An Accurate and Efficient Method to Detect Critical Links to Maintain Information Flow in Network

Kazumi Saito[1(✉)], Kouzou Ohara[2], Masahiro Kimura[3], and Hiroshi Motoda[4,5]

[1] School of Administration and Informatics, University of Shizuoka, Shizuoka, Japan
k-saito@u-shizuoka-ken.ac.jp
[2] Department of Integrated Information Technology, Aoyama Gakuin University, Sagamihara, Japan
ohara@it.aoyama.ac.jp
[3] Department of Electronics and Informatics, Ryukoku University, Kyoto, Japan
kimura@rins.ryukoku.ac.jp
[4] Institute of Scientific and Industrial Research, Osaka University, Suita, Japan
motoda@ar.sanken.osaka-u.ac.jp
[5] School of Computing and Information Systems, University of Tasmania, Hobart, Australia

Abstract. We address the problem of efficiently detecting critical links in a large network. Critical links are such links that their deletion exerts substantial effects on the network performance such as the average node reachability. We tackle this problem by proposing a new method which consists of one existing and two new acceleration techniques: redundant-link skipping (RLS), marginal-node pruning (MNP) and burn-out following (BOF). All of them are designed to avoid unnecessary computation and work both in combination and in isolation. We tested the effectiveness of the proposed method using two real-world large networks and two synthetic large networks. In particular, we showed that the new method can compute the performance degradation by link removal without introducing any approximation within a comparable computation time needed by the bottom-k sketch which is a summary of dataset and can efficiently process approximate queries, *i.e.*, reachable nodes, on the original dataset, *i.e.*, the given network.

1 Introduction

Studies of the structure and functions of large networks have attracted a great deal of attention in many different fields of science and engineering [9]. Developing new methods/tools that enable us to quantify the importance of each individual node and link in a network is crucially important in pursuing fundamental network analysis. Networks mediate the spread of information, and it sometimes happens that a small initial seed cascades to affect large portions of networks [11]. Such information cascade phenomena are observed in many situations: for example, cascading failures can occur in power grids (*e.g.*, the August

M. Kryszkiewicz et al. (Eds.): ISMIS 2017, LNAI 10352, pp. 116–126, 2017.
DOI: 10.1007/978-3-319-60438-1_12

10, 1996 accident in the western US power grid), diseases can spread over networks of contacts between individuals, innovations and rumors can propagate through social networks, and large grass-roots social movements can begin in the absence of centralized control (*e.g.*, the Arab Spring). These problems have mostly been studied from the view point of identifying influential nodes under some assumed information diffusion model. There are other studies on identifying influential links to prevent the spread of undesirable things. See Sect. 2 for related work.

We have studied this problem from a slightly different angle in a more general setting [10], that is to answer "Which links are most critical in maintaining a desired network performance?". For example, when the desired performance is to minimize contamination, the problem is reduced to detecting critical links to remove or block. If the desired performance is to maximize evacuation or minimize isolation, the problem is to detect critical links that reduce the overall performance if these links do not function. This problem is mathematically formulated as an optimization problem when a network structure is given and a performance measure is defined. In this paper, we define the performance to be the average node reachability with respect to a link deletion, *i.e.* average number of nodes that are reachable from every single node when a particular link is deleted/blocked. The problem is to rank the links in accordance with the performance and identify the most critical link(s).

Since the core of the computation is to estimate reachability, an efficient method of counting reachable nodes is needed. In our previous work we borrowed the idea of bottom-k sketch [4] which can estimate the approximate number of reachable nodes efficiently by sampling a small number of nodes. Our focus was how to implement bottom-k sketch algorithm to make it run fast and devised two acceleration techniques, called *redundant-link skipping* (*RLS*) and marginal-link updating (MLU). The difference from the previous paper is that, in this paper, we compute reachability accurately without resorting to approximation method, evaluate the accuracy of the approximation method and reduce the computation time to a comparable level by generalizing MLU which was tailored to bottom-k sketch, now called *marginal-node pruning* (*MNP*), and further devising the third technique called *burn-out following* (*BOF*).

We have tested our method using two real-world benchmark networks taken from Stanford Network Analysis Project and two synthetic networks which we designed to control the structural properties. We confirmed that the new method can compute the performance degradation by link removal without introducing any approximation within a computation time comparable to that needed by bottom-k sketch. We showed that depending on the network structure bottom-k sketch needs a larger k (than those used in the experiments) to obtain a result close to the correct one and in this case it needs much larger computation time even using all three techniques. We also analyzed which acceleration technique works better. The results depend on the network structure, but using all three together always works best.

The paper is organized as follows. Section 2 briefly explains studies related to this paper. Section 3 explains the proposed method with the three acceleration techniques. Section 4 reports four datasets used and the experimental results: computational efficiency and comparison with the results obtained by bottom-k sketch. Section 5 summarizes the main achievement and future plans.

2 Related Work

The problem of finding critical links in a network is related to the *influence maximization problem*, which has recently attracted much attention in the field of social network mining [3,6]. This is the problem of finding a limited number of influential nodes that are effective for the spread of information under an appropriate diffusion model. This problem is motivated by *viral marketing*, where in order to market a new product, we target a small number of influential individuals and trigger a cascade of influence by which their friends will recommend the product. Kempe et al. [6] introduced basic probabilistic diffusion models called the *independent cascade* and *linear threshold* models, and formulated this problem as a combinatorial optimization problem under these diffusion models. However, their algorithm was inefficient since it used the Monte-Carlo simulation to evaluate the influence function. Thereafter, a large number of studies have been made to reduce the running time to solve this problem, and several techniques [3,6] that efficiently prune unnecessary evaluations of the influence function have been proposed to speed up the greedy algorithm.

Recently, Borgs et al. [2] proposed an algorithm based on reverse reachability search, which can be regarded as a kind of *sketch-based* method, and proved that it runs in near liner time and provides theoretical guarantees on the approximation quality. Cohen et al. [5] presented another bottom-k sketch-based method of the greedy algorithm for influence maximization, which is called the greedy Sketch-based Influence Maximization (SKIM). Bottom-k sketch is a summary of dataset and can efficiently process approximate queries on the original dataset [4]. It is obtained by assigning a random value independently drawn from some probability distribution to each node. The bottom-k estimator requires the set of nodes having the k smallest values, and the kth smallest value is used for the estimation. This estimate has such a Coefficient of Variation (CV) that is never greater than $1/\sqrt{k-2}$ and well concentrated [4], where CV is defined by the ratio of the standard deviation to the mean. Moreover, for any $c > 0$, it is enough to set $k = (2+c)\epsilon^{-2} \log N$ to have a probability of having relative error larger than ϵ bounded by N^{-c} [4], where N is the number of nodes. The bottom-k sketch in the network can be efficiently calculated by reversely following links over the network. In Sect. 4 we compare the proposed method with the bottom-k sketch method.

The problem we pose is more closely related to efficiently preventing the spread of undesirable things such as contamination and malicious rumors. Many studies have been made on finding effective strategies for reducing the spread of infection by removing nodes in the network [1]. Moreover, there exists a study of

contamination minimization [7] that is converse to the influence maximization problem, where an effective method of minimizing the spread of contamination by blocking a small number of links was explored under a probabilistic information diffusion model. In this paper, we deal with the problem of efficiently finding critical links in terms of reachability degradation.

3 Proposed Method

Let $G = (\mathcal{V}, \mathcal{E})$ be a given simple directed network without self-loops, where $\mathcal{V} = \{u, v, w, \cdots\}$ and $\mathcal{E} = \{e, f, g, \cdots\}$ are sets of nodes and directed links, respectively. Each link e is also expressed as a pair of nodes, *i.e.*, $e = (u, v)$. Below we denote the numbers of nodes and links by $N = |\mathcal{V}|$ and $M = |\mathcal{E}|$, respectively. Let $\mathcal{R}(v; G)$ and $\mathcal{Q}(v; G)$ be the sets of reachable nodes by forwardly and reversely following links from a node v over G, respectively, where note that $v \in \mathcal{R}(v; G)$ and $v \in \mathcal{Q}(v; G)$. Also, let $\mathcal{R}_1(v; G)$ and $\mathcal{Q}_1(v; G)$ be the sets of those nodes adjacent to v, *i.e.*, $\mathcal{R}_1(v; G) = \{w \in \mathcal{R}(v; G) \mid (v, w) \in \mathcal{E}\}$ and $\mathcal{Q}_1(v; G) = \{u \in \mathcal{Q}(v; G) \mid (u, v) \in \mathcal{E}\}$, respectively. Now, let $G_e = (\mathcal{V}, \mathcal{E} \setminus \{e\})$ be the network obtained after removing a link $e = (v, w)$, then we can define the reachability degradation value with respect to $e \in \mathcal{E}$ as follows:

$$F(e; G) = \sum_{x \in \mathcal{V}} (|\mathcal{R}(x; G)| - |\mathcal{R}(x; G_e)|)/N. \tag{1}$$

In this paper, we focus on the problem of accurately and efficiently calculating $F(e; G)$ for every $e \in \mathcal{E}$. Of course, network performance measure is not unique. It varies from problem to problem, but computing $\mathcal{R}(v; G_e)$ for every node $v \in \mathcal{V}$ can be a fundamental task. Note that our proposed method and techniques can directly contribute to this task.

A simple method would be to straightforwardly compute the reachability size, *i.e.*, $|\mathcal{R}(x; G_e)|$, for every node $x \in \mathcal{V}$ after removing every link $e \in \mathcal{E}$. Let $R(G)$ be the average number of reachable nodes by forwardly following links over G, *i.e.*, $R(G) = \sum_{x \in \mathcal{V}} |\mathcal{R}(x; G)|/N$. Then, the computational complexity of this simple method is approximately $O(M \times N \times R(G))$ under the situation that $R(G) \approx R(G_e)$ for most links $e \in \mathcal{E}$, and it generally requires a large amount of computation for large-scale networks. In fact, we obtain $M \times N \times R(G) \in [2.6 \times 10^{13}, 2.0 \times 10^{14}]$ for our networks used in our experiment. In order to overcome this problem, we propose a new method by borrowing and extending the basic ideas of pruning techniques proposed in [8,10]. Below we revisit an existing technique called *redundant-link skipping* (*RLS*) [10] for the sake of readers' convenience. After that, we describe a revised technique called *marginal-node pruning* (*MNP*) which shares a basic idea of the marginal component pruning (MCP) technique proposed in [8] and the marginal link updating (MLU) tailored to bottom-k sketch in [10], and then propose a new acceleration technique called *burn-out following* (*BOF*).

3.1 RLS: Redundant-Link Skipping

The RLS technique selects each link $e \in \mathcal{E}$ for which $F(e;G) = 0$ and prune some subset of such links. Here, we say that a link $e = (v, w) \in \mathcal{E}$ is a *skippable link* if there exists some node $x \in \mathcal{V}$ such that $f = (v, x) \in \mathcal{E}$ and $g = (x, w) \in \mathcal{E}$, *i.e.*, $x \in \mathcal{R}_1(v;G) \cap \mathcal{Q}_1(w;G)$, which means $|\mathcal{R}_1(v;G) \cap \mathcal{Q}_1(w;G)| \geq 1$. Namely, we can skip removing the link e for the purpose of solving our problem due to $F(e;G) = 0$. Moreover, we say that a link $e = (v, w) \in \mathcal{E}$ is a *prunable link* if $|\mathcal{R}_1(v;G) \cap \mathcal{Q}_1(w;G)| \geq 2$. Namely, we can prune such a link e for our problem by setting $G \leftarrow G_e$ due to $F(f;G_e) = F(f;G)$ for any link $f \in \mathcal{E}$.

For each node $v \in \mathcal{V}$, let $\mathcal{S}(v)$ and $\mathcal{P}(v)$ be sets of skippable and prunable links from v. We can compute $\mathcal{S}(v)$ and $\mathcal{P}(v)$ as follows: for each child node $w \in \mathcal{R}_1(v;G)$, we first initialize $c(v, w;G) \leftarrow 0$, $\mathcal{S}(v) \leftarrow \emptyset$ and $\mathcal{P}(v) \leftarrow \emptyset$. Then, for each node $x \in \mathcal{R}_1(v;G)$, we repeatedly set $c(v, w;G) \leftarrow c(v, w;G) + 1$ and $\mathcal{S}(v) \leftarrow \mathcal{S}(v) \cup \{(v, w)\}$ if $w \in \mathcal{R}_1(x;G)$, and set $\mathcal{P}(v) \leftarrow \mathcal{P}(v) \cup \{(v, w)\}$ and $G \leftarrow G_{(v,w)}$ if $c(v, w;G) \geq 2$.

3.2 MNP: Marginal-Node Pruning

The MNP technique recursively performs pruning every node with degree 1 such that its in- and out-degrees are 1 and 0 (or 0 and 1), respectively. Let v be such a node with degree 1, *i.e.*, $|\mathcal{Q}_1(v;G)| = 1$ and $|\mathcal{R}(v;G)| = |\{v\}| = 1$ ($|\mathcal{R}_1(v;G)| = 0$). Then, after removing a link $e = (u, v)$, we can compute the reachability degradation value as $F(e;G) = |\mathcal{Q}(u;G)||\mathcal{R}(v;G)|/N$ where note that $|\mathcal{R}(x;G_e)| = |\mathcal{R}(x;G)| - 1$ if $x \in \mathcal{Q}(u;G)$; $|\mathcal{R}(x;G_e)| = |\mathcal{R}(x;G)|$ otherwise. Now, let $\eta(x)$ be the number of the pruned nodes which are reachable from node x, *i.e.*, after initializing $\eta(w) \leftarrow 1$ for each $w \in \mathcal{V}$, we count the number of forwardly reachable nodes $|\mathcal{R}(x;G_e)|$ by adding $\eta(w)$ when the node w is followed. Then, by updating $\eta(u)$ as $\eta(u) \leftarrow \eta(u) + \eta(v)$, we can recursively prune each node whose in- and out-degrees are 1 and 0 with keeping the accurate calculation of the reachability size for each node.

Clearly, we can apply the similar arguments for each node v such that in- and out-degrees are 0 and 1. Namely, after removing a link $e = (v, w)$, we can also compute the reachability degradation value as $F(e;G) = |\mathcal{Q}(v;G)||\mathcal{R}(w;G)|/N$. By introducing $\mu(x)$ for counting the number of reversely reachable nodes $|\mathcal{Q}(x;G_e)|$, just like $\eta(x)$, and updating $\mu(w)$ as $\eta(w) \leftarrow \eta(w) + \eta(v)$, we can recursively prune each node whose in- and out-degrees are 0 and 1 with keeping the accurate calculation of the reachability size for each node. Namely, we can see that the MNP technique can recursively perform pruning every node with degree 1.

3.3 BOF: Burn-Out Following

For a removed link $e = (v, w)$, we can state that $|\mathcal{R}(x;G)| = |\mathcal{R}(x;G_e)|$ if $x \notin \mathcal{Q}(v;G)$ or $x \in \mathcal{Q}(w;G_e)$. Namely, the reachable size of a node $x \in \mathcal{V}$ changes when x is reachable to v, *i.e.*, $x \in \mathcal{Q}(v;G)$, but becomes not reachable

to w after removing a link e, *i.e.*, $x \notin \mathcal{Q}(w; G_e)$. Thus, we can obtain a baseline method which computes the reachability size $|\mathcal{R}(x; G_e)|$ for every node $x \in \mathcal{Q}(v; G) \setminus \mathcal{Q}(w; G_e)$ after removing every link $e \in \mathcal{E}$. Then, by noting that $Q(G) = R(G)$ where $Q(G)$ means the average number of reachable nodes by reversely following links over G, we can see that the computational complexity of this baseline method can be bounded by $O(M \cdot R(G)^2)$. Thus, the baseline method also requires a large amount of computation for large-scale networks. In fact, we still obtain $M \times R(G)^2 \in [1.6 \times 10^{11}, 6.8 \times 10^{12}]$ for our networks used later.

The BOF technique further reduces the computation time needed to follow the same links repeatedly. More specifically, for each node $x \in \mathcal{Q}(v; G) \setminus \mathcal{Q}(w; G_e)$ that is utilized by the baseline method when a link $e = (v, w)$ is removed, we propose to compute the reachable size of x by $|\mathcal{R}(x; G_e)| = |\mathcal{R}(v; G_e)| + |\mathcal{R}(x; G_e) \setminus \mathcal{R}(v; G_e)|$. Namely, after calculating (burning out) the set of reachable nodes from v, *i.e.*, $\mathcal{R}(v; G_e)$, we can compute the reachable size $|\mathcal{R}(x; G_e)|$ by only following the nodes uniquely reachable from x, *i.e.*, $\mathcal{R}(x; G_e) \setminus \mathcal{R}(v; G_e)$. Below we summarize the BOF technique that computes $F(e; G)$ from a network G and its removal link $e = (v, w) \in \mathcal{E}$.

B1: Compute $\mathcal{R}(v; G_e)$ by forwardly following links from v over G_e, and if it happens that $w \in \mathcal{R}(v; G_e)$, output $F(e; G) \leftarrow 0$ and terminate.

B2: Compute $\mathcal{Q}(w; G_e)$ by backwardly following links from w over G_e, and then compute $\mathcal{Q}(v; G) \setminus \mathcal{Q}(w; G_e)$ by backwardly following each link x from v over G unless $x \in \mathcal{Q}(w; G_e)$.

B3: After initializing $F(e; G) \leftarrow 0$, for each node $x \in \mathcal{Q}(v; G) \setminus \mathcal{Q}(w; G_e)$, compute $|\mathcal{R}(x; G_e) \setminus \mathcal{R}(v; G_e)|$ forwardly following each link y from x over G_e unless $y \in \mathcal{R}(v; G_e)$, and then set $F(e; G) \leftarrow F(e; G) + |\mathcal{R}(v; G_e)| + |\mathcal{R}(x; G) \setminus \mathcal{R}(v; G_e)|$.

B4: Output $F(e; G) \leftarrow F(e; G)/N$ and terminate.

Here we should note that the above step **B1:** can be regarded as a generalized version of skippable link calculation by the *RLS* technique.

3.4 Summary of Proposed Method

In our proposed method referred to as PM, we apply the RLS, MNP and BOF techniques to the baseline method in this order, since it is naturally conceivable that the RLS and MNP techniques decrease the numbers of links and nodes in our network G. Clearly we can individually incorporate these techniques into the baseline method. Hereafter, we refer to the proposed method without the RLS technique as the \RLS method, the method without the MNP technique as the \MNP method, and the method without the BOF technique as the \BOF method. Since it is difficult to analytically examine the effectiveness of these techniques, we empirically evaluate the computational efficiency of the proposed method in comparison to these three other methods in which only two techniques are used and the remaining one not used.

4 Experiments

We evaluated the effectiveness of the proposed method using two benchmark and two synthetic networks as we did in [10]. Namely, we employed two benchmark networks obtained from SNAP (Stanford Network Analysis Project)[1]. The first one is a high-energy physics citation network from the e-print arXiv[2], which covers all the citations within a dataset of 34,546 papers (nodes) with 421,578 citations (links). If a paper u cites paper v, the network contains a directed link from u to v. The second one is a sequence of snapshots of the Gnutella peer-to-peer file sharing network from August 2002[3]. There are total of 9 snapshots of Gnutella network collected in August 2002. The network consists of 36,682 nodes and 88,328 directed links, where nodes represent hosts in the Gnutella network topology and links represent connections between the Gnutella hosts. In addition, we utilized two synthetic networks (around 35,000 nodes and 350,000 links) with a DAG (Directed Acyclic Graph) property, which were generated by using the DCNN and DBA methods described in [8], respectively. Here, networks generated by DCNN have both the small-world and scale-free properties, while those by DBA have only the scale-free property.

We refer to these two benchmark networks of citation and pear-to-pear and those generated by the DCNN an DBA methods as CIT, P2P, DCN and DBA networks. Table 1 summarizes the basic statistics of these networks, consisting of the numbers of nodes and links, N and M, the average number of reachable nodes $R(G)$, the number of nodes with in-degree 1 and out-degree 0, $|\mathcal{D}_{1,0}|$, the number of node with in-degree 0 and out-degree 1, $|\mathcal{D}_{0,1}|$, and the numbers of skippable and prunable links, $|\mathcal{S}|$ and $|\mathcal{P}|$. From this table, we can conjecture that the \BOF method will be comparable to the PM method for the DCN network because $R(G)$ is relatively small. On the other hand, the \MNU method may work poorly for the P2P network because $|\mathcal{D}_{1,0}|$ is relatively large, and the \RLS method may also work poorly for the CIT and DCN networks because $|\mathcal{S}|$ and $|\mathcal{P}|$ are relatively large. Here note that the numbers of skippable and prunable links in the DCN network inevitably become larger than the DBA network because the DCNN method has a link creation mechanism between potential pairs.

Table 1. Basic statistics of networks

Name	N	M	$R(G)$	$\lvert\mathcal{D}_{1,0}\rvert$	$\lvert\mathcal{D}_{0,1}\rvert$	$\lvert\mathcal{S}\rvert$	$\lvert\mathcal{P}\rvert$
CIT	34,546	421,578	14,059.0	469	858	302,248	176,224
DBA	35,000	351,317	12,225.1	1,651	1,649	85,815	24,690
DCN	35,000	350,807	2,137.6	2,943	2.839	289,398	175,211
P2P	36,682	88,328	8,482.6	16,409	24	1,502	29

[1] https://snap.stanford.edu/.
[2] https://snap.stanford.edu/data/cit-HepPh.html.
[3] https://snap.stanford.edu/data/p2p-Gnutella30.html.

4.1 Evaluation of Acceleration Techniques

First, we evaluated the efficiency of the proposed acceleration techniques by comparing the computation times of the \BOF, \MNP, \RLS, and the proposed (PM) methods. Figure 1 shows our experimental results which compares the actual processing times of these methods, where our programs implemented in C were executed on a computer system equipped with two Xeon X5690 3.47 GHz CPUs and a 192 GB main memory with a single thread within the memory capacity. From Fig. 1, we can clearly see that except for the DCN network, the \BOF method required much computation times compared with the other three methods. As described earlier, these experimental results can be naturally explained from our conjecture that the \BOF method would work well for the DCN network because $R(G)$ is relatively small. We can also see that the \MNU method exhibited the worst performance for the P2P network, while the \RLS for the DCN network. These experimental results are to be expected and explained from the $|\mathcal{D}_{1,0}|$ value and the pair of the $|\mathcal{S}|$ and $|\mathcal{P}|$ values in Table 1.

Which technique works best depends on the network characteristics. Overall BOF which was newly introduced in this paper works the best. MNP and RLS are similar and work less. The proposed method PM combining all the three techniques BOF, MNP and RLS is most reliable and produces the best performance, but the actual reduction of computation time depends on network structure. These results demonstrate the effectiveness of the proposed method.

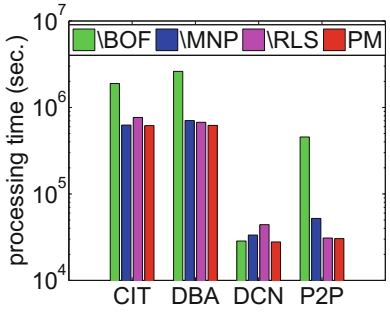

Fig. 1. Evaluation of acceleration techniques

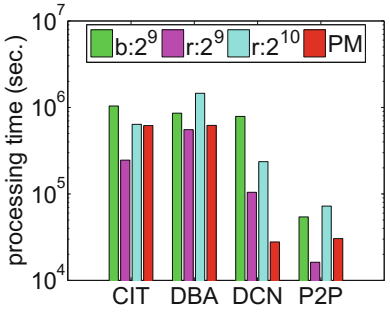

Fig. 2. Comparison with approximation method

4.2 Comparison with Approximation Methods

Next, we evaluated the efficiency of the proposed method in comparison to the approximation method based on bottom-k sketch [4]. The merit of bottom-k sketch is mentioned in Sect. 2. Here, we briefly revisit the implementation algorithm and describe the way to estimate the number of the reachable nodes from

each node $v \in \mathcal{V}$, *i.e.*, $|\mathcal{R}(v;G)|$. First, we assign to each node $v \in \mathcal{V}$ a value $r(v)$ uniformly at random in $[0,1]$. When $|\mathcal{R}(v;G)| \geq k$, let $\mathcal{B}_k(v;G)$ be the subset of the k smallest elements in $\{r(w) \mid w \in \mathcal{R}(v;G)\}$, and $b_k(v;G) = \max \mathcal{B}_k(v;G)$ be the k-th smallest element. Here, $\mathcal{B}_k(v;G)$ is set to $\mathcal{R}(v;G)$ when $|\mathcal{R}(v;G)| < k$. Then, we can unbiasedly estimate the number of the reachable nodes from v by $H(v;G) = |\mathcal{B}_k(v;G)|$ if $|\mathcal{B}_k(v;G)| < k$; otherwise $H(v;G) = (k-1)/b_k(v;G)$. We can efficiently calculate the bottom-k sketch $\mathcal{B}_k(v;G)$ for each node $v \in \mathcal{V}$ by reversely following links $k|\mathcal{E}|$ times. Namely, we first initialize $\mathcal{B}_k(v;G) \leftarrow \emptyset$ and sort the random values as $(r(v_1), \cdots, r(v_i), \cdots, r(v_{|\mathcal{V}|}))$ in ascending order, *i.e.*, $r(v_i) \leq r(v_{i+1})$. Then, from $i = 1$ to $|\mathcal{V}|$, for $w \in \mathcal{Q}(v_i;G)$, we repeatedly insert $r(v_i)$ into $\mathcal{B}_k(w;G)$ by reversely following links from v_i if $|\mathcal{B}_k(w;G)| < k$.

Based on the bottom-k sketches described above, we can estimate our reachability degradation value $F(e;G)$ as $J(e;G) = N^{-1} \sum_{v \in \mathcal{V}} (H(v;G) - H(v;G_e))$. Then, we can straightforwardly obtain a baseline approximation method which re-calculates the bottom-k sketches, $\mathcal{B}_k(v;G_e)$ with respect to G_e for all nodes each time from scratch. We can accelerate this baseline approximation method by introducing two acceleration techniques: RLS and MLU as mentioned in Sect. 1. RLS is the same as the first acceleration technique explained in Sect. 3. MLU locally updates the bottom-k sketches of some nodes when removing links incident to a node with in-degree 0 or out-degree 0 in the network G. Hereafter, the baseline approximation method is referred to as baseline BKS, while the revised approximation method that uses these two acceleration techniques as revised BKS.

Figure 2 shows our experimental results by setting the parameter k of the baseline BKS method to 2^9 and the revised BKS method to 2^9 and 2^{10}, denoted as b:2^9, r:2^9, and r:2^{10}, respectively. From these results, we can see that the proposed method substantially outperforms both the baseline BKS and the revised BKS methods for the DCN network. For the other networks, it is better than the baseline BKS method for $k = 2^9$ and the revised BKS method for 2^{10}. Below we will see that setting k at 2^{10} is not large enough to attain a good accuracy especially for the P2P network. We can say that the proposed method is competitive to the approximation method in terms of computation efficiency and has a merit of computing the correct values for reachability degradation.

The exact solutions obtained by our method can be used as the ground-truth for evaluating the approximation method. Let $E(m)$ be the set of the top-m links according to $F(e;G)$. Figure 3 shows the average relative error of the estimated value $J(e;G)$ over $E(5)$, *i.e.*, $\sum_{e \in E(5)} |1 - J(e;G)/F(e;G)|/5$, where we set k to one of $\{2^7, 2^8.2^9, 2^{10}\}$ for each network. From these experimental results, we observe that quite accurate estimation results were obtained for the CIT network, and the relative errors decreased monotonically by using a larger k. If we request the relative error to be less than 0.01, we need the parameter settings greater than $k = 2^8$. For other networks we observe that the results of the DBA and DCN networks are somewhat accurate around 0.1 when $k = 2^{10}$, but the results of the P2P network were quite inaccurate. We need much larger k and the computation time for BKS will overly exceed that of the present method.

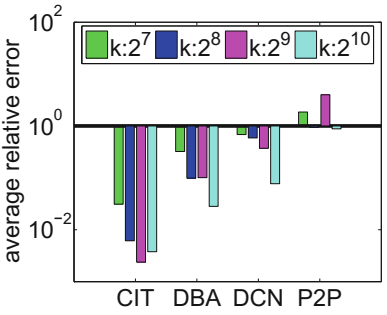

Fig. 3. Relative errors of approximation method

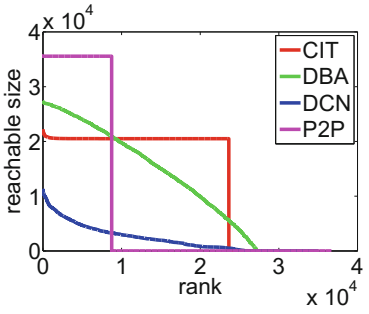

Fig. 4. Reachability distributions of networks

We discuss below why the BKS method worked very poorly for the P2P network. As a typical situation for a given removed link $e = (u, v) \in \mathcal{E}$, assume that $\mathcal{R}(w; G_e) \cap \mathcal{R}(v; G_e) = \emptyset$ for any $w \in \mathcal{Q}(u; G_e)$, then, we obtain $\mathcal{Q}(u; G) = \mathcal{Q}(u; G_e)$, $\mathcal{R}(v; G) = \mathcal{R}(v; G_e)$, and the reachability degradation value $F(e; G) = |\mathcal{Q}(u; G)| \times |\mathcal{R}(v; G)|/N$. However, when $|\mathcal{R}(u; G)| \geq k$, the BKS method returns its estimation as $J(e; G) = 0$ if $b_k(u; G) < \min_{w \in \mathcal{R}(v;G)} r(w)$, where recall that $r(w)$ and $b_k(u; G)$ mean a random value assigned to the node w and the k-th smallest element in $\{r(w) \mid w \in \mathcal{R}(u; G)\}$. This situation is likely to occur when $|\mathcal{R}(u; G)| \approx N$ and $|\mathcal{R}(v; G)|$ is quite small. On the other hand, when $|\mathcal{R}(u; G)| \approx N$, the BKS method may widely underestimate $J(e; G)$ if $b_k(u; G) > \min_{w \in \mathcal{R}(v;G)} r(w)$. This is because for a large number $|\mathcal{R}(u; G)|$ of nodes, say $x \in \mathcal{R}(u; G)$, $H(x; G_e)$ substantially decreases due to the removal of $\min_{w \in \mathcal{R}(v;G)} r(w)$ from $\mathcal{B}_k(x; G)$. We confirm that such situations are likely to occur on the P2P network. Figure 4 shows distributions of reachability size $|\mathcal{R}(v; G)|$ with respect to its rank. From this figure, we can clearly see that there exist two groups of nodes in the P2P network, those reachable to almost all of the other nodes, just like $|\mathcal{R}(u; G)| \approx N$ discussed above, and those reachable to almost only themselves, which includes the nodes in $\mathcal{D}_{1,0}$. These results clearly support our above explanation.

5 Conclusion

In this paper we have proposed a novel computational method that can detect critical links efficiently without introducing any approximation for a large network. The problem is reduced to finding a link that reduces the network performance substantially with respect to its removal. Such a link is considered critical in maintaining the good performance. There are many problems that can be mapped to this critical link detection problem, e.g. contamination minimization be it physical or virtual, evacuation trouble minimization, road maintenance prioritization, etc.

There are many things to do. Reachability computation is a basic operation and is a basis for many applications. We continue to explore techniques to further reduce computation time, to elaborate other useful network performance measures, and to clarify the difference in characteristics of extracted critical links from those of the links chosen by the existing measures such as edge betweenness that has no notion of reachability. Our immediate future plan is to apply our method to a real world application and show that it can solve a difficult problem efficiently, e.g. identifying important hot spots in transportation network or evacuation network.

Acknowledgments. This material is based upon work supported by the Air Force Office of Scientific Research, Asian Office of Aerospace Research and Development (AOARD) under award number FA2386-16-1-4032, and JSPS Grant-in-Aid for Scientific Research (C) (No. 17K00314).

References

1. Albert, R., Jeong, H., Barabási, A.L.: Error and attack tolerance of complex networks. Nature **406**, 378–382 (2000)
2. Borgs, C., Brautbar, M., Chayes, J., Lucier, B.: Maximizing social influence in nearly optimal time. In: Proceedings of the 25th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2014), pp. 946–957 (2014)
3. Chen, W., Lakshmanan, L., Castillo, C.: Information and influence propagation in social networks. Synth. Lect. Data Manag. **5**(4), 1–177 (2013)
4. Cohen, E.: Size-estimation framework with applications to transitive closure and reachability. J. Comput. Syst. Sci. **55**, 441–453 (1997)
5. Cohen, E., Delling, D., Pajor, T., Werneck, R.F.: Sketch-based influence maximization and computation: scaling up with guarantees. In: Proceedings of the 23rd ACM International Conference on Information and Knowledge Management, pp. 629–638 (2014)
6. Kempe, D., Kleinberg, J., Tardos, E.: Maximizing the spread of influence through a social network. Theory Comput. **11**, 105–147 (2015)
7. Kimura, M., Saito, K., Motoda, H.: Blocking links to minimize contamination spread in a social network. ACM Trans. Knowl. Discov. Data **3**, 9:1–9:23 (2009)
8. Kimura, M., Saito, K., Ohara, K., Motoda, H.: Speeding-up node influence computation for huge social networks. Int. J. Data Sci. Anal. **1**, 1–14 (2016)
9. Newman, M.: The structure and function of complex networks. SIAM Rev. **45**, 167–256 (2003)
10. Saito, K., Kimura, M., Ohara, K., Motoda, H.: Detecting critical links in complex network to maintain information flow/reachability. In: Proceedings of the 14th Pacific Rim International Conference on Artificial Intelligence, pp. 419–432 (2016)
11. Watts, D.: A simple model of global cascades on random networks. Proc. Natl. Acad. Sci. U. S. A. **99**, 5766–5771 (2002)