Parallel Seed Selection for Influence Maximization Based on *k-shell* Decomposition

Hong Wu[1,2], Kun Yue[1](✉), Xiaodong Fu[3], Yujie Wang[1], and Weiyi Liu[1]

[1] School of Information Science and Engineering, Yunnan University, Kunming, China
kyue@ynu.edu.cn
[2] School of Information Engineering, Qujing Normal University, Qujing, China
[3] Faculty of Information Engineering and Automation, Kunming University of Science and Technology, Kunming, China

Abstract. Influence maximization is the problem of selecting a set of seeds in a social network to maximize the influence under certain diffusion model. Prior solutions, the greedy and its improvements are time-consuming. In this paper, we propose candidate shells influence maximization (*CSIM*) algorithm under heat diffusion model to select seeds in parallel. We employ *CSIM* algorithm (a modified algorithm of greedy) to coarsely estimate the influence spread to avoid massive estimation of heat diffusion process, thus can effectively improve the speed of selecting seeds. Moreover, we can select seeds from candidate shells in parallel. Specifically, First, we employ the *k-shell* decomposition method to divide a social network and generate the candidate shells. Further, we use the heat diffusion model to model the influence spread. Finally, we select seeds of candidate shells in parallel by using the *CSIM* algorithm. Experimental results show the effectiveness and feasibility of the proposed algorithm.

Keywords: Parallel · Social networks · Influence maximization · *K-shell* decomposition

1 Introduction

With the rising popularity of online social works (OSNs) such as Facebook, Twitter and WeChat and etc., OSNs play a critical role range from the dissemination of information to the adoption of political opinions and technologies [1, 2]. OSNs can be ubiquitously used to various applications, e.g., viral marketing, popular topic detection, and virus prevention [3]. A problem that received considerable attention in this context is that of influence maximization, first proposed by Domingos et al. [4, 5] and formulated by Kempe et al. [6].

Formally, given a social network $G = (V, E)$, budget k and a stochastic model, the problem of influence maximization is to find a k-node set of maximizing the influence spread under certain stochastic model. Kempe et al. [6] proposed two classic diffusion models: linear threshold model (LTM) and independent cascade model (ICM), and they proved the influence maximization problem under these two diffusion models is

NP-hard. Further, it was proved that the objective function of influence spread under these two diffusion models is monotone and submodular, and thus the greedy algorithm can be used to approximately select the optimal seed set based on the theory of [7]. However, the greedy algorithm is time consuming. Consequently, extensive follow-up studies along with the above work were launched [7–13] and mainly focus on improving the greedy algorithm or proposing new heuristic algorithm.

Despite the immense progress has been made in the past decades, parallel seed selection is also challenging. Actually, we can obtain the seed set timely by selecting seeds in parallel. We consider the following scenario of viral marketing. A company develops a new product and wants to advertise this new product via viral marketing within a social network. If the advertiser takes weeks to select some initial user as seeds to provide them free sample or discount to promote products, then they may lose their superiority because of non-timeliness [14].

It is known that the *k-shell* decomposition method partitions a network into sub-structures, and this process assigns an integer index k_s to each node, where the index k_s represents its location according to successive layers (i.e., shells) in the network [18]. The *k-shell* decomposition can depict the structure feature of social network and discover the layer feature [19]. We can further obtain multiple candidate shells, which are independent with each other. We further select seeds of multiple candidate shells in parallel. In this paper, we mainly discuss the problem of parallel seed selection for influence maximization based on *k-shell* decomposition. For this purpose, we need consider the following questions:

(1) How to model the influence spread (i.e., diffusion model)?
(2) How to obtain the *k-shell* structure of social network?
(3) How to select seeds in parallel for influence maximization?

For the question (1), we adopt the heat diffusion model presented by Ma et al. [15] due to its time-dependent property, which can simulate the product adoptions step by step and help companies divide their marketing strategies in to several phases. For example, a company may want to know the production adoption incurred by the initial user (i.e., seeds) in two days, five days or a week, etc.

For the question (2), we first borrow the idea from [16–18] and divide the social network by employing the method of *k-shell* decomposition. We further obtain the candidate shells and the number of their seeds based on the number of nodes in shell and the value of k_s (i.e., a k shell with index k_s).

For the question (3), we propose candidate shells influence maximization (*CSIM*) algorithm to select seeds in parallel based on the GraphX framework on Spark [20]. The influence maximization problem based on heat diffusion model is NP-hard, and the greedy algorithm can approximate the optimal result with $1-1/e$ [15]. In this paper, we employ the *CSIM* algorithm (a modified algorithm of greedy) to coarsely estimate the influence spread based on the seed set, the active set and non-seed nodes, which can avoid massive estimation of heat diffusion process, thus can effectively improve the speed of selecting seeds. Specifically, we first select the max-degree nodes of candidate shells in parallel as the first seed. For any shell, if its $n(k_s = i) = j > 1$, here, $n(k_s = i)$ denotes the number of seeds with index of shell $k_s = i$, then we compute the mean of shortest distance (MSD) from seed set to its active set. Further we compute the number of neighbors of v in MSD range. Here, v excludes from the seed set. Finally, we compute the value that the number of neighbors of v in MSD range subtracts the number of intersection of the neighbors of v in MSD range and the active set, and select node v_j with the maximum value added into the seed set as the jth seed node.

Experimental results on real-world social networks show the effectiveness and feasibility of the method proposed in this paper.

2 Related Work

With the popularity of online social networks, diffusion models have received considerable attention. In addition to the heat diffusion model adopted in this paper, there are some classic diffusion models. Kempe et al. [6] proposed two diffusion models: linear threshold (LT) model and independent cascade (IC) model. In the LT model, if the total weight from active in-neighbors reaches the threshold of a node, then this node is activated. In the IC model, an active node tires to active its inactive out-neighbors with a given probability, and this activation process is independent with other activations. Comparing to the LT model and IC model, the heat diffusion model adopted in this paper is a realistic model, which can predict the future behavior of the social network (e.g., Amazon networks) since it includes the time factor.

Kempe et al. [6] proved that the objective function under the LT model and IC model is NP-hard, and they further prove the monotonicity and submodularity of this objective function, thus the greedy algorithm can be used to approximately select seeds based on the theory of Nemhauser et al. [7]. However, the greedy algorithm is time consuming. Aimed at addressing this issue, many follow-up studies tried to improve the greedy algorithm or propose new heuristics [8–14, 21–23]. In terms of greedy selection, our *CSIM* algorithm is similar to the Core Covering Algorithm of [21], which is assigned a covering distance, but our *CSIM* algorithm estimates the influence spread based on the active set of seed set and converge area of non-seed set, and the computation of our converge area by using the MSD not assigning the covering distance. Moreover, in the aspect of selecting seeds, we select seeds from the candidate shells in parallel.

The approaches of graph analysis have high computation complexity in large-scale graphs [24]. In order to solve this problem, some frameworks have appeared including Hama [25], Giraph [26], GraphLab [27], GraphX [28] and etc. In terms of parallel framework, we adopt the GraphX, which combines the advantages of both data-parallel and graph-parallel systems by efficiently expressing graph computation within the Spark data-parallel framework and extends Spark's Resilient Distributed Dataset (RDD) abstraction to introduce the Resilient Distributed Graph (RDG) and leverage advances in data-flow systems to exploit in-memory computation and fault-tolerance.

3 Heat Diffusion Model

A social network is modeled as an undirected graph $G = (V, E)$, where V is the set of nodes representing individuals, and E is the set of edges representing the relationships between the individuals. The heat diffusion model can be formulated as follows [15]

$$f(t) = (I + \frac{\alpha t H}{P})^P \times f(0) \tag{1}$$

where I, α, H, P and $f(0)$ is the identity matrix, thermal conductivity-heat diffusion coefficient, matrix, positive integer and initial distribution of heat respectively.

Here, H is denoted as

$$H = \begin{cases} 1, & (v_i, v_j) \in E \quad or \quad (v_j, v_i) \in E \\ -d(v_i), & i = j \\ 0, & otherwise \end{cases} \tag{2}$$

where $d(v_i)$ is the degree of node v_i.

Given the activation threshold θ at time t, if the amount heat of node v_i exceeds θ, then node v_i is active.

4 Generating Candidate Shells and Selecting Seeds Based on Candidate Shells in Parallel

In this section, we first give the approach of k-shell decomposition. Further, we generate the candidate shells based on the value of shell and the number of nodes in shell. Finally, we select the seeds of candidate shells in parallel.

4.1 Generating Candidate Shells

We first introduce the basic idea of *k-shell* decomposition [18]. We first remove the nodes with degree k = 1. After removing the nodes with degree k = 1, some nodes may be left with degree k = 1, so we continue pruning the system iteratively until there is no node with degree k = 1 left. The removed nodes along with the corresponding links for a *k shell* with index k_s = 1. Similarly, we continue removing higher-k shells until all nodes are removed.

We then generate the candidate shells based on the value of k_s and the number of nodes in shell. Given the number of seeds (i.e., k), we select $k_1 = ck$ seeds based on the number of nodes in shell, and select $k_2 = k{-}ck$ seeds based on the index of k_s. We employ Eq. (3) to compute the number of seeds based on the number of nodes in each shell as follows

$$k_1(k_s = i) = k_1 \times \frac{n(k_s = i)}{n(G)} \tag{3}$$

where $k_1(k_s = i)$, $n(k_s = i)$ and $n(G)$ denote the number of seeds with $k_s = i$, the number of nodes with $k_s = i$ and the number of nodes with social network $G = (V, E)$. If $k_1(k_s = i)$ is equal to 0, then we will do not select the seeds based on the number of nodes.

We further compute the number of seeds in each shell by using Eq. (4) based on the value of k_s. The maximum of k_s is *max*, then we only consider these shells range from *max* to *max-m*. Here, the number of seeds of k_s = *max-i* is $\beta_{max\text{-}i} \times k_2$.

$$k_2 = (\beta_{max} + \ldots + \beta_{max-i} \cdots + \beta_{max-m}) \times k_2 \tag{4}$$

Here, $\beta_{max} + \ldots + \beta_{max\text{-}i} \ldots + \beta_{max\text{-}m} = 1$ and $\beta_{max} > \ldots > \beta_{max\text{-}i} \ldots > \beta_{max\text{-}m}$. Then, we can filter some shells with $k_1(k_s = i) = 0$ and $k_s \leq max\text{-}i-1$, and there are n' candidate shells left.

4.2 Selecting Seeds of Candidate Shells

We first select the nodes of candidate shells with maximum degree as the first seed in parallel, i.e.,

$$S^1_{k_s=i} = argmax(d(v_j)) \quad v_j \in G_{k_s=i} \tag{5}$$

where $S^1_{k_s=i}$ denotes the first seed with index $k_s = i$.

For any shell, if $k_1(k_s = i) > 1$ or $k_2(k_s = i) > 1$, to obtain the *jth* seed, we first compute the *MSD* from $S^{j-1}_{k_s=i}$ to its active set $A_S(S^{j-1}_{k_s=i})$ by Eqs. (6) and (7).

$$SP(v_j \to A_s(v_j)) = \frac{\sum_{u \in A_s(v_j)} SP(v_j \to u)}{|A_s(v_j)|} \tag{6}$$

$$MSD = \frac{\sum_{v_j \in S^{j-1}_{k_s=i}} SP(v_j \to A_s(v_j))}{\left|S^{j-1}_{k_s=i}\right|} \tag{7}$$

We further compute the number of neighbors of $v_k \in G_{k_s=i} \backslash S^{j-1}_{k_s=i}$ with *MSD* steps, i.e., $|N_{msd}(v_k)|$ and the number of intersection between $N_{msd}(v_k)$ and $A_s(S^{j-1}_{k_s=i})$, i.e., $|N_{msd}(v_k) \cap A_s(S^{j-1}_{k_s=i})|$. Finally, we select the following v_k by computing Eq. (8) as the *jth* seed with index $k_s = i$.

$$v_k = argmax(|N_{msd}(v_k)| - \left|N_{msd}(v_k) \cap A_t(S^{j-1}_{k_s=i})\right|) \tag{8}$$

4.3 Parallel Algorithm for Seed Selection Based on Candidate Shells

Based on the above descriptions of selecting seeds of candidate shells, the algorithm for the candidate shells influence maximization (*CSIM*) is given in Algorithm 1. Ma et al. proved the greedy algorithm approximate the optimal result with $1 - 1/e$ [15]. However, the greedy algorithm is time consuming due to the massive matrix computation, and thus we propose the *CSIM* algorithm (a modified algorithm of greedy) to coarsely estimate the influence spread based on the seed set, the active set and non-seed nodes, which can avoid massive estimation of heat diffusion process. At the same time, we obtain the candidate shells by the *k-shell* decomposition, and these candidate shells are independent of each other, thus we can further efficiently select seeds of candidate shells in parallel. Specifically, the *CSIM* algorithm can be described as follows. First, we employ the *k-shell* decomposition approach to divide the social network $G = (V, E)$ (lines 1–3). For each shell, we compute its number of seeds based on its index k_s and prune these shells with $k_1(k_s = i) = 0$ and $k_2(k_s = i) \leq max\text{-}m-1$ to generate the candidate shells (lines 4–8). Finally, we select the seeds from candidate shells in parallel (lines 9–18).

```
Algorithm 1. CSIM (G, k)
Input: Graph of social network $G=(V, E)$; Number of seeds $k$; Parameters $\alpha, t, P, \theta$
Output: $S$: seed set
 // k-shell decomposition of social network $G=(V, E)$
 1: for each $v \in V$ do
 2:   construct $v$'s shell by k-shell decomposition in parallel
 3: end for   // candidate shells generation
 4: $k_1 \leftarrow ck$
 5: $k_2 \leftarrow k-k_1$
 6: if $k_1(k_s=i)=0$ and $k_2(k_s=i)=max-m-1$ then
 7:    prune these shells with $k_1(k_s=i)=0$ and $k_2(k_s=i) \leq max-m-1$
 8:    candidate shells $CS = \{ks'_1, ks'_2, \ldots, ks'_n\}$
       // selecting seeds of candidate shells in parallel
 9: for each $ks'_1$ in $CS$ do
10:   $n(k'_{s_i}) \leftarrow k_1(k_s = i) + k_2(k_s = i)$
11:   parallel for $ks'_i \in CS$ do
12:     $S^1_{k_s=i} \leftarrow argmax(d(v_j))$   $v_j \in G_{k_s=i}$
13:     execute the heat diffusion process on $G_{k_s=i}$ with seed set $S^{j-1}_{k_s=i}$
14:     if $n(k_{s_i}') > 1$ then
15:      for $n(k_{s_i}') = 2$ to $m$ do
16:        compute $MSD \leftarrow \frac{\sum_{\blacksquare_j \in S^{j-1}_{k_s=i}} SP(v_j \rightarrow A_t(v_j))}{|S^{j-1}_{k_s=i}|}$
17:        select $v_k$ with $v_k = argmax(|N_{msd}(v_k)| - |N_{msd}(v_k) \cap A_t(S^{j-1}_{k_s=i})|)$ as the $j$th
           seed in $k_s=i$
18:     end for
19:  end for
20:return $S$
```

5 Experimental Results

5.1 Experiment Setup

Datasets: We choose ca-GrQc, ca-HepPh and com-DBLP for our experiments [6, 29]. The ca_Hepth and Ca-GrQc are collaboration networks extracted from the e-print arXiv (http://www.arXiv.org). The former is extracted from the "High Energy Physics-Theory" and the latter is extracted from the General Relativity. The com-DBLP is a much larger collaboration network extracted from the DBLP (http://dblp.uni-trier.de/db/), which is Computer Science Bibliography Database. The nodes in these two networks are authors and an edge between two nodes means the two coauthored at least one paper.

Running environment: All algorithms were implemented in Scala. All experiments were conducted on a machine (i.e., master node) with 3.3GHZ 32-Core CPUs and 32 GB memory, and 10 machines (i.e., worker nodes) with 3.3GHZ 8-Core CPUs and 16 GB memory.

5.2 Performance Studies

We measured the following metrics: (1) the influence spread of *CSIM* algorithm, max-degree algorithm and random algorithm under heat diffusion model with 30 seeds; (2) the influence spread of *CSIM* algorithm with different activation threshold θ; (3) the influence spread with different flow duration t; (4) the Speed-up of *CSIM* algorithm; (5) the parallel efficiency of *CSIM* algorithm.

All the experiments presented in this paper use the heat diffusion model, where we specified two parameters the initial heat of each heat source and P. Here, we choose N/k as the amount of heat for each heat source, where N is the number of nodes in social network, k the number of seeds and $P = 30$.

(1) ***Influence spread of three algorithms***. Table 1 shows the influence spread of three algorithms with the following parameters on $\alpha = 1$, $t = 0.1$, $\theta = 0.01$ and $|k| = 30$. From Table 1, we can observe that the number of active nodes with *CSIM* algorithm is larger the max-degree and random algorithms. This is because max-degree algorithm does not consider the overlap between seeds and the random algorithm as baseline algorithm, some selected seeds cannot spread the influence effectively.

Table 1. The influence spread of different algorithms

Data	*CSIM*	Max-degree	Random
ca-GrQc	329	311	278
ca-HepPh	741	606	374

(2) ***Activation threshold θ***. Given $\alpha = 1$ and $t = 0.1$, Fig. 1 shows the influence spread of *CSIM* algorithm of 30 seeds with different activation threshold θ from

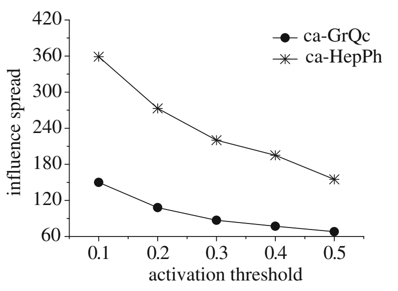

Fig. 1. Influence spread with different activation threshold

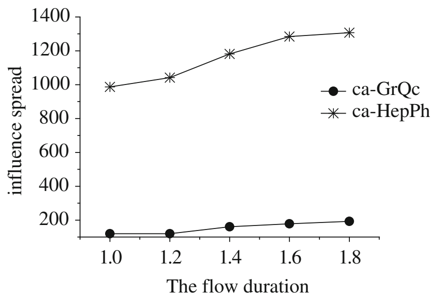

Fig. 2. Influence spread with different flow duration

0.1 to 0.5 with a span of 0.1, respectively. The x-axis indicates the activation threshold, and y-axis indicates the influence spread. From Fig. 1, we can see that the influence spread of *CSIM* algorithm will decrease with the increase of activation threshold θ. The reason is that the nodes are hard to be influenced when the activation threshold is larger.

(3) ***Flow duration t***. Given $\alpha = 1$ and $\theta = 0.5$, Fig. 2 shows the influence spread of *CSIM* algorithm of 30 seeds with different flow duration t from 1 to 1.8 with a span of 0.2, respectively. The x-axis indicates the flow duration, and the y-axis indicates the influence spread. From Fig. 2, we can obtain that the influence spread of *CSIM* will increase with the increase of flow duration t. This is because the larger flow duration t will lead to more nodes influenced.

(4) ***Speed-up***. Speed-up of a parallel algorithm is a ration of the processing time between of singer worker and multiple workers. Figure 3 shows the speed-up trends for parallel *CSIM* algorithm. In all datasets, the speed-up will increase as the number the workers increase.

(5) ***Parallel efficiency***. Figure 4 shows the trends for parallel *CSIM* with different datasets. To the same dataset, the parallel efficiency will decrease as the number of workers increase.

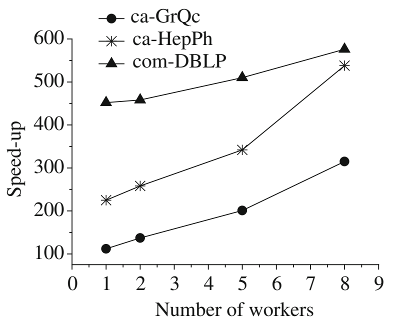

Fig. 3. Speed-up

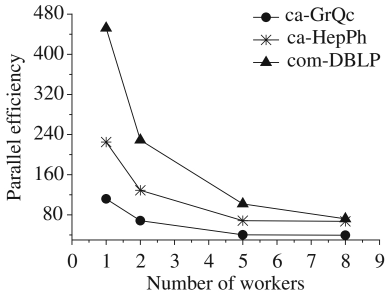

Fig. 4. Parallel efficiency

6 Conclusions and Future Work

To select seeds timely, we propose *CSIM* algorithm for parallel seed selection for influence maximization based on *k-shell* decomposition. First, we employ the *k-shell* decomposition method to divide a social network and generate the candidate shells. Further, we use the heat diffusion model to model the influence spread. Finally, we select seeds of candidate shells in parallel by using the *CSIM* algorithm. In a candidate shell, if the number of seeds is larger than 1, then we adopt the *CSIM* algorithm (a modified algorithm of greedy) to select seeds in parallel.

In this paper, we employ the value of index *k*-shell and the number of nodes to generate candidate shells based on experience. For our future work, we are to adopt modified algorithm to generate candidate shells. Moreover, the heat diffusion model in this paper only includes one kind of information spread, while there exist competitive influence spread in reality. Meanwhile, we are to analyze the competitive influence spread of heat diffusion model.

Acknowledgement. This work was supported by the National Natural Science Foundation of China (Nos. 61472345, 61462056, 61402398), Natural Science Foundation of Yunnan Province (Nos. 2014FA023, 2014FA028), Program for Excellent Young Talents of Yunnan University (No. XT412003), Research Foundation of the Education Department of Yunnan Province (Nos. 2014C134Y, 2016YJS005, 2016ZZX013).

References

1. Granovetter, M.: Threshold models of collective behavior. Am. J. Soc. **83**, 1420–1443 (1978)
2. Watts, D.J., Strogatz, S.H.: Collective dynamics of ‘small-world’ networks. Nature **393** (6684), 440–442 (1998)
3. Chen, W., Wang, C., Wang, Y.: Scalable influence maximization for prevalent viral marketing in large-scale social networks. In: SIGKDD, pp. 1029–1038 (2010)
4. Domingos, P., Richardson, M.: Mining the network value of customers. In: KDD, pp. 57–66 (2001)
5. Richardson, M., Domingos, P.: Mining knowledge-sharing sites for viral marketing. In: KDD, pp. 61–70 (2002)
6. Kempe, D., Kleinberg, J., Tardos É.: Maximizing the spread of influence through a social network. In: KDD, pp. 137–146 (2003)
7. Nemhauser, G.L., Wolsey, L.A., Fisher, M.L.: An analysis of approximations for maximizing submodular set functions—I. Math. Program. **14**(1), 265–294 (1978)
8. Leskovec, J., Krause, A., Guestrin, C., et al.: Cost-effective outbreak detection in networks. In: KDD, pp. 420–429 (2007)
9. Wang, C., Chen, W., Wang, Y.: Scalable influence maximization for independent cascade model in large-scale social networks. Data Min. Knowl. Disc. **25**(3), 545–576 (2012)
10. Cheng, S., Shen, H., Huang, J., Zhang, G., Cheng, X.: Staticgreedy: solving the scalability-accuracy dilemma in influence maximization. In CIKM, pp. 509–518 (2013)
11. Chen, W., Wang, Y., Yang, S.: Efficient influence maximization in social networks. In: KDD, pp. 199–208 (2009)

12. Chen, Y.C., Zhu, W.Y., Peng, W.C., Lee, W.C., Lee, S.Y.: CIM: community-based influence maximization in social networks. ACM Trans. Intell. Syst. Technol. **5**(2), 25 (2014)
13. Horel, T., Singer, Y.: Scalable methods for adaptively seeding a social network. In: WWW, pp. 441–451 (2015)
14. Chen, Y.C., Peng, W.C., Lee, S.Y.: Efficient algorithms for influence maximization in social networks. Knowl. Inf. Syst. **33**(3), 577–601 (2012)
15. Ma, H., Yang, H., Lyu, M.R., King, I.: Mining social networks using heat diffusion processes for marketing candidates selection. In: CIKM, pp. 233–242 (2008)
16. Bollobás, B.: Graph theory and combinatorics. In: Proceedings of the Cambridge Combinatorial Conference in honor of Paul Erdös, Academic, p. 35 (1984)
17. Carmi, S., Havlin, S., Kirkpatrick, S., Shavitt, Y., Shir, E.: A model of Internet topology using k-shell decomposition. In: PNAS, pp. 11150–11154 (2007)
18. Kitsak, M., Gallos, L.K., Havlin, S., Liljeros, F., Muchnik, L., Stanley, H.E., Makse, H.A.: Identification of influential spreaders in complex networks. Nat. Phys. **6**(11), 888–893 (2010)
19. Ren, Z.M., Liu, J.G., Shao, F., Hu, Z.L.: Guo, Q: Analysis of the spreading influence of the nodes with minimum K-shell value in complex networks. Acta. Phys. Sin **62**(10), 108902-1–108902-6 (2013)
20. https://spark.apache.org/docs/latest/graphx-programming-guide.html
21. Cao, J.X., Dong, D., Xu, S., Zheng, X., Liu, B., Luo, J.Z.: A *k*-core based algorithm for influence maximization in social networks. Chin. J. Comput. **38**(2), 238–248 (2015)
22. Song, G., Zhou, X., Wang, Y., Xie, K.: Influence maximization on large-scale mobile social network: a divide-and-conquer method. IEEE Trans. Parallel Distrib. Syst. **26**(5), 1379–1392 (2015)
23. Kim, J., Kim, S.K., Yu, H.: Scalable and parallelizable processing of influence maximization for large-scale social networks. In: ICDE, pp. 266–277 (2013)
24. Bello-Orgaz, G., Jung, J.J., Camacho, D.: Social big data: recent achievements and new challenges. Inf. Fusion **28**, 45–59 (2016)
25. Seo, S., Yoon, E.J., Kim, J., Jin, S., Kim, J.S., Maeng, S.: Hama: an efficient matrix computation with the mapreduce framework. In: CloudCom, pp. 721–726 (2010)
26. Avery, C.: Giraph: Large-scale graph processing infrastruction on hadoop. In: Hadoop Summit (2011)
27. Low, Y., Gonzalez, J.E., Kyrola, A., Bickson, D., Guestrin, C.E., Hellerstein, J.: Graphlab: a new framework for parallel machine learning. In: UAI, p. 10, (2014)
28. Xin, R.S., Gonzalez, J.E., Franklin, M.J., Stoica, I.: GraphX: a resilient distributed graph system on spark. In: GRADES, pp. 2:1–2:6 (2013)
29. Yang, J., Leskovec, J.: Defining and evaluating network communities based on ground-truth. In: ICDM, pp. 745–754 (2012)