrractors increases due to the fact that the Inflation operation increases the transitional probability of edges that already have high transitional probability. Also, the transitional probability of edges to the non-attractors decreases upon the repetition of the two operations. Therefore, if an self-looping edge's transitional probability increases in the shrinking phase, that node is likely to become an attractor. With such intuition, we proposed a novel algorithm that stores the transitional probability of self-loops in the last iteration of the growing phase, and compares it with the values of the first iteration of the shrinking phase. The nodes whose transitional probability increased are selected as the "Attractor

Candidate". The detected attractor candidates can be used as candidates of the influential nodes in the Influence Maximization Problem.

4.3 Hybrid Algorithms for Influence Maximization Problem

We propose a novel heuristics for the Influence Maximization Problem using the attractor detection algorithm. First, MCL heuristic selects k attractors with the biggest cluster sizes. Assuming that an attractor will influence most of the nodes in the cluster, selecting k attractors with the biggest cluster size can maximize the influence spread. Algorithm 3 shows the pseudo code for the MCL heuristic. The AttractorDetection function refers to the before-mentioned attractor detection algorithm.

Algorithm 3. `MCL Heuristic`(G, k, r)

```
Require: graph $G = (V, E)$, $k$ for the number of seeds to be selected, inflation rate r
Ensure: the set $S$ that maximizes the influence spread
1: $S = \emptyset$
2: $AC = AttractorDetection(G, r)$
3: for $i = 1tok$ do
4:     select $u = \arg\max_v \{clusterSize(v) | v \in AC \backslash S\}$
5:     $S = S \cup u$
6: end for
7: return $S$
```

Secondly, MCL Greedy heuristic applies the greedy algorithm only to the attractor candidates obtained by the attractor detection algorithm. Conventional greedy algorithm need to calculate the influence spread of each nodes and therefore have large runtimes. However, the MCL Greedy heuristic only calculates the influence spread of attractor candidates, and therefore can reduce the runtime. But as the number of attractor candidates increases, the runtime of MCL Greedy heuristic also increases due to the fact that it uses Monte-Carlo simulations to simulate the influence spread for the attractor candidates.

Lastly, the MCL Degree Discount heuristic combines the attractor detection algorithm and the degree discount heuristics which shows the best performance among the heuristics. MCL Degree Discount heuristic considers the community structure of the social network, but does not simulate the influence spread. Therefore it can achieve better performance while running faster than the conventional greedy algorithm.

5 Experiment

5.1 Datasets

Three datasets are used for the experiment. The 'High Energy Physics - Theory Collaboration Network' dataset proposed in [11] [12] [5] are the mostly used dataset in the literature. Also, 'Computational Geometry Collaboration Network'[2] is also used. Both datasets are co-authorship networks. Since real-world dataset of facebook or twitter are large in size and therfore greedy algorithm cannot be run on such datasets. However the co-authorship networks of various sizes are open to public and are known to imply the features of general social networks[17]. Both graphs have authors as nodes and edges when two authors have co-authored a paper. The weight of the edge is the number of papers that the two authors co-written. Each dataset will be referred to HEPT and GEOM in the following sections. Lastly a small-sized real world social network dataset is used to demonstrate the effect of the novel heuristices. This dataset consists of 9 communities and were open to public at NodeXL Graph Gallery[5]. This dataset will be referred as FB. The statistics of each dataset are as follows.

Table 1. Datasets

graph data name	# of nodes	# of edges
HEPT	15,233	58,891
GEOM	7,343	11,898
FB	367	3,728

5.2 Experiment Setup

We compare the proposed heuristics with the new greedy algorithm proposed in [5], degree discount heuristic, random selection, and lastly eigenvector centrality heuristic[3]. This experiment measures the influence spread of each algorithms for the datasets under the IC model and the WC model. The size of the seed set, k, is varied from 1 to 10. The restart probability for the eigenvector centrality heuristic is set to 15%. To measure the influence spread of each node, 1000 Monte-Carlo simulations are executed.

5.3 Effect of Attrator Detection Algorithm

In this experiment, we aim to show the effect of the attractor detection algorithm. Let us define the attractors obtained by fully executing the MCL algorithm as MCL, and the attractors obtained with the attractor detection algorithm as $eMCL$(early-terminated MCL). The recall and precision for the two datasets are calculated using Eq.5.

$$precision = \frac{eMCL \cap MCL}{eMCL}, recall = \frac{eMCL \cap MCL}{MCL} \tag{5}$$

The precision and recall values for the HEPT and the GEOM datasets are as follows.

[5] `http://www.nodexlgraphgallery.org/Pages/Graph.aspx?graphID=584`

Table 2. precision & recall

graph data name	precision	recall
HEPT	0.7692	0.7135
GEOM	0.8495	0.8234

As the result of the experiment shows, attractors with about 80% precision and 76% recall in average are obtained with the attractor detection algorithm. This is due to the fact that the attractor detection algorithm only detects attractor candidates and not the precise clusters. The runtime of each algorithm are shown the table 3.

Table 3. runtime comparison (sec)

graph data name	MCL	eMCL
HEPT	1058.97	235.11
GEOM	118.73	29.76

eMCL in comparison to MCL terminates about 7.45 times faster in HEPT dataset and 4.87 times faster in GEOM dataset. When finding the top 10 nodes that maximizes the influence spread, 8 of the nodes obtained with eMCL were identical to the node obtained with MCL in HEPT dataset. In GEOM dataset, all 10 nodes were identical. The performance comparison between eMCL and MCL will be explained in the next experiment. Shown that the eMCL finds most of the attractors that MCL finds, it is shown that the attractor detection algorithm is efficient that it terminates faster than MCL.

5.4 Influence Spread Comparison

In this experiment, the size of the seed set k is varied under IC model and WC model to demonstrate the effectiveness of each algorithm. Also, runtime of each algorithms are compared for a given value of k which being 10.

Figure 1 shows the influence spread of each algorithm for the three datasets. For all datasets, random selection heuristic show the lowest influence spread and the eigenvector centrality heuristic show second lowest. Degree discount heuristic outperforms other heuristics, but shows slightly lower influence spread than the MCL, eMCL, eMLC Greedy and eMCL Degreee Discount heuristics. The conventional greedy algorithm shows similar influence spread as the proposed heuristics.

For the HEP dataset, the eMCL heuristic, eMCL Greedy heuristic, eMCL Degree Discount heuristic have influence spread that are slightly larger compared to the Degree Discount heuristics. The newly proposed heuristics influence about 95% of the nodes that are influenced by the greedy algorithm. For the GEOM dataset, the hybrid heuristics show 9.3% 20.7% increase in the influence spread.

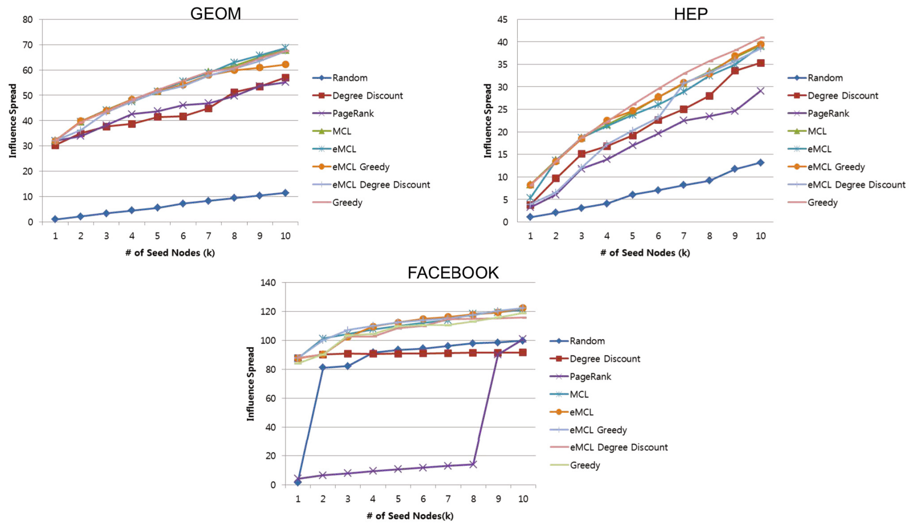

Fig. 1. Influence spread comparison under the IC Model

The eMCL Greedy heuristic influences about 92.1% compared to the greedy algorithm, whereas the eMCL Degree Discount and the eMCL heuristic slightly outperform the greedy algorithm. It can be also seen that eMCL heurisitic influences as much nodes as the MCL heuristic. The result for the facebook dataset proves that regarding the community structure in the Influence Maximization Problem can improve the influence spread in the social network. As the size of the seed set increases, the newly proposed heuristics select influential nodes that do not belong to the same community. When k is set to 10, eMCL Greedy and eMCL Degree Discount show similar influence spread as the greedy algorithm. MCL and eMCL heuristic shows 2.9% and 3% increase in the influence spread. However, the Degree Discount heuristic only shows 77.1% compared to the greedy algorithm.

Figure 2 show the influence spread of each algorithm for the three datasets. The overall result are similar to the experiment conducted on the IC model. For all datasets, random selection heuristic show the lowest influence spread and the eigenvector centrality heuristic follows. Degree discount heuristic show larger influence spread than other heuristics, but smaller compared to the MCL, eMCL, eMLC Greedy and eMCL Degreee Discount heuristics. The conventional greedy algorithm shows similar influence spread as the proposed heuristics. For the HEP dataset, it is shown that the newly proposed heuristics have influence spread that are about 7.9% 13% larger compared to the Degree Discount heuristics. The newly proposed heuristics influence about 94.8% of the nodes with the eMCL heuristic, and 96.9% with eMCL Greedy and eMCL Degree Discount compared to the greedy algorithm. It is shown that for the GEOM dataset the hybrid heuristics show 5%, increase in the influence spread in average. Under the

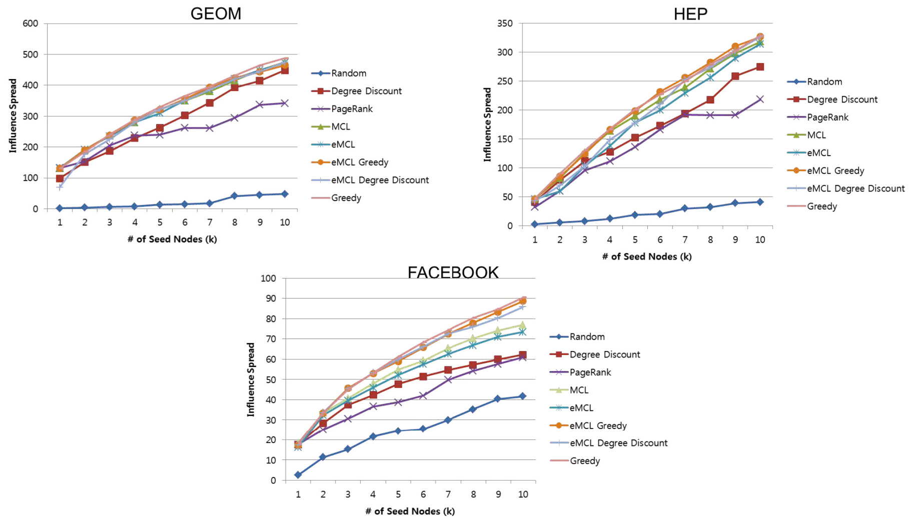

Fig. 2. Influence spread comparison under the WC Model

WC model, the newly proposed heuristics largely outperformed the degree discount heuristic in the GEOM dataset. All three heuristics showed about 96.9% of the influence spread compared to the greedy algorithm. Similar to the experiment under the IC model, regarding community strutures under the WC model improves the influence spread. For the Facebook dataset it is observable that regarding the community structure improves the influence spread. As the size of the seed set increases, the newly proposed heuristics' influence spread outperforms the conventional heuristics. eMCL Greedy shows similar influence spread as the greedy algorithm. eMCL Degree Discount heuristic shows about 94.9% influence spread, whereas the MCL and eMCL heuristics show 85.1% and 81.2% influence spread. The Degree Discount heuristic only shows 68.7% compared to the greedy algorithm.

5.5 Runtime Comparison

The greedy algorithm's runtime under the IC model takes longest to complete and the eMCL Greedy heuristics follows. Greedy algorithms runtime depends on multiple factors such as size of the graph, number of Monte-Carlo simulation, and the size of the seed set. As the size of the seed set, k, increases, the greedy algorithm will take longer to terminate. The eMCL and eMCL Degree Discount heuristics proposed in this paper terminates about 15 times faster than the greedy algorithm whereas their influence spread are similar to the greedy algorithm. The eMCL Greedy heuristic terminates 11 time faster than the greedy algorithm. The greedy algorithm's runtime also takes longest under the WC model. eMCL Greedy heuristics terminates 24.4 times faster than the greedy algorithm. Lastly, eMCL and eMCL Degree Discount heuristics terminates 46 times faster than the greedy algorithm in average. whereas their influence spread are similar to the greedy algorithm.

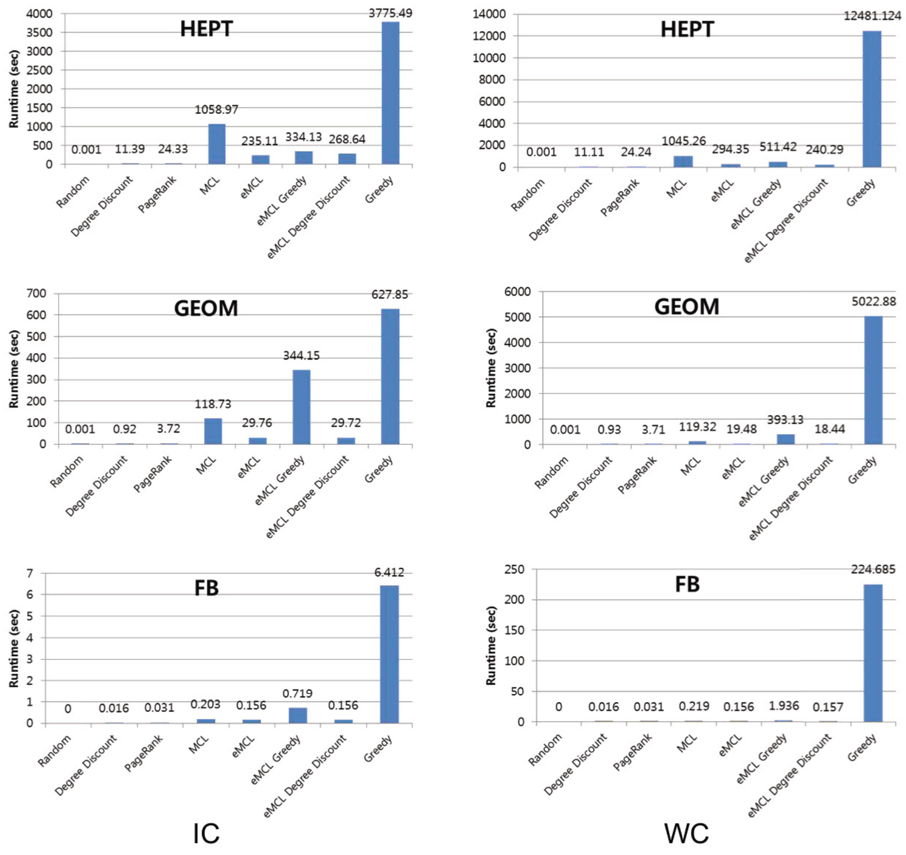

Fig. 3. runtime comparison

The last experiment is to show the scalability of each algorithm. Figure 4 shows the increase in runtime while varying the network size from 3000 to 15000. Simple heuristics such as Degree Discount heuristic or Random selection show almost insignificant increase in the runtime. However, greedy algorithm's runtime tend to drastically increase as the network size increases. On the contrary, the increase in the newly proposed heuristics are up to 4 times less than the greedy algorithm, and therefore is more scalable. As the network size increases, hybrid heuristics can handle larger networks than the greedy algorithm in limited timespan.

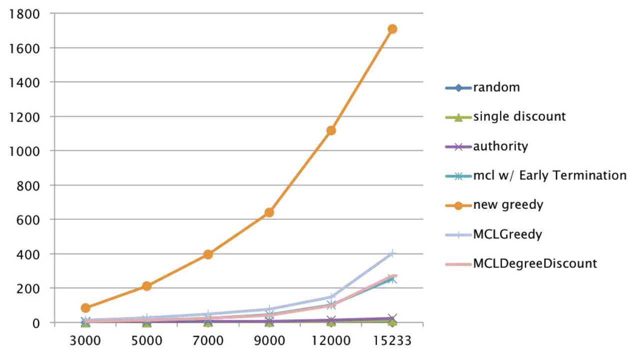

Fig. 4. comparison of runtime varying the network size

6 Conclusion

In this paper, we proposed novel heuristics for the Influence Maximization Problem that regards the inherent community structures in a social network. Also, we proposed an efficient algorithm that only selects the influential nodes in each communities as candidate nodes for the seed set. The efficiency of the attractor detection algorithm is experimentally shown in the experiment section. Using the attractor detection algorithm, we propose three hybrid heuristics for the Influence Maximization Problem. Our heuristics are advantageous in the means that it is more scalable than the conventional greedy algorithm, whereas shows larger influence spread than currently existing heuristics.

There are several future directions for this reserach. First, if MCL can be run in parallel, the scalability of the proposed heuristics will also improve. Extending the MCL to run parallely will be one of the directions. Secondly, extending the attrator detection algorithm to let the user choose its termination point. For example, a user might want more precise attractor candidates while sacrificing the runtime or vice versa. This extention would allow the user to choose what he/she values more, either the performance of the algorithm or the runtime.

Acknowledgments. This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MEST) (No. 20110017480).

References

1. Bass, F.M.: A new product growth for model consumer durables. Management Science 15(5), 215–227 (1969)
2. Batagelj, V., Mrvar, A.: Pajek datasets (2006)
3. Brin, S., Page, L.: The anatomy of a large-scale hypertextual web search engine. In: Proceedings of the Seventh International Conference on World Wide Web 7, WWW7, pp. 107–117. Elsevier Science Publishers B. V., Amsterdam (1998)

4. Brown, J.J., Reingen, P.H.: Social ties and word-of-mouth referral behavior. Journal of Consumer Research 14(3), 35–62 (1987)
5. Chen, W., Wang, Y., Yang, S.: Efficient influence maximization in social networks. In: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2009, pp. 199–208. ACM, New York (2009)
6. Chen, W., Yuan, Y., Zhang, L.: Scalable influence maximization in social networks under the linear threshold model. In: ICDM, pp. 88–97 (2010)
7. Domingos, P., Richardson, M.: Mining the network value of customers. In: Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2001, pp. 57–66. ACM, New York (2001)
8. Dongen, S.: A cluster algorithm for graphs. Tech. rep., CWI (Centre for Mathmatics and Computer Science), Amsterdam, The Netherlands, The Netherlands (2000)
9. Goldenberg, J.: 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, 1–8 (2001)
10. Goldenberg, J., Libai, B., Muller, E.: Talk of the network: A complex systems look at the underlying process of word-of-mouth. Marketing Letters (2001)
11. Kempe, D., Kleinberg, J., Tardos, É.: Maximizing the spread of influence through a social network. In: Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2003, pp. 137–146. ACM, New York (2003)
12. Kempe, D., Kleinberg, J., 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)
13. 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)
14. Leskovec, J., Krause, A., Guestrin, C., Faloutsos, C., VanBriesen, J., Glance, N.: Cost-effective outbreak detection in networks. In: Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2007, pp. 420–429. ACM, New York (2007)
15. Lopez-Pintado, D.: Diffusion in complex social networks. Games and Economic Behavior 62(2), 573–590 (2008)
16. Montgomery, A.L.: Applying quantitative marketing techniques to the internet (2001)
17. Newman, M.E.J.: The structure of scientific collaboration networks. Proceedings of the National Academy of Sciences of the United States of America 98(2), 404–409 (2001)
18. Richardson, M., Domingos, P.: Mining knowledge-sharing sites for viral marketing. In: Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2002, pp. 61–70. ACM, New York (2002)
19. Wasserman, S., Faust, K.: Social Network Analysis: Methods and Applications (Structural Analysis in the Social Sciences), vol. 63. Cambridge University Press (1994)