Annales Mathematicae et Informaticae
44 (2015) pp. 141–150
http://ami.ektf.hu

An improved Community-based Greedy algorithm for solving the influence maximization problem in social networks*

Gábor Rácz, Zoltán Pusztai, Balázs Kósa, Attila Kiss

Eötvös Loránd University
{gabee33,puzsaai,balhal,kiss}@inf.elte.hu

Submitted September 15, 2014 — Accepted March 30, 2015

Abstract

The influence maximization problem is to find a subset of vertexes that maximize the spread of information in a network. The Community-based Greedy algorithm (CGA) is one of the many that approximates the optimal solution of this problem. This algorithm divides the social network into communities, and then it takes into account for each node only its influence inside the cluster to which it belongs. Our method improves this algorithms with two modifications. We replace the clustering method of the CGA with a commonly used algorithm, namely the Louvain method, which runs by even one magnitude faster. We performed measurements to test how this replacement affects the running time and the precision of the algorithm. The results show that our variant significantly reduces the running time and the precision loss is less than five percent.

Keywords: influence spread, social network, community detection

MSC: AMS classification number(s): 91D30, 91C20, 51E23

*This work was partially supported by the European Union and the European Social Fund through project FuturICT.hu (grant no.: TAMOP-4.2.2.C-11/1/KONV-2012-0013). This work was completed with the support of the Hungarian and Vietnamese TET (grant agreement no. TET 10-1-2011-0645).

1. Introduction

Over the last few years a large variety of on-line social networks has become available. There are general purpose social networks such as Facebook[1] or VK[2] which provide medium to their users for sharing thoughts or talk about their everyday life. Other social networks have special interests such as the business-orientated LinkedIn[3] or the music-oriented Last.fm[4]. In addition to the above mentioned ones, social networks can be constructed based on email communications, phone call records, or co-authorship of scientific papers. The diversity and the volume of these networks have posed serious challenges to the scientists, however, they also offer great opportunity to understand human relationships. One interesting question among many others is to find a fixed number of vertexes through which the largest possible part of a network can be reached. This problem is mainly referred as influence maximization problem. It has a lot of practical usages, for example, in case of viral marketing the question is who should be targeted with sample products or who should be conceivably paid in a marketing campaign in order to influence as many members of the network as it is possible. In addition, if the most influential members of the network are found, it can be investigated why they are the most influential members [10].

In [1], Kempe et al. introduced two basic models, namely the *Independent Cascade Model* and the *Linear Threshold Model* for representing the diffusion of influence in networks. The influence maximization was considered as a discrete optimization problem. It was proven that the problem is NP-hard in both cases; nevertheless, it was also shown that based on submodularity of the scoring function the simple greedy algorithm assuredly approaches the optimal solution by a factor of $1 - \frac{1}{e}$. However, a serious drawback of this algorithm is that the influence of the candidate sets should be evaluated in each turn, which owing to the non-deterministic nature of the process is accomplished by using Monte Carlo simulations. As for large graphs these simulations can be very time consuming, several improvements were introduced since the greedy algorithm was published. In this paper, we focus on the *Independent Cascade Model* only.

In [5], a Cost-Effective Lazy Forward (CELF) optimization was presented that can significantly reduce the number of evaluations by exploiting the submodularity of the scoring function.CELF results a candidate set that has the same influence spread as the original greedy algorithm but is much faster (even 700 times faster [5]). Chen et al. in [2] proposed the NewGreedy algorithm that is an improvement of the original method in which at the beginning of an iteration each edge of the input graph is deleted with a certain probability. In this way the original problem can be converted into a reachability problem where the influence spread of a node set S is measured as the number of reachable nodes from S. It constructs a candidate set that has the same influence as the original

[1] `https://facebook.com`
[2] `https://vk.com`
[3] `https://www.linkedin.com`
[4] `http://www.last.fm`

greedy algorithm but it has shorter running time. In [3], Wang et al. introduced the COMMUNITY-BASED GREEDY algorithm, referred as CGA, which consists of two phases, a clustering and a dynamic programming phase. Their main idea is to divide the network into communities. The influence degree of a node in the community approximates its influence degree in the whole network. In addition, a dynamic programming method is used to select which cluster should contain the next member of the candidate set in each turn.

In this paper we present a solution for the influence maximization problem which relies on the CGA. In our solution, the clustering method of the CGA is replaced by a community detection algorithm, called LOUVAIN METHOD [4], which is a simple method and it can be computed extremely fast even in the case of large networks. However, in contrast to the original one, this method does not provide theoretical bound to the precision loss that the approximation can cause. Moreover, the dynamic programming phase is also simplified in our solution. Namely, in each turn only those nodes are re-evaluated which belong to the community that contains the previously selected member of the candidate set. We evaluated how these changes affect the running time and the precision of the algorithm in comparison with the CGA and to the NEWGREEDY algorithms. Our results show that the modified algorithm can run ten times faster than NEWGREEDY three times faster than CGA and its precision loss is less than five percent.

2. Background

A social network is modeled as an undirected graph $G = (V, E)$, where nodes represent individual persons while an edge between two nodes models some sort of relationships. The influence maximization problem is to find an S subset of V with cardinality k, where k is a fixed constant, that maximize the σ *influence function* which assigns a non-negative real value to each subset of V. Two basic diffusion models were introduced in [1] by means of which the influence function can be calculated. In both models, each node has an active or an inactive state, where the active nodes represent influenced persons who themselves can also influence others.

In the *Linear Threashold Model*, a node v has a random threshold θ_v, and v is influenced by its neighbour w according to a weight b_{vw} such that $\sum_{w \text{ neighbours of } v} b_{vw} \leq 1$. The diffusion process starts from an arbitrary set of nodes S, called seeds and the process unfolds in discrete steps: in step t, all the active nodes remain active, and any v node becames active for which the total weight of its active neighbors is at least θ_v, formally $\sum_{w \; active, w \text{ neighbours of } v} b_{vw} \geq \theta_v$.

In the *Independent Cascade Model*, the diffusion process also starts from an arbitrary set of nodes S and it unfolds in discrete steps: in the $(i+1)^{th}$ step, each node that has become active in the i^{th} step has a single attempt to influence its currently non-active neighbours. More precisely, for such a node the connected edges are taken one after the other with a fixed activation probability p. If an edge

was chosen, then the other endpoint is also get activated. The process stops if no new node has become active in a round or every node has been activated. The influence of S will be the number of activated nodes. In the rest of this paper, we focus on only the latter diffusion model.

In [1], it was shown that the influence function is submodular and monotone in the *Independent Cascade Model*. In other words, for each $S \subseteq V$ and a node v: $\sigma(S) \leq \sigma(S \cup \{v\})$. Moreover, the marginal gain of adding the same node to a growing set decreases as the set becomes larger, i.e. for each $S \subseteq H \subseteq V$ and a node v: $\sigma(S \cup \{v\}) - \sigma(S) \geq \sigma(H \cup \{v\}) - \sigma(S)$. With these properties, it can be guaranteed that the result of the greedy algorithm is less than $(1 - \frac{1}{e})$ times of the optimal solution. Formally, $\sigma(S_{greedy}) \geq (1 - \frac{1}{e})\sigma(S_{opt})$, where S_{greedy} denotes the result of the greedy algorithm, while S_{opt} the optimal solution respectively. Owing to the non-deterministic nature of the diffusion model in practice the values of σ are approximated by means of Monte Carlo simulations. For a given node v, usually 10.000 simulations are performed to approximate $\sigma(S \cup \{b_v\})$, where S denotes the set of nodes selected in the previous steps of the algorithm, therefore the algorithm is time consuming in case of large networks.

An improvement was introduced in [2], in which at the beginning of an iteration each edge of the original graph is deleted with probability $1-p$. Then, the influence of a set of nodes S can be measured by the number of reachable nodes from S. In addition, the computation of the marginal gain of a node v with respect to an $S \subseteq V$ can be seen as a reachability problem which is defined in the following way:

$$\sigma(S \cup \{v\}) - \sigma(S) = \begin{cases} 0, & \text{if } v \in R(S), \\ |R(\{v\})| & \text{otherwise,} \end{cases}$$

where $R(S)$ denotes the set of the reachable nodes from S.

In this paper, we focus on the Community-based Greedy algorithm that was introduced in [3]. Its approach is orthogonal with the improvement applied in NewGreedy, it is based on graph partition. The algorithm consists of two phases, a clustering and dynamic programming phase. In the first phase, a community detection algorithm is performed on the input graph, this algorithm has two subphases, namely a label propagation and a combination step. Initially, each node has a unique community label. Next, for each node the set of its influenced neighbours are computed using the *Independent Cascade Model*. Then the community labels are propagated iteratively in τ rounds (where τ is given in advance) through the network. The main principle of the propagation is that a node v should belong to the community that contains the majority of its influenced neighbors. Formally, $v.c^t = maxCMT(w_1.c^{t-1}, ..., w_k.c^{t1-1})$, where t denotes the tth round, $w_1, ..., w_k$ are the neighbours of v, $v.c$ denotes the community label of v, and $maxCMT$ is to compute the majority of the labels.

In the combination phase, the algorithm combines community C_l and C_m, if the combination entropy of C_l to C_m is above a given threshold. This phase helps to reduce the difference between the node's influence degree in its community and its influence degree in the whole network. The *Combination entropy* was introduced

to measure the connection between two communities and it is defined as:

$$CoEntropy(C_l, C_m) = max_{v \in C_m, u \in C_l, isLive(e_{uv})} \frac{\bar{R}_m(\{u\})}{R_m(\{v\})},$$

where $R_m(\{v\})$ is the influence degree of v in C_m, $\bar{R}_m(\{u\})$ is the influence degree of u outside C_m. $isLive(e_{uv})$ denotes that the node u and the node v are connected with a live edge. An $(u, v) \in E$ edge is a live edge, if the node v influenced the node u, namely u becomes active from inactive for at least Q/r times out of Q simulations of the previous step. (In the original paper, the r was set to 2, however, during the evaluation we experiments additional values.) The second phase of the CGA algorithm is a dynamic programing method for selecting the communities which includes the best candidates. To mine the k^{th} seed, the method chooses the community that will yield the largest increase of influence degree. Any existing algorithms can be used to calculate the influence in the chosen community. The CGA algorithm is the basis of our solution which is described in the next section.

3. LouvainGreedy

In this section, we present our solution, namely the LouvainGreedy algorithm, to solve the influence maximization problem. Our algorithm is based on the Community-based Greedy algorithm with two modifications.

First, the clustering phase was replaced by a lately introduced community detection method called Louvain method presented in [4]. The Louvain method is a hierarchical agglomerative community detection algorithm which uses modularity maximization. The modularity measures the quality of a partition; and it is defined as in the following:

$$Q = \frac{1}{2m} \sum_{i,j} [A_{ij} - \frac{k_i k_j}{2m}] \delta(c_i, c_j),$$

where A denotes the weighted adjacency matrix of the graph,

$$A_{ij} = \begin{cases} weight(e_{ij}), & \text{if } e_{ij} = (v_i, v_j) \in E \\ 0 & \text{otherwise.} \end{cases}$$

$k_i = \sum_j A_{ij}$ denotes the degree of node v_i, $m = \frac{1}{2} \sum_{ij} A_{ij}$ denotes the total weight of the edges, and c_i, c_j denotes the cluster of the node v_i and v_j respectively, δ is the Kronecker delta

$$\delta(c_i, c_j) = \begin{cases} 1, & \text{if } c_i = c_j, \\ 0 & \text{otherwise.} \end{cases}$$

The algorithm consists of a label propagation and a node merging step. Initially, each node has a unique label. Next, each node adopts the community label of its neighbors, if the overall modularity increasing with the label adoption. Namely, for

all neighbors j of a node i, the gain of modularity are evaluated mean by removing i from c_i and by placing it into c_j. Then node i is placed in the community for which the gain is maximum, but only if it is positive. This propagation step is repeated until a local maximum has been obtained. (Note, that the propagation may depend on in the order the nodes are processed.)

When a local maximum has been obtained, the nodes with the same community label are merged into one single node keeping the outgoing edges and transforming the inside edges into weighted self-loops. After the merging step, the label propagation starts again. These two steps are repeated iteratively. The process terminates when each node has a different label at the end of the label propagation step, since in that case, there are no more merge-able nodes. The process results a hierarchical decomposition of the input graph. Because of the simplicity of the algorithms, it can be computed extremely fast even in case of large graphs. Moreover, according to [8], it is one of the best modularity based community detection algorithm.

The second important modification that we made on the CGA is the replacement of the dynamic programming phase. In our solution, after a graph has been partitioned into communities, the most influential node is computed within each community using the NewGreedy algorithm. The node with the maximum influence degree is selected as the first member of the candidate set. Then, in the community that belongs to the selected node, the influence degree of the nodes are recomputed. The process is repeated until all the seeds are selected.

Note, that if in the k^{th} turn, a node v has been selected from the cluster C_v, then in the $(k+1)^{th}$ turn, the marginal gain of nodes that are not members of C_v remain unchanged. That is because we compute the influence of a node inside the cluster only. Therefore, for each u that $C_u \neq C_v$ the following holds $\sigma_{C_u}(S \cup \{u\}) = \sigma_{C_u}(S \cup \{v\} \cup \{u\})$, where S denotes the candidate set in the k^{th} turn and σ_{C_u} denotes the influence of a set inside C_u.

Algorithm 1 shows the pseudo code of our solution. Initialy, the seed set S is empty, and the Louvain mehod is called to compute the clusters or communities (line 2). Next, for each cluster (line 3-7) the subGraph submethod computes the subgraph which belongs to the cluster. A subgraph contains the nodes of a cluster and the edges among them, but the outgoing edges are not included. After the subgraphs are computed, the NewGreedy algorithm assigns the influence degree to each node within each subgraph. The node that has the maximum influence degree in the cluster is recorded by $C.max$. After this initialization, a process is repeated k times (the cardinality of the candidate set). The process (line 8-13) selects the cluster ($max\_cluster$) containing the most influential node ($max\_cluster.max$) in each step. The most influental node is added to the seed set S, and then the marginal gains of nodes in $max\_cluster$ are recomputed. The node with the maximum marginal gain within the cluster is refreshed. At the end of the process, the algorithms returns S which contains the selected seeds.

Algorithm 1 LouvainGreedy

Input: $G = (V, E, W)$, number of seeds k, activation probability p, MC count r;
Output: list of seeds S;

```
1: S ← the empty list
2: Clusters = Louvain(G)                          ▷ community detection
3: for all C ∈ Clusters do
4:     C.SG ← subGraph(G, C)
5:     NewGreedy(C.SG, p, r)   ▷ assign marginal gain to each node in cluster C
6:     C.max ← argmax_{v∈C}{v.influence}
7: end for
8: for i ← 1, k do
9:     max_cluster ← argmax_{C∈Clusters}{C.max.influence}
10:    S = S ∪ {max_cluster.max}
11:    NewGreedy(max_cluster.SG, p, r)   ▷ refresh marginal gains in cluster C
12:    max_cluster.max ← argmax_{v∈C}{v.influence}
13: end for
14: return S
```

4. Results and discussion

We compared our LouvainGreedy (LG) algorithm with the NewGreedy (NG) and the Community-based Greedy Algorithm to reveal how our modifications on CGA affect the running time and precision. Section 4.1 describes our experiments and Section 4.2 discusses the precision of the methods in details.

4.1. Experiments

In the comparison process, two real-life networks were used. The first, which is called NetPHY, is extracted from the arXiv[5] academic collaboration network by Wei Chen et al. [2]. It is constructed using the full paper list of Physics section from 1991 to 2003. Each node represents an author and an edge is added between two authors whenever they jointly wrote a paper. The numbers of nodes and edges are respectively 37 154 and 231 584. The second data set, which is referred EmailEnr[6], is derived from the Enron email network, which consists of around half million emails. Nodes represent email addresses and if an address i has sent at least one email to address j, then an undirected edge between i and j is contained in the graph. It consists of 36 692 nodes and 183 831 edges. The experiments were done on a server with 12-core 2.67 GHz Intel Xeon CPU and 24 GB memory.

All the three algorithms were re-implemented in Java 1.7. In the combination step of (CGA) we computed the live edges as follows. We performed the edge-deleting part of the NewGreedy algorithm 100 times and we recorded for each

[5] `http://arXiv.org`
[6] It is available at `http://research.microsoft.com/enus/people/weic/graphdata.zip`

edge that how many times it was not deleted in the resulted graphs. If an edge has remained intact at least 1/8 part of the simulation count, then the edge was marked as a live edge. In addition, we used the Gephi Toolkit[7] [9] implementation of the Louvain community detection algorithm in our solution.

Table 1 contains the results belonging to the NetPHY data set where the cardinality of the seed sets was 20, the activation probability was 0.02. In the greedy steps 10 000 Monte Carlo simulations were performed. The running times of the algorithms consist of a clustering and a greedy phase. The clustering phase can be performed in advance as a pre-processing step and its result is reusable afterwards. As the table shows, the main differences among the running times of the investigated algorithms are in the lengths of the greedy phases. That is because the size distributions of the resulted communities are significantly different in each clustering algorithm which affect the running time of the greedy phases as the greedy algorithms run faster on smaller graphs.

	Running time (sec)				**Influence**	
	clustering	**greedy**	**all**	**relatively**	**average**	**relatively**
LG	7	373	380	9.5%	890	97.2%
CGA	21	1203	1224	30.0%	915	99.9%
NG	–	4021	4021	100%	916	100%

Table 1: $NetPHY, k = 20, p = 0.02, MC = 10\,000$

The quality of results was tested by starting with 10, 000 random cascade diffusion processes and taking the average number of the influenced nodes at the end of the processes. It can be seen in Table 1, that our LouvainGreedy algorithm ran ten times faster than the NewGreedy and its precision loss was less then 3% of the result of the NewGreedy.

	Running time (sec)				**Influence**	
	clustering	**greedy**	**all**	**relatively**	**average**	**relatively**
LG	5	564	569	10.7%	4500	99.0%
CGA	524	4555	5079	95.1%	4535	99.7%
NG	–	5339	5339	100%	4547	100%

Table 2: $EmailEnr, k = 20, p = 0.02, MC = 10\,000$

Table 2 includes the results on EmailEnr data set with the same parameters as above. As can be seen, CGA is much slower on this data set. It is because EmailEnr network has one and a half times more edges than NetPHy. Moreover, the clustering steps of CGA results a cluster that contains approximately two-thirds of the nodes, therefore the running time of the greedy algorithm could not be decreased. However, our algorithm was ten times faster than NewGreedy with 1% loss of precision using this data set as well.

[7] `http://gephi.github.io/toolkit/`

4.2. Precision

In [3], Wang et al. proved that using the *CGA* algorithm, the influence degree of the resulted set $R(I)$ (where I is the resulted set) is $(1 - e^{-\frac{1}{1+\triangle d*\theta}})$ approximate by the influence degree of the optimal solution, denoted by $R(I^*)$, where θ is the threshold used in the combination step and $\triangle d$ is the maximal difference between the number of nodes affected by a node in the network and that in its community. That is $R(I) \geq (1 - e^{-\frac{1}{1+\triangle d*\theta}})R(I^*)$.

As can be seen, the approximation highly depends on the threshold of the combination phase, where the communities are combined based on the *combination entropy*. Therefore, we conducted experiments by applying the combination step of the CGA algorithm on the communities that are resulted by the Louvain community detection method. However, these experiments gives very similar running times and precisions as the original algorithm. This is because in the combination step many communities were merged as they combination entropy was above the threshold. The threshold was set to 0.3 as in the original paper.

However, our experiments described in the previous section show that the *LouvainGreedy* algorithm can achieve high precision without the combination step. We suppose that is because the other factor of the approximation, the $\triangle d$ that is the maximal difference between the influence degree of nodes affected by a node in the whole network and that in the community, remains low when the *Louvain method* is used. It suggests the nodes did not effect each other across the resulted communities.

As we saw in Section 3, the *Louvain method* is based on modularity maximization, which is a measure of the quality of a graph partition. Therefore, to give theoretical bound to the approximation factor of the *LouvainGreedy* algorithm, we should describe how the modularity affects the result. But it remains an open question. Although our experimental results are promising, without such a theoretical bounds, we can not be sure how precise result we have got.

5. Summary and future plans

We presented a new method for solving influence maximization problem which is based on the Community-based Greedy algorithm. Our method combines the Louvain method, a wildly used community detection algorithm, with the NewGreedy which is a greedy algorithm that approximates the optimal solution of the problem.

We compared the presented algorithms w.r.t. running time and quality of their results measured by the number of influenced nodes at the end of random cascade processes starting from the resulted seed sets. The experiments show that LouvainGreedy can run ten times faster than NewGreedy and the precision loss is less than five percent. However, our solution can not provide theoretical bound to the goodness of its result. Thus, we tested the Louvain community detection algorithm along with the combination step of the CGA, which merges communities

if their combination entropy is above a threshold. The tests showed that in the combination step a large community is formed because of the community merging. this has a significant effect on the running time as the greedy step is time consuming on large clusters.

In the future, we would like to improve the presented algorithms using parallization. The most consuming part of the presented algorithms is the performance of Monte Carlo simulations. Running these simulations in parallel can significantly reduce the computation time of the greedy steps. Currently, Apache Hadoop [6] and the Pregel [7] systems are under investigation for this purpose.

References

[1] Kempe, D., Kleinberg, J., and 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 '03, pp. 137–146. ACM, 2003.

[2] Chen, W., Wang, Y., and Yang, S., Efficient influence maximization in social networks. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 199-208. ACM, 2009.

[3] Wang, Y., Cong, G., Song, G., and Xie, K., Community-based greedy algorithm for mining top-k influential nodes in mobile social networks. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 1039–1048. ACM, 2010.

[4] Blondel, V. D., Guillaume, J., Lambiotte, R., and Lefebvre, E., Fast unfolding of communities in large networks. In *Journal of Statistical Mechanics: Theory and Experiment*, Vol. 10 (2008): P10008.

[5] Leskovec, J., Krause, A., Guestrin, C., Faloutsos, C., VanBriesen, J., and Glance, N., Cost-effective outbreak detection in networks. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '07, pp. 420-429. ACM, 2007.

[6] White, T., *Hadoop: The Definite Guide*. O'Reailly Media, 2009.

[7] Malewicz, G., Austern, M. H., Bik, A. J., Dehnert, J. C., Horn, I., Leiser, N., and Czajkowski, G., Pregel: a system for large-scale graph processing. In *Proceedings of the 2010 ACM SIGMOD International Conference on Management of Data*, pp. 135-146. ACM, 2010.

[8] Lancichinetti, A., and Fortunato, S., Community detection algorithms: a comparative analysis. *Physical review E*, Vol. 80(5) (2009): 056117.

[9] Bastian, M., Heymann, S. and Jacomy, M., Gephi: an open source software for exploring and manipulating networks. In *Proceedings of the third International Conference on Weblogs and Social Media*, pp. 361-362, 2009.

[10] Kósa, B., Rácz, G., Pinczel, B. and Kiss, A., Properties of the Most Influential Social Sensors. In *Proceedings of 2013 IEEE 4th International Conference on Cognitive Infocommunications (CogInfoCom)*, pp. 469-474. IEEE, 2013.