# Efficient computation of target-oriented link criticalness centrality in uncertain graphs

Kazumi Saito [a,*], Takayasu Fushimi [b], Kouzou Ohara [c], Masahiro Kimura [d] and Hiroshi Motoda [e]

[a] *Faculty of Science, Kanagawa University, Kanagawa, Japan*
*Center for Advanced Intelligence Project, RIKEN, Tokyo, Japan*
*E-mail: k-saito@kanagawa-u.ac.jp, kazumi.saito@riken.jp*
[b] *School of Computer Science, Tokyo University of Technology, Tokyo, Japan*
*E-mail: takayasu.fushimi@gmail.com*
[c] *College of Science and Engineering, Aoyama Gakuin University, Kanagawa, Japan*
*E-mail: ohara@it.aoyama.ac.jp*
[d] *Faculty of Advanced Science and Technology, Ryukoku University, Shiga, Japan*
*E-mail: kimura@rins.ryukoku.ac.jp*
[e] *Institute of Scientific and Industrial Research, Osaka University, Osaka, Japan*
*E-mail: motoda@ar.sanken.osaka-u.ac.jp*

**Abstract.** We challenge the problem of efficiently identifying critical links that substantially degrade network performance if they do not function under a realistic situation where each link is probabilistically disconnected, *e.g.*, unexpected traffic accident in a road network and unexpected server down in a communication network. To solve this problem, we utilize the bridge detection technique in graph theory and efficiently identify critical links in case the node reachability is taken as the performance measure.To be more precise, we define a set of target nodes and a new measure associated with it, *Target-oriented latent link Criticalness Centrality (TCC)*, which is defined as the marginal loss of the expected number of nodes in the network that can reach, or equivalently can be reached from, one of the target nodes, and compute $TCC$ for each link by use of detected bridges. We apply the proposed method to two real-world networks, one from social network and the other from spatial network, and empirically show that the proposed method has a good scalability with respect to the network size and the links our method identified possess unique properties. They are substantially more critical than those obtained by the others, and no known measures can replace the $TCC$ measure.

Keywords: Critical link, bridge detection, network analysis, uncertain graph

## 1. Introduction

Identifying critical nodes and links that play an important role in a large complex network is one of the essential and crucial issues in various fields including communication network analysis, urban design and evacuation planning, etc. For instance, in the field of social network mining [1, 2], detecting a limited number of influential nodes that are effective for widely spreading information is known as the influence maximization problem, which is motivated by viral marketing. Traditional centrality measures such as

*Corresponding author. E-mail: k-saito@kanagawa-u.ac.jp, kazumi.saito@riken.jp.

degree centrality and betweenness centrality are often used to quantify the importance of individual nodes and links. However, since these measures are based only on network topology, other factors that are more realistic, such as probabilistic information diffusion model (functional) and geodesic distance (structural) have also been used to assess a certain network performance metric [3–9].

The objective of these studies is identifying the most critical nodes/links in maintaining or maximizing a desired network performance, that is, identifying those nodes/links that degrade the performance substantially if they fail to function. For example, if the desired performance is maintaining an ability of information spread of a given social network, the critical links (information paths between two people) to be identified are those which maximally reduce the total number of people who can receive information issued by specific persons (information sources) when those links fail to pass the information. If the desired performance is maximizing evacuation or minimizing isolation in evacuation planning, the critical links (roads) to be identified are those which maximally reduce the total number of people who can reach one of the evacuation facilities when these links are blocked. This problem can be mathematically formulated as an optimization problem once a quantitative performance measure is defined when the network structure is given.

In reality, every link in a given network is not necessarily always functioning. In case of a social network some information paths between two persons may not be open due to server down or some other reason, and no information is passed from one to the other. In case of a road network some roads between two intersections may be blocked by traffic accident, road repair or nearby construction, and no persons and vehicles are allowed to pass. We do not know when these happen in advance. To account for this uncertainty, we assume that each link is disconnected probabilistically and propose a probabilistic link disconnection model. This is a standard approach taken in most studies on uncertain graphs. In other words, we consider a problem of identifying critical links associated with a set of target nodes specified in advance in a given uncertain network. Under this problem setting, we adopt, as the performance measure, the marginal loss of the expected number of nodes that can reach one of the target nodes from the rest of the nodes in the network, or equivalently the marginal loss of the expected number of nodes in the network that can be reached from one of the target nodes, when a particular link is disconnected. For example, assuming a social network, this measure quantifies how a link affects the overall communication to a set of people (target nodes) over the network if it does not pass information, and assuming a road network, this measure quantifies how a particular road connecting two intersections affects overall reachability to a set of facilities (target nodes) over the network if it stops allowing people and vehicles to pass. We refer to this marginal loss as the *Target-oriented latent criticalness centrality (TCC)* and propose a method to efficiently identify critical links, *i.e.*, links for which $TCC$ values are large.

We notice that bridge in graph theory is the key concept to solve this problem. A bridge is a link in a connected component such that its deletion divides the component into two disjoint ones. In our problem setting, if one of the disjoint components does not contain any target node, the marginal loss of the bridge is greater than 0 because no node in the component can reach, or equivalently can be reached from, any target node in the other component and thus, the link can be a candidate of critical links. We propose an algorithm that incorporates the standard bridge detection algorithm whose computational complexity is $O(|\mathcal{E}|)$, where $\mathcal{E}$ is a set of links in the network, and identify all bridges and compute their $TCC$ values efficiently. To the best of our knowledge, our approach is the first to identify critical links based on the bridge detection algorithm under a probabilistic link disconnection model.

Conference version of this paper appeared in [10], where we investigated the performance for two real world road networks in a evacuation problem setting with fixed sets of target nodes (evaluation facilities).

In this paper, we detail the explanation of the algorithm, but we do not repeat the same experiments. We redesign the experiments in a more general setting and show the new results for two different networks from two different domains, one from social and the other from spatial domains, and two different target node settings with a parameter to change their size. The results reconfirm what was reported in [10], and reveal new findings. We have applied a similar idea to identify critical nodes in uncertain graph [11, 12]. However, no notion of target nodes was introduced and the performance measure is the marginal loss of the expected total reachability. Thus, the problem setting is different.

Below we summarize our contributions in this paper. 1) We present a realistic problem of detecting critical links under a probabilistic link disconnection model. 2) We propose a new centrality: *Target-oriented latent link Criticalness Centrality* which is defined as the marginal loss of the expected number of nodes that can reach, or equivalently can be reached from, one of the target nodes. 3) We propose an efficient algorithm to compute $TCC$ value of each link based on bridge detection, whose computational complexity is $O(H \times |\mathcal{E}|)$, where $H$ is the number of networks generated from an original one based on the probabilistic link disconnection model (possible worlds). 4) We experimentally demonstrate the effectiveness of our proposed method for two real-world networks, Enron e-mail network and Washington DC Road network, using two different target node settings (medoid and random) with different target sizes. The results show that our method is much faster than computing the traditional centrality measures, has a good scalability with respect to the network size, and the links our method identified possess unique properties. They are substantially more critical based on $TCC$ measure than those obtained by the other centralities. No known measures can replace the $TCC$ measure.

The paper is organized as follows. Section 2 briefly explains the related work of this paper. Section 3 formulates the critical link detection problem based on a probabilistic link disconnection model. Section 4 presents the proposed method based on bridge detection. Section 5 shows experimental results on two real-world networks. Section 6 summarizes the main achievements and future directions.

## 2. Related Work

### 2.1. Centrality measure

A variety of node-centrality measures have been used to quantify the criticalness (importance) of each node/link. These include traditional centrality measures that are based only on network topology such as the degree centrality, the betweenness centrality [13] and PageRank centrality [14] and new measures that take into account other factors from functional view point, such as percolation centrality [15] which is defined by percolation states of individual nodes in a social network. Traditional centralities have often been used to analyze spatial networks embedded in the real world from structural viewpoints [7, 16, 17]. More problem specific performance measures have also been devised and used to quantify the criticalness of links. For example, typical performance measures for a road network are based on traveling time for a link or a path in the network [3, 4]. We proposed a performance measure based on the node reachability in our past work [6, 8], which is close to the one we use in this paper. There exist many studies on centrality measures over uncertain graphs [18–21] which do not focus on reachability. Our work is different from these existing studies. What we focus on in this paper is the critical link detection problem in a situation where links are probabilistically blocked.

Originally centrality is defined for each individual node or link. However, in order to deal with a more realistic situation such as information diffusion over a social network from multiple source nodes, it

becomes more important to examine which group of nodes are the most central. Thus, a notion of group centralities has been proposed [22] as an extension of traditional node-centrality measures, *e.g.*, group degree centrality, group closeness centrality and group betweenness centrality. Among them the group closeness centrality is closely related to real world problems such as location planning of commercial or evacuation facilities in a wide area [9]. In this paper, to quantify the criticalness of a link, we measure the network performance by the node reachability from/to a set of target nodes. Thus, the problem we tackle in this paper is also related to maximizing a group centrality since the performance measure we adopt can be a kind of group closeness centrality for the set of target nodes. $TCC$ is different in that we count the number of nodes based on reachability and the graph we treat is uncertain.

### 2.2. Critical link detection

There are a variety of investigations for mining critical links. Grady et al. [23] proposed the notion of link salience to extract the skeleton of a network as an extension of link betweenness centrality. Fang et al. [24] presented link capacity allocation methods to prevent cascading failures under limited investment costs for power transmission networks. Critical link identification problem has also been explored in various scenarios, including communication networks [25], wireless sensor networks [26] and water distribution systems [27]. On the other hand, the problem of injecting appropriate $k$ new links for improving network performance has been examined from various perspectives. Papagelis [28] and Parotsidis et al. [29] investigated a problem of minimizing the average shortest path distance over all pairs of nodes in a network by finding optimal $k$ new shortcut links. Crescenzi et al. [30] and Parotsidis et al. [31] addressed the problem of maximizing the closeness centrality of a specific node in a network by suitably selecting $k$ links to be injected. Chaoji et al. [32], Tong et al. [33] and Li et al. [34] explored a problem of promoting information diffusion in a social network by injecting new links. Ohara et al. [9] proposed a method to create new links that maximizes the gain of the group closeness centrality and applied it to evacuation problem. Unlike these previous studies, we focus on the node reachability from/to a set of target nodes as the network performance, and attempt to efficiently identify critical links in an uncertain situation where links are probabilistically disconnected. We propose a new link-centrality measure, *Target-oriented latent link Criticalness Centrality*, which can be considered as a kind of vitality index [35], and develop a method of efficiently calculating the $TCC$ value of every link.

Our proposed method is based on the standard bridge detection algorithm [36] in graph theory. The bridge itself is critical in a network since its removal breaks the connectivity of the network. Thus, bridge detection is embedded into problems encountered in various fields such as wireless sensor networks [26], while it is also utilized to improve computational efficiency of conventional centrality measures [37]. However, these studies assume that the network structure is stable. Unlike these previous studies, our work deals with an uncertain graph [38, 39] where each link has independent disconnection probability. There have been substantial studies on uncertain graphs. Potamias et al. [40] designed an algorithm for efficiently answering $k$-nearest neighbor queries, and Liu et al. [41] presented a technique for reliable clustering. To the best of our knowledge, there is no work that uses the bridge detection technique to identify critical links under the probabilistic link disconnection model.

We addressed in [42] the contamination minimization problem in information diffusion that is closely related to the critical link detection in this paper. The aim of the problem is minimizing the spread of contamination via a network by blocking a small number of links. In this paper, we adopt the same idea in [42] and sample graphs from a given original network by deciding connectivity of each link according to the disconnection probability to estimate the expected number of nodes based on the reachability from/to

a set of prespecified nodes. However, the difference is that we succeeded in substantially reducing the number of sampled graphs by computing the difference of reachability sizes for both connected and disconnected cases for each link, given a sampled graph, while the method in [42] requires much larger number of samples for each link as it computes the expected value of both when it is connected and disconnected.

## 3. Problem Formulation

Let $G = (\mathcal{V}, \mathcal{E})$ be a given simple undirected (or bidirectional) network without self-loops, where $\mathcal{V} = \{u, v, w, \cdots\}$ and $\mathcal{E} = \{e, \cdots\}$ are sets of nodes and undirected links, respectively. We also express each link $e$ as a pair of nodes, *i.e.*, $e = \{u, v\}$. In our problem setting, we assume a fixed group of nodes $\mathcal{U} \subset \mathcal{V}$, called target nodes, such as people who serve as source nodes in information spread on a social network or evacuation facilities on a spatial network. Let $\mathcal{R}(u; G)$ be the set of reachable nodes by following links from a node $u$ over $G$, where note that $u \in \mathcal{R}(u; G)$. Then, we can define a set $\mathcal{R}(\mathcal{U}; G)$ of reachable nodes by following links from a node $u \in \mathcal{U}$ over $G$, *i.e.*, $\mathcal{R}(\mathcal{U}; G) = \bigcup_{u \in \mathcal{U}} \mathcal{R}(u; G)$. Since $G$ is an undirected network, this definition of $\mathcal{R}(\mathcal{U}; G)$ also means that a node $v \in \mathcal{R}(\mathcal{U}; G)$ can reach at least one of the target nodes $u \in \mathcal{U}$ by following links in $G$. For example, the direction of the information flow on a social network is from $\mathcal{U}$ (information source nodes) to $\mathcal{R}(\mathcal{U}; G)$ (information receiver nodes), whereas the direction of the flow of people in the case of disaster evacuation is from $\mathcal{R}(\mathcal{U}; G)$ (each point in a road network) to $\mathcal{U}$ (evacuation facilities).

For each link $e \in \mathcal{E}$, let $x_e$ be a random variable expressing the link connectivity, *i.e.*, $x_e = 1$ if the link $e$ is disconnected; otherwise $x_e = 0$, where we denote its disconnection probability by $p(x_e = 1) = p_e$. As performed in studies on uncertain graphs, we introduce such probabilities that each link is accidentally missed and disfunctions in order to reflect the realistic situation. For example, in terms of information diffusion on a social network, those probabilities are assigned according to some link down model based on users' network environments. In terms of disaster evacuation over a spatial network, they are assigned according to some road blockage model based on geographical properties. By using a set of random variables defined by $\mathcal{X} = \{x_e \mid e \in \mathcal{E}\}$, we can define a graph $G_{\mathcal{X}} = (\mathcal{V}, \mathcal{E}_{\mathcal{X}})$, where $\mathcal{E}_{\mathcal{X}} = \{e \mid e \in \mathcal{E}, x_e = 0\}$. Now, by assuming independent Bernoulli trials for all the links, we can compute the occurrence probability of each graph $G_{\mathcal{X}}$ by

$$p(G_{\mathcal{X}}) = \prod_{x_e \in \mathcal{X}} p_e^{x_e} (1 - p_e)^{1 - x_e}. \tag{1}$$

For each graph $G_{\mathcal{X}}$, let $G^+_{\mathcal{X}}(e)$ and $G^-_{\mathcal{X}}(e)$ be graphs constructed by adding and removing a link $e$, respectively, *i.e.*, $G^+_{\mathcal{X}}(e) = (\mathcal{V}, \mathcal{E}_{\mathcal{X}} \cup \{e\})$ and $G^-_{\mathcal{X}}(e) = (\mathcal{V}, \mathcal{E}_{\mathcal{X}} \setminus \{e\})$. Based on $G^+_{\mathcal{X}}(e)$ and $G^-_{\mathcal{X}}(e)$, we define the following reachability contribution value of a link $e \in \mathcal{E}$ over $G_{\mathcal{X}}$.

$$\phi_{\mathcal{X}}(e) = |\mathcal{R}(\mathcal{U}; G^+_{\mathcal{X}}(e))| - |\mathcal{R}(\mathcal{U}; G^-_{\mathcal{X}}(e))|. \tag{2}$$

Evidently, in order to obtain the expected reachability contribution value, we need to compute $\phi(e)$ defined as

$$\phi(e) = \langle |\mathcal{R}(\mathcal{U}; G^+_{\mathcal{X}}(e))| - |\mathcal{R}(\mathcal{U}; G^-_{\mathcal{X}}(e))| \rangle_{\mathcal{X} \setminus \{x_e\}} \tag{3}$$

where $\langle\cdot\rangle_{\mathcal{X}\setminus\{x_e\}}$ means an expected value taken for all the possible assignments to random variables except for $x_e$. Note that the expected value $\phi(e)$ can be interpreted as the expected number of nodes that become unreachable from any of the prespecified target nodes $\mathcal{U}$ when link $e$ is blocked under the condition that each link $e'$ other than $e$ is blocked with its probability $p_{e'}$. In the case of information diffusion on a social network, $\phi(e)$ represents the expected number of people who become unable to receive information issued by specific persons when an information path $e$ between two persons is down under the condition that each of the other paths $e'$ becomes non-functional with the probability $p_{e'}$. Here we should emphasize that by introducing some weight $\rho(v)$ for each node $v$ (which we set 1 uniformly for the sake of simplicity), we can straightforwardly extend our problem setting to ones that can take into account more realistic conditions. For example, we can find critical information paths for viral marketing over a social network under a more realistic condition if buying power of each person $v$ is given by its weight $\rho(v)$. In the case of evacuation planning, we can take into account population distribution in a city if $\rho(v)$ is defined as population around the node (junction). Then, $\phi(e)$ denotes the expected number of people who become unable to successfully move to any of the evacuation facilities when a road $e$ is blocked under the condition that each of the other roads $e'$ is blocked with the probability $p_{e'}$.

Evidently, it is difficult to exactly compute $\phi(e)$ defined in Eq. (3) due to a large number of possible network configurations, which amounts to $2^{|\mathcal{E}|}$ (possible worlds). Thus, we employ an approach based on Monte Carlo simulation. Let $\mathcal{H}$ be a set of integers defined by $\mathcal{H} = \{1, \cdots, H\}$. Now, we repeat simulations $H$ times based on the probabilistic model defined in Eq. (1), and sample a set $\mathcal{G}_H$ of $H$ graphs *i.e.*, $\mathcal{G}_H = \{G_h = (\mathcal{V}, \mathcal{E}_h) \mid h \in \mathcal{H}\}$, where $\mathcal{E}_h \subset \mathcal{E}$ is the set of non-disconnected links at the $h$-th simulation. Then, by defining $\phi_h(e)$ as follows:

$$\phi_h(e) = |\mathcal{R}(\mathcal{U}; G_h^+(e))| - |\mathcal{R}(\mathcal{U}; G_h^-(e))| \tag{4}$$

where $G_h^+(e) = (\mathcal{V}, \mathcal{E}_h \cup \{e\})$ and $G_h^-(e) = (\mathcal{V}, \mathcal{E}_h \setminus \{e\})$, we can define the reachability contribution value $F(e; \mathcal{G}_H)$ of a link $e \in \mathcal{E}$ for $\mathcal{G}_H$ as follows:

$$F(e; \mathcal{G}_H) = \frac{1}{H} \sum_{h \in \mathcal{H}} \phi_h(e) \tag{5}$$

Hereafter, we denote $F(e; \mathcal{G}_H)$ simply as $F_H(e)$ because the set of generated graphs $\mathcal{G}_H$ is fixed in our experiments. Figure 1 illustrates the overall process to compute $F_H(e)$. Evidently, $F_H(e)$ is an unbiased estimator, *i.e.*,

$$\langle F_H(e) \rangle_{\mathcal{H}} = \frac{1}{H} \sum_{h \in \mathcal{H}} \langle \phi_{\mathcal{X}}(e) \rangle_{\mathcal{X}} = \phi(e). \tag{6}$$

Thus, when $H$ is a sufficiently large number, $F_H(e)$ can be a close approximation to the expected reachability contribution value $\phi(e)$. In this paper, we refer to each contribution value $F_H(e)$ as the *Target-oriented latent link Criticalness Centrality* ($TCC$ for short) for a link $e \in \mathcal{E}$, and focus on the problem of accurately and efficiently calculating $F_H(e)$ for every $e \in \mathcal{E}$.

## 4. Proposed Method

To develop an effective algorithm for computing contribution value $F_H(e)$ for every $e \in \mathcal{E}$, we focus on the following two facts. For a generated graph $G_h = (\mathcal{V}, \mathcal{E}_h)$, 1) each link $e \in \mathcal{E}_h$ has a positive

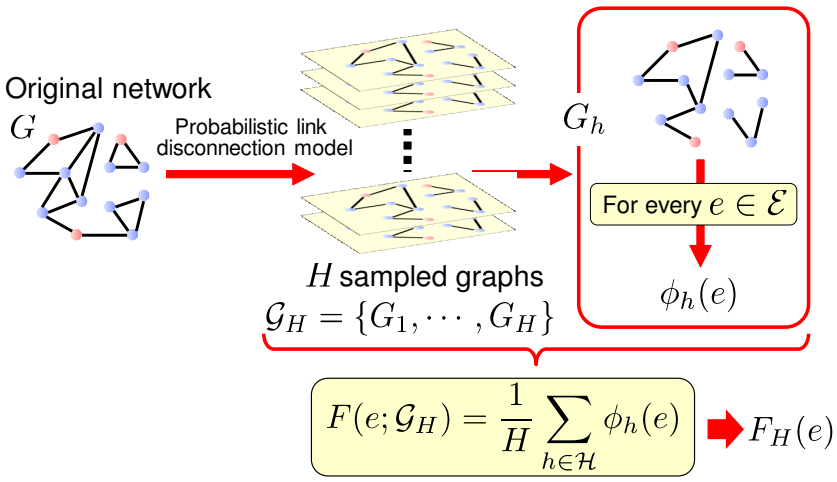



Fig. 1. Intuitive illustration of the problem setting of computing the expectation of the contribution value of each node in an uncertain network.

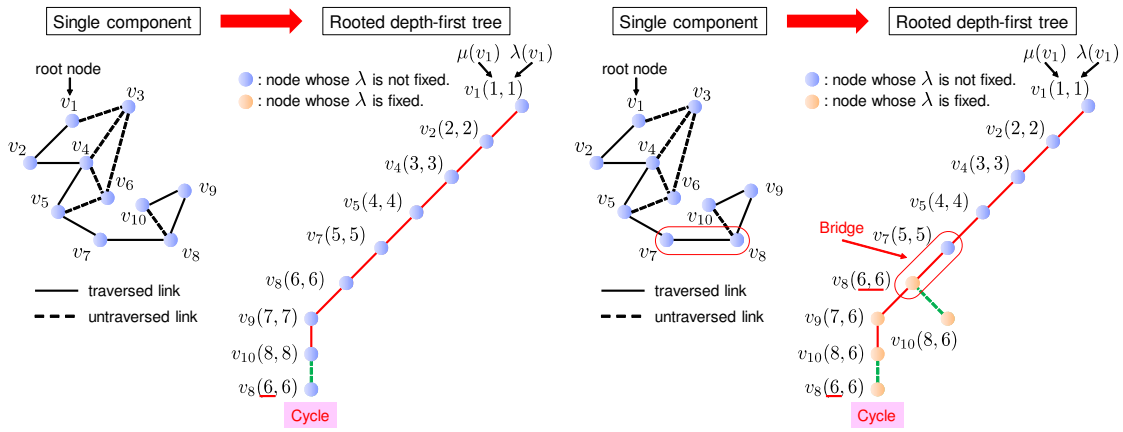
contribution value, *i.e.*, $\phi_h(e) = |\mathcal{R}(\mathcal{U}; G_h^+(e))| - |\mathcal{R}(\mathcal{U}; G_h^-(e))| > 0$, if $e$ is a bridge in a connected component in $G_h$, and only if one of the components which are separated after removing $e$, does not contain any node $u \in \mathcal{U}$; 2) each link $e \in \mathcal{E} \setminus \mathcal{E}_h$ has a positive contribution value if $e$ connects two connected components in $G_h$, and only if one of these components contains a node $u \in \mathcal{U}$. Hereafter, the connected components each containing at least one node $u \in \mathcal{U}$ are referred to as DR (definitely reachable) components, while the other components as DU (definitely unreachable) components.

In order to compute all bridges of $G_h$, we employ the idea of standard bridge detection algorithm originally proposed by Tarjan [36], which constructs a rooted depth-first tree for each connected component by traversing it from an arbitrary selected node $v \in \mathcal{V}$ in the depth-first manner, as described below. Here, we propose to select each target node *i.e.*, $u \in \mathcal{U}$ as the starting node. If a connected component contains other target nodes, these nodes are skipped. Then, in order to examine whether a bridge $e$ has a positive contribution value or not, we only need to examine by the depth-first search that the descendant part of the component after removing $e$ is a DU component because its ascendant part is guaranteed to be a DR component.

Here, we revisit the bridge detection algorithm. For a given connected graph $G = (\mathcal{V}, \mathcal{E}_h)$, let $\mu(v)$ be the order number assigned to each node $v \in \mathcal{V}$ during the depth-first search and $\lambda(v)$ be the minimum order number in a cycle that is detected during the search and contains node $v$, just as used in the standard bridge detection algorithm, which yields $\mu(w) = \lambda(w)$ if a link $e = \{v, w\}$ is a bridge. This means that any descendant node of $w$ cannot go to any ancestor nodes of $w$ without passing through the link $e$. Formally, the algorithm is described as follows: after initializing $\mu(v) \leftarrow -1$ for all $v \in \mathcal{V}$ and the ordering number as $Numb \leftarrow 1$, we select an arbitrary node $u \in \mathcal{V}$ as a root and perform the following depth-first search procedure $dfs(\epsilon, u)$, where $\epsilon$ stands for an empty node and means that $u$ has no parent node. As for the depth-first search procedure $dfs(v, w)$, after setting $\mu(w) \leftarrow Numb$, $\lambda(w) \leftarrow Numb$ and $Numb \leftarrow Numb + 1$, for every link $\{w, x\} \in \mathcal{E}_h$ such that $x \neq v$, we recursively perform the depth-first search procedure $dfs(w, x)$ if $\mu(x) = -1$, and set $\lambda(w) \leftarrow \lambda(x)$ if $\lambda(w) > \lambda(x)$. By applying this algorithm, we can construct a network consisting only of links $\{w, x\} \in \mathcal{E}_h$ over which the depth first-search procedure $dfs(w, x)$ is actually performed (traversed). Hereafter, we refer to this network as a rooted depth-first tree.

For example, let us consider constructing a rooted depth-first tree from a single component shown in the left in Fig. 2(a) by traversing it from node $v_1$ in the leftmost manner, which results in the tree shown in the right in Fig. 2(f). It is noted that although the order of traversing links of an undirected graph to visit all nodes it contains is not unique and there are multiple choices, all bridges in the graph can be identified correctly no matter which order we may choose. As shown in Fig. 2(a), each node of the rooted depth-first tree is basically given its node name and a pair of values $\mu(\cdot)$ and $\lambda(\cdot)$. For example, $(1,1)$ for node $v_1$ stands for $\mu(v_1) = 1$ and $\lambda(v_1) = 1$. As mentioned previously, for every node $v$, $\lambda(v)$ is initialized to $Numb = \mu(v)$, and, during the construction of the tree, it is updated to $\lambda(w)$ if $\lambda(w) < \lambda(v)$ holds for its adjacent node $w$ such that link $\{v,w\}$ exists in the component and $w$ is not the parent node of $v$ in the depth-first tree. For example, when visiting node $v_{10}$ first time by doing $dfs(v_9, v_{10})$, $\lambda(v_{10})$ is set to 8 as shown in Fig. 2(a). $v_{10}$ has two adjacent nodes $v_8$ and $v_9$, among which $v_9$ is its parent node in the depth-first tree. Thus, $v_8$ is a unique candidate to search at the next step. But, $\mu(v_8)$ is 6 and not $-1$, which means $v_8$ has already been visited and a cycle is detected. Therefore, without performing the depth-first search $dfs(v_{10}, v_8)$, we update $\lambda(v_{10})$ to $\lambda(v_8) = 6$ because $\lambda(v_8) < \lambda(v_{10})$ holds. Then, as for node $v_9$, after performing the depth-first search $dfs(v_9, v_{10})$, we need to update $\lambda(v_9)$ to $\lambda(v_{10}) = 6$ as shown in Fig. 2(b) because $\lambda(v_{10}) < \lambda(v_9)$. Intuitively, when detecting a cycle during the depth-first search, the minimum value of $\lambda(\cdot)$ in the cycle, 6 in this case, propagates from child node to its parent on the depth-first tree through the backtracking process of the search. In this example, after the backtrack, the values of $\lambda(v_{10})$ and $\lambda(v_9)$ are fixed and never changed because they have no further adjacent nodes to visit. $v_8$ has another adjacent node $v_{10}$, but we can find $v_{10}$ has already been visited because $\mu(v_{10}) \neq -1$. Since $v_8$ has no further adjacent nodes to visit, we backtrack to node $v_7$ and fix $\lambda(v_8) = 6$. Then, we can detect the first bridge $\{v_7, v_8\}$ because of $\mu(v_8) = \lambda(v_8) = 6$ as shown in Fig. 2(b). Similarly, after backtracking to node $v_5$, we can detect the second bridge $\{v_5, v_7\}$ because $\mu(v_7) = \lambda(v_7) = 5$ as shown in Fig. 2(c). After traversing nodes $v_6$ and $v_3$, we can detect a cycle because we reach node $v_1$ and $\mu(v_1) = 1 < \mu(v_3) = 10$ as shown in Fig. 2(d). Similarly, while backtracking to node $v_2$, two more cycles are detected, and $\lambda(v_3)$, $\lambda(v_6)$, $\lambda(v_5)$, and $\lambda(v_4)$ are respectively updated from 10, 9, 4, and 3 to 1 as shown in Fig. 2(e). $\lambda(v_4) = 1$ propagates to the parent node of $v_4$, *i.e.*, $v_2$, and eventually, we can obtain the rooted depth-first tree shown in Fig. 2(f).

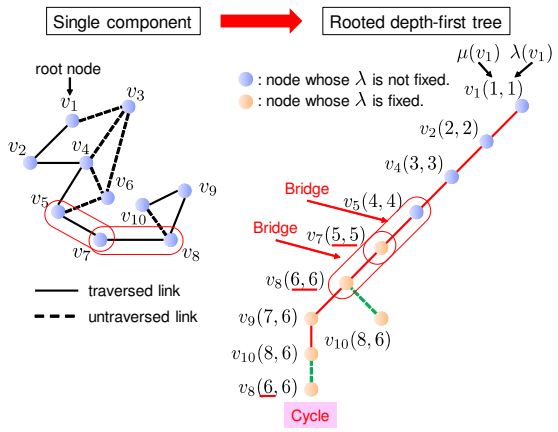
Further, let $\eta(v)$ be the induced connected component from descendant nodes of $v$ in the rooted depth-first tree obtained by the depth-first search. Also, let $\zeta(v)$ be the connected component containing node $v$. Here, we denote the numbers of nodes in $\eta(v)$ and $\zeta(v)$ by $|\eta(v)|$ and $|\zeta(v)|$, respectively. Then, if a bridge $e = \{v,w\}$ is detected and $\zeta(w)$ and $\eta(w)$ are DR and DU components, respectively, in the rooted depth-first tree that is resulted from a sampled graph $G_h$ for $h \in \mathcal{H}$, we can add $|\eta(w)|$ to the contribution value of $e$, *i.e.*, $F_H(e)$, because no node in $\eta(w)$ can reach any node $u \in \mathcal{U}$ if $e$ is removed from $G_h$. For example, suppose the sampled graph $G_h$ shown in the right in Fig. 3(a) is given in which $v_1$, $v_7$, and $v_{11}$ are the prespecified target nodes and there exist three disjoint components. Then, by traversing it from node $v_1$ in the depth-first manner, we can obtain the rooted depth-first tree shown in the middle in Fig. 3(b) and detect two bridges $\{v_5, v_7\}$ and $\{v_6, v_5\}$. Removing each bridge from $G_h$ yields two disjoint components as shown in the right in Fig. 3(b), but we cannot update the contribution value of these bridge links because both of the resulting components contain one of the target nodes, *i.e.*, all of them are a DR component. Similarly, we can obtain the rooted depth-first tree shown in the middle in Fig. 3(c) and detect two bridges $\{v_{13}, v_{12}\}$ and $\{v_{11}, v_{13}\}$ by traversing $G_h$ from node $v_{11}$ in the depth-first manner. In this case, since by removing each of them $\eta(v_{12})$ and $\eta(v_{13})$ become a DU component as shown in the right in Fig. 3(c), we can add $|\eta(v_{12})| = 1$ to $F_H(\{v_{13}, v_{12}\})$ and $|\eta(v_{13})| = 2$ to $F_H(\{v_{11}, v_{13}\})$ according to each case.



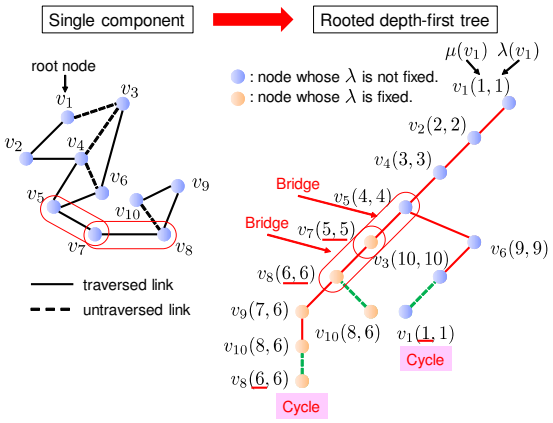

(a) The rooted depth-first tree after the 2nd visit of $v_8$



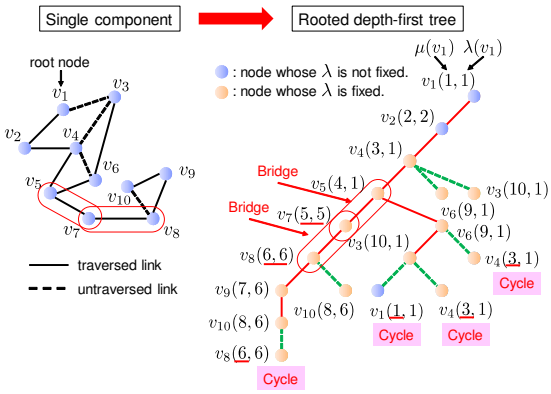

(b) The rooted depth-first tree after $\lambda(v_8)$ was fixed



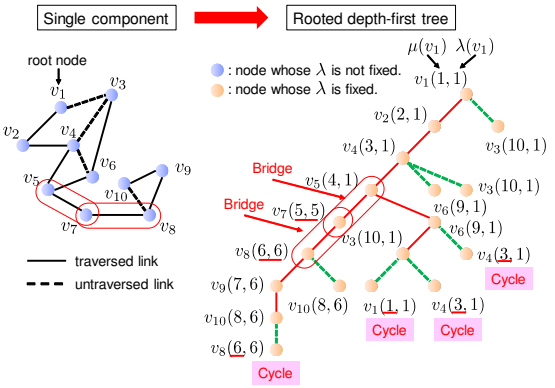

(c) The rooted depth-first tree after $\lambda(v_7)$ was fixed




(d) The rooted depth-first tree after the 2nd visit of $v_1$

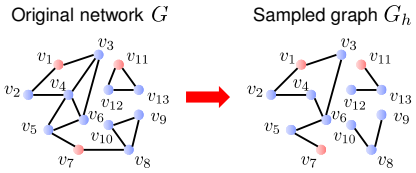



(e) The rooted depth-first tree after $\lambda(v_4)$ was fixed

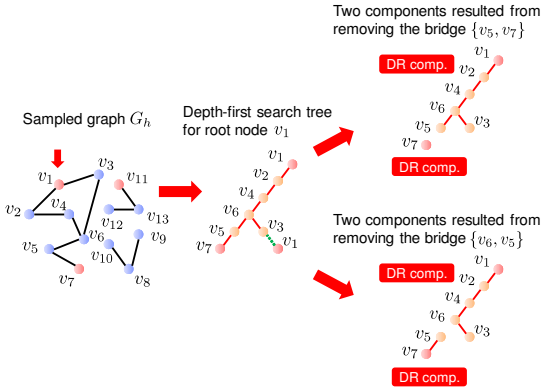



(f) The resulting rooted depth-first tree

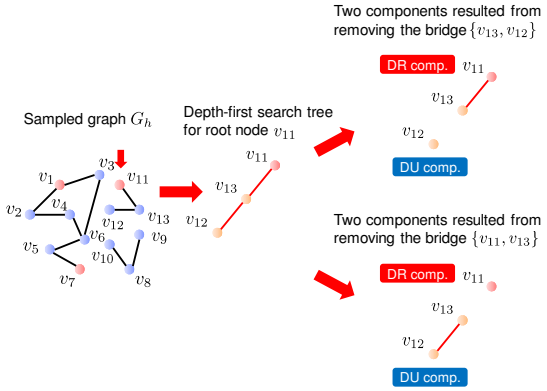
Fig. 2. An example of constructing a rooted depth-first tree from a given single component.



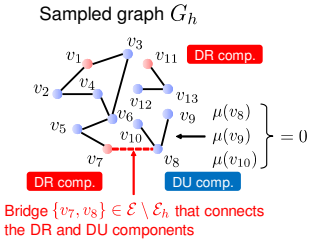

(a) A sampled graph $G_h$ of the original network $G$




(b) The rooted depth-first tree resulted from traversing the largest component from node $v_1$ and derived components by removing each of the two detected bridges $\{v_5, v_7\}$ and $\{v_6, v_5\}$




(c) The rooted depth-first tree resulted from traversing one of the rest components from node $v_{11}$ and derived components by removing each of the two detected bridges $\{v_{13}, v_{12}\}$ and $\{v_{11}, v_{13}\}$




(d) The bridge $\{v_7, v_8\} \in \mathcal{E} \setminus \mathcal{E}_h$ that connects two disjoint components of $G_h$ which are the DR component and the DU component, respectively.

Fig. 3. An illustration to explain how bridges can be used to compute the reachability contribution value $\phi_h(e)$

While the above update of the reachability contribution value is based on the fact 1) mentioned at the beginning of this section, based on the aforementioned fact 2), if a link $e = \{v, w\} \in \mathcal{E} \setminus \mathcal{E}_h$ is a bridge that connects two disjoint components $\zeta(v)$ and $\zeta(w)$ in the sampled graph $G_h$ and only either one of them is a DU component, then we can add $|\zeta(v)|$ to $F_H(e)$ if $\zeta(v)$ is a DU component, and vice versa. For example, consider the sampled graph $G_h$ in Fig. 3 again. After traversing it starting from nodes $v_1$ and $v_{11}$, we can detect the last component that is a DU component by traversing $G_h$ from one of the rest nodes, $v_8$, $v_9$, and $v_{10}$. Then, we can find that link $\{v_7, v_8\} \in \mathcal{E} \setminus \mathcal{E}_h$ is a bridge that connects two disjoint components $\zeta(v_7)$ and $\zeta(v_8)$ as shown in Fig. 3(d). Since $\zeta(v_7)$ is a DR component and $\zeta(v_8)$ is a DU component, we can add $|\zeta(v_8)| = 3$ to $F_H(\{v_7, v_8\})$, the contribution value of the bridge link $\{v_7, v_8\}$.

We summarize our proposed algorithm as Algorithm 1. Note that $\mu(v) = -1$ at Step 9 means that $v$ is not traversed during the depth-first search at Step 4 to compute DR components and that $v$ is not

**Algorithm 1** Proposed algorithm

**Require:** A set of $H$ graph samples, $\mathcal{G}_H = \{G_h = (\mathcal{V}, \mathcal{E}_h) \mid h \in \mathcal{H}\}$.
**Ensure:** The set of reachability contribution values for $\mathcal{G}_H$, $\{F_H(e) \mid e \in \mathcal{E}\}$.
1: Initialize contribution value $F_H(e)$ as $F_H(e) \leftarrow 0$ for every $e \in \mathcal{E}$.
2: **for all** $h \in \mathcal{H}$ **do**
3: Initialize the original number as $\mu(v) \leftarrow 0$ for each node.
4: Compute DR components by repeatedly doing the depth-first search from a node $u \in \mathcal{U}$ s.t. $\mu(u) = -1$ after setting $Numb \leftarrow 1$, $\mu(u) \leftarrow Numb$, $\lambda(u) \leftarrow Numb$, and $v \leftarrow u$.
5: **for all** link $e = \{v, w\}$ obtained during the depth-first search **do**
6: Compute $\mu(w)$, $\lambda(w)$ and $\eta(w)$ by the depth-first search.
7: Update $F_H(e)$ as $F_H(e) \leftarrow F_H(e) + |\eta(w)|$ if the link $e$ is a bridge, *i.e.*, $\mu(w) = \lambda(w)$, and $\eta(w)$ is a DU component.
8: **end for**
9: Compute DU components by repeatedly doing the depth-first search starting from nodes $v \in \mathcal{V}$ s.t. $\mu(v) = -1$ and after setting $Numb \leftarrow 1$, $\mu(v) \leftarrow Numb$, and compute $\mu(v)$ for $v$ in DU components.
10: **for all** link $e = \{v, w\} \in \mathcal{E} \setminus \mathcal{E}_h$ **do**
11: Update $F_H(e)$ as $F_H(e) \leftarrow F_H(e) + |\zeta(w)|$ if $\zeta(v)$ and $\zeta(w)$ are DR and DU components, in this order.
12: Update $F_H(e)$ as $F_H(e) \leftarrow F_H(e) + |\zeta(v)|$ if $\zeta(v)$ and $\zeta(w)$ are DU and DR components, in this order.
13: **end for**
14: **end for**
15: **return** $\{F_H(e) \mid e \in \mathcal{E}\}$ after setting $F_H(e) \leftarrow F_H(e)/H$ for every $e \in \mathcal{E}$.

reachable from any node $u \in \mathcal{U}$. Thus, we can compute a DU component by finding the whole set of reachable nodes from an arbitrary node $v$ satisfying $\mu(v) = -1$ through the depth-first search at Step 9. Evidently, for a given $h \in \mathcal{H}$, the computational complexity of the steps 3 to 13 is $O(|\mathcal{E}|)$ which is the same as the standard bridge detection algorithm. Thus, the total computational complexity of our algorithm becomes $O(H \times |\mathcal{E}|)$. Hereafter, we refer to our proposed algorithm described above as the $TCC$ method.

## 5. Experiments

We evaluated the effectiveness of the proposed method and the generality of the results using two real networks.

### *5.1. Experimental Settings*

The networks we used are from two different domains, one from social and the other from spatial domains. The former is Enron network derived from the Enron Email Dataset [43]. Each node represents a distinct email address and two nodes are connected if and only if there are bidirectional communications. The resulting network has $4,254$ nodes and $22,157$ links. The latter is Road network of Washington

D.C., which is constructed from OSM (OpenStreetMap) data obtained from Metro Extracts [1] in August, 2015 [7]. The resulting network has $114,758$ nodes and $128,746$ links. The average degree of a node, *i.e.*, the average number of links incident to a node, is 10.42 for Enron network, and 2.2 for Road network, while the maximum degree is 384 for Enron network, and 8 for Road network. Enron network is much denser than Road network as can be expected. We decided to set $p_e = p$ for every $e \in \mathcal{E}$, since the main purpose of this work is to evaluate the fundamental performance of the proposed method. Thus, $p$ is the unique parameter for controlling the disconnection probability. Here we should emphasize that this uniform setting has been widely employed in many previous studies in the filed of information diffusion over social networks [1, 2].

We compare the proposed $TCC$ measure with the extended version of the conventional link betweenness centrality measure by taking into account the target nodes, $\mathcal{U}$, which is referred to as *TBC* (Target-oriented link Betweenness Centrality). Namely, *TBC* is defined by

$$TBC(e;\mathcal{G}_H) = \frac{1}{H}\sum_{h=1}^{H} tlbc(e;G_h)$$

for a link $e \in \mathcal{E}$, where $tlbc(e)$ is provided by

$$tlbc(e) = \sum_{x\in\mathcal{V}} \frac{1}{|\mathcal{U}(x)|} \sum_{u\in\mathcal{U}(x)} \frac{N_L^{sp}(x,u;e)}{N^{sp}(x,u)}.$$

Here, $\mathcal{U}(x)$ indicates the set of nodes $u \in \mathcal{U}$ such that $u$ is the closest node from node $x$ in $\mathcal{U}$ in terms of a distance function $d$ over network $G$, $N^{sp}(x,u)$ denotes the number of the shortest paths from node $x$ to node $u$, and $N_L^{sp}(x,u;e)$ denotes the number of those paths passing through a link $e$. In addition, we also compare the $TCC$ measure with a variant without considering target nodes, which is referred to as *CC* (link Criticalness Centrality) given by the following equation:

$$CC_H(e) = \frac{1}{H}\sum_{h\in\mathcal{H}}\Big(\sum_{v\in\mathcal{V}} |\mathcal{R}(v;G_h^+(e))| - \sum_{v\in\mathcal{V}} |\mathcal{R}(v;G_h^-(e))|\Big). \tag{7}$$

Here note that *CC* can be regarded as the link centrality version of the node centrality proposed by Ohara *et al.* [11] called *LCC* (Latent node Criticalness Centrality).

Assigning target nodes needs problem specific information, *e.g.*, location of emergency hospitals in case of ambulance call for a road network. In our experiments we artificially assigned target nodes by use of notion of medoid and changed the number by specifying the rate $q$ to the total number of nodes $|\mathcal{V}|$. Here, medoid is a center node in a cluster. In our problem setting it is a node that is closest to the rest of the nodes, *i.e.*, the sum of the shortest path to the rest of the nodes is minimum. For multiple number of medoid, each node has its own medoid to which the shortest path is minimum. To be more precise, the number of target nodes is set to the nearest integer of $q \times |\mathcal{V}|$. We also assigned target nodes randomly with the same parameter $q$ for comparison purpose. We employed a greedy algorithm to assign a set of medoid target nodes, *i.e.*, the best node in terms of the medoid criterion is repeatedly selected by fixing the already assigned nodes until the number of selected nodes reaches the number of target nodes.

[1] https://mapzen.com/data/metro-extracts

We notice that the medoid criterion satisfies the submodular property, *i.e.*, marginal loss of sum of the shortest path from each node to its own medoid over all nodes monotonically decrease as the number of target nodes increases, thus the minimum performance of the greedy algorithm is guaranteed. In our experiments, we set the rate $q$ as $q \in \{0.002.0.05\}$ and denote the experimental results with medoid and random target nodes as m002, r002, m05, and r05, respectively.

### 5.2. Experimental results

First, we evaluated the computational efficiency of the proposed $TCC$ method by comparing it with the straightforward extended versions of conventional node centrality measures, *i.e.*, degree, and PageRank centralities, and a conventional link centrality measure, *i.e.*, link betweenness centrality, in our uncertain network framework. First, we extend degree centrality and define node centrality measure $DC$ by

$$DC(v;\mathcal{G}_H) = \frac{1}{H}\sum_{h=1}^{H} deg(v;G_h)$$

for a node $v \in \mathcal{V}$, where $deg(v;G_h)$ indicates the degree of a node $v$ in $G_h$. Second, we define node centrality measure $PRC$ from PageRank centrality by

$$PRC(v;\mathcal{G}_H) = \frac{1}{H}\sum_{h=1}^{H} pgrk(v;G_h)$$

for a node $v \in \mathcal{V}$, where $pgrk(v;G_h)$ is the PageRank score of a node $v$ in $G_h$ and is given by the PageRank algorithm with random jump factor $0.15$. Third, we extend betweenness centrality and define link centrality measure $BC$ by

$$BC(e;\mathcal{G}_H) = \frac{1}{H}\sum_{h=1}^{H} lbc(e;G_h)$$

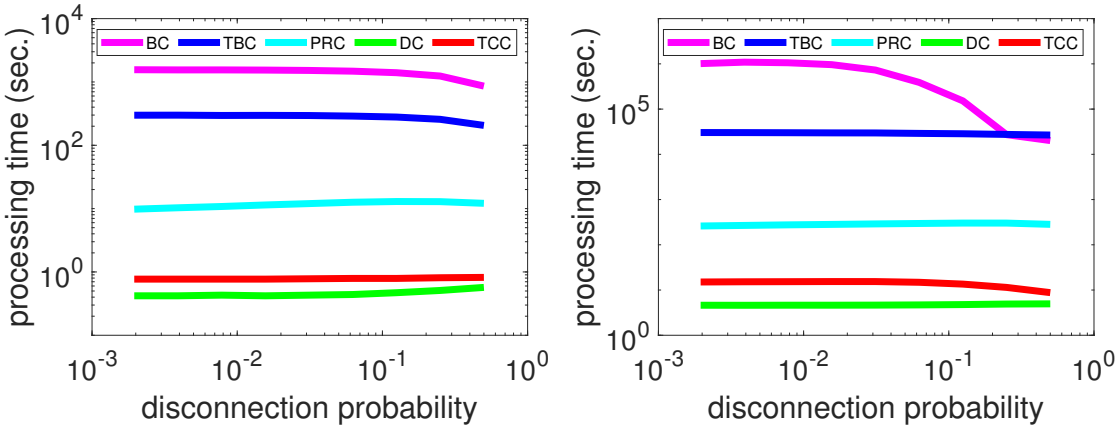
for a link $e \in \mathcal{E}$, where $lbc(e;G_h)$ is the betweenness of a link $e$ in $G_h$ and is given by

$$lbc(e) = \sum_{x\in\mathcal{V}}\sum_{y\in\mathcal{V}} \frac{N_L^{sp}(x,y;e)}{N^{sp}(x,y)}.$$

Here, $N^{sp}(x,y)$ and $N_L^{sp}(x,y;e)$ also denote the number of the shortest paths from node $x$ to node $y$ and the number of those paths passing through a link $e$, respectively. The former two are node centralities and not link centralities, but the purpose is to compare the computational efficiency and no precise definition is needed here.

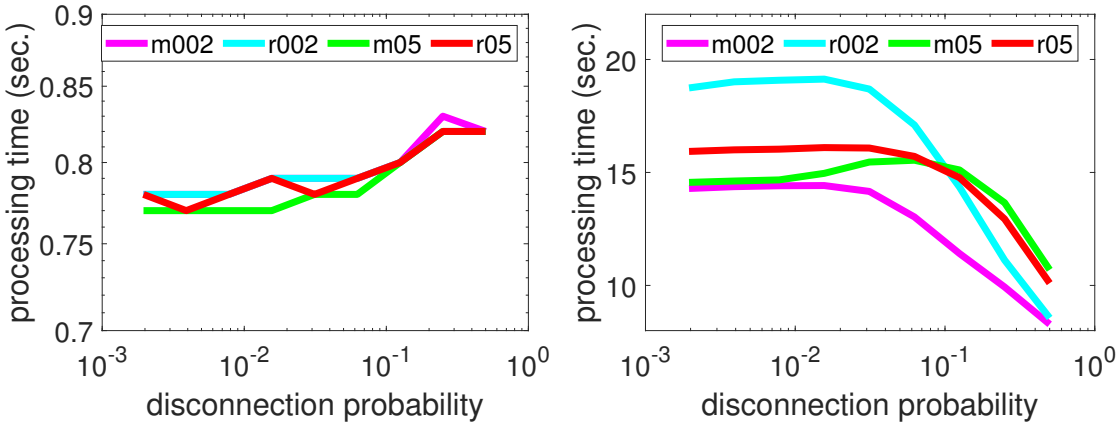
We compared the processing time of the proposed $TCC$ with those of $BC$, $PRC$ and $DC$ to compute the centralities for $\mathcal{G}_H$ with a setting $H = 10^3$. We set the disconnection probability $p_e = p$ to $2^{-k}$ for every link $e \in \mathcal{E}$ and varied the integer $k$ from 1 to 9, leading to the range of $p$, $(0.0019, 0.5]$. Figure 4 shows the experimental results [2], where $TBC$ and $TCC$ are computed for r002. Figures 4(a) and 4(b)

[2]Our programs are written in C, and run on a computer with Xeon X5690 3.47GHz CPUs using a single thread within a 192GB main memory capacity.




(a) Enron (b) Road

Fig. 4. Comparison of processing time with the representative centrality measures as a function of the link disconnection probability. $TCC$ and $TBC$ are for r002.

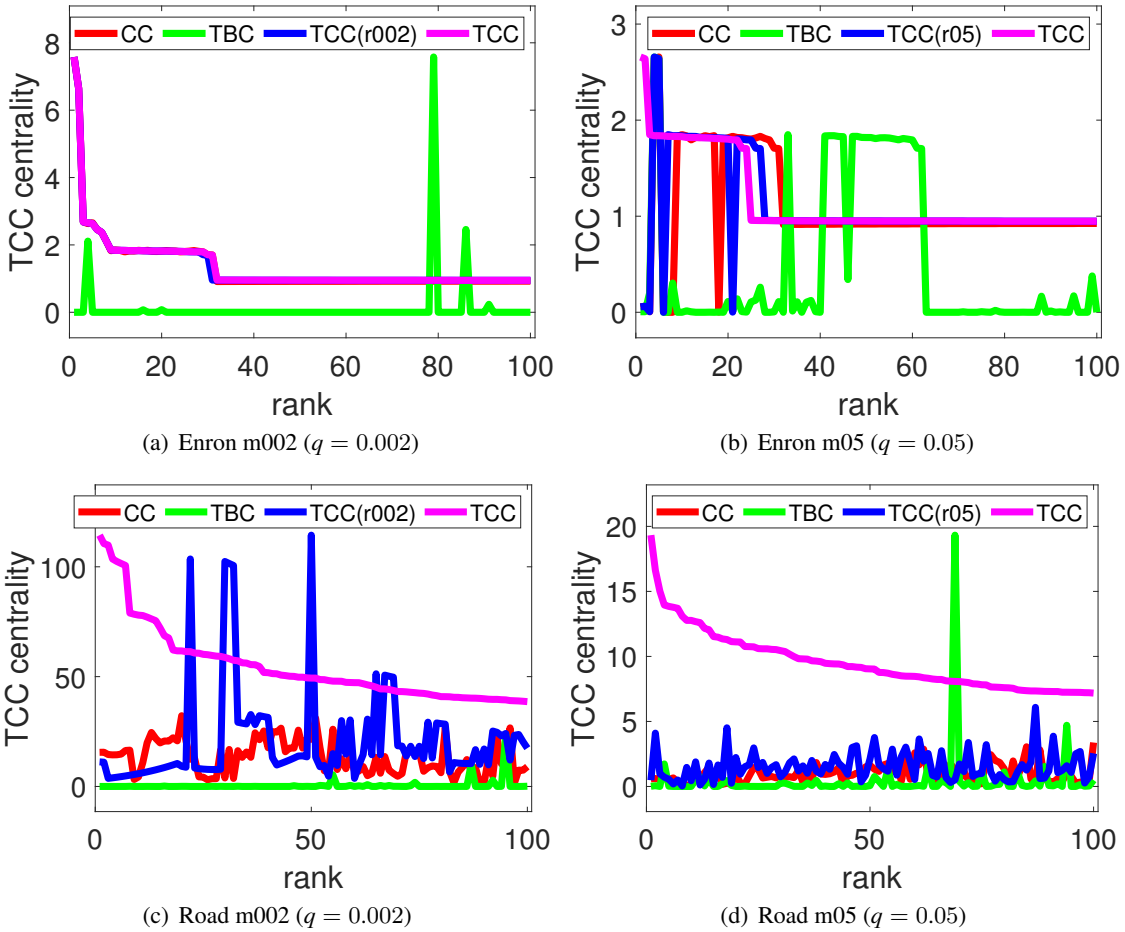



(a) Enron (b) Road

Fig. 5. Comparison of processing time with different target node settings as a function of the link disconnection probability.

are the results for the Enron and Road networks, respectively. From Fig. 4, it can be observed that these two networks with different configuration and size exhibit quite similar tendency. The processing times for computing $TCC$ are substantially smaller than those for $BC$, $TBC$ and $PRC$ although they are roughly twice as large as those for $DC$, most efficiently computable one. $BC$ is most expensive because its computational complexity approximately becomes a square order of the network size. $TBC$ becomes substantially faster than $BC$ due to the restricted target nodes $\mathcal{U}$. $PRC$ needs power-iterations for solving the eigenvector equation. It is noted that the processing times for $BC$ are likely to decrease when the disconnection probability $p$ becomes large, especially for the Road network. This is because the number of disconnected node pairs increases for a large $p$. From these results, we can claim that our proposed $TCC$ method has a good scalable property with respect to the size and the configuration of networks.

Next, we evaluated the computational efficiency of the proposed $TCC$ method by changing target nodes settings. Figure 5 shows the experimental results, where we show the results of the settings of



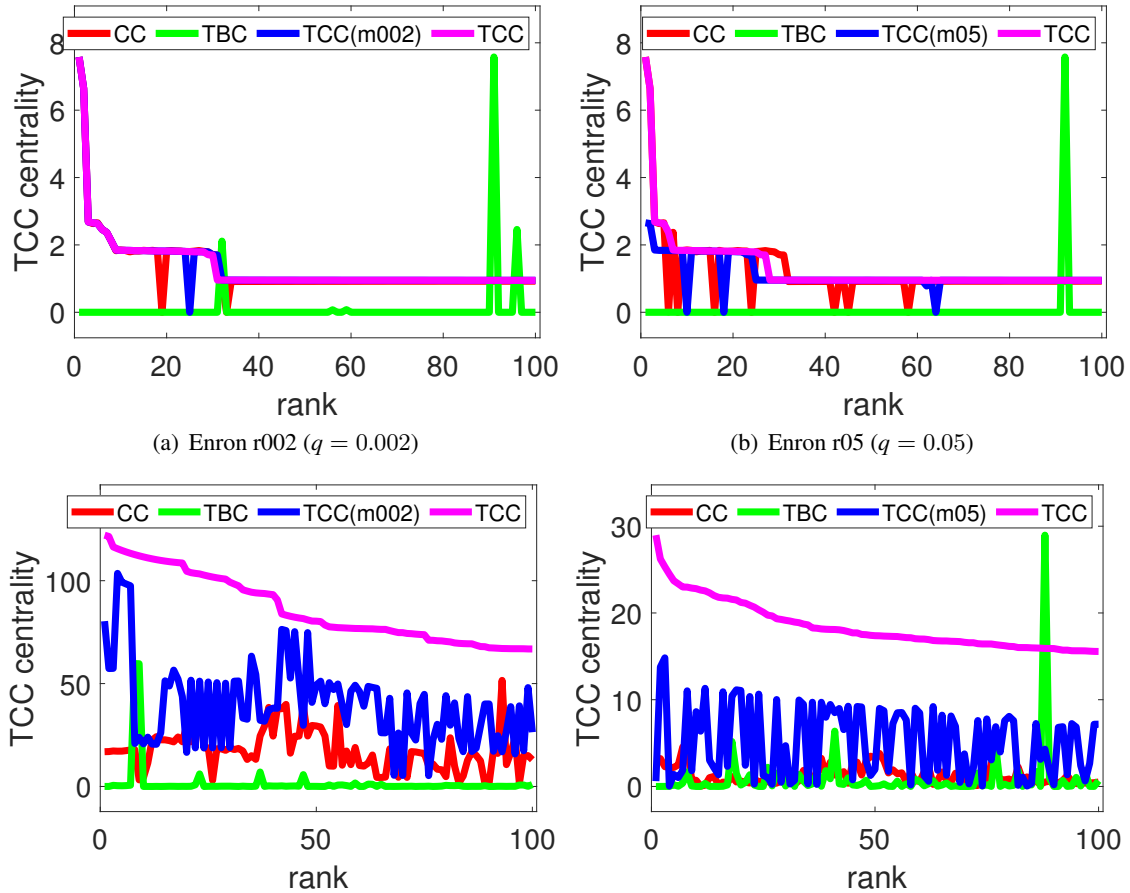

(a) Enron m002 ($q = 0.002$) (b) Enron m05 ($q = 0.05$)

(c) Road m002 ($q = 0.002$) (d) Road m05 ($q = 0.05$)

Fig. 6. $TCC$ value of the top-100 links obtained by each centrality measure for two different medoid target rates ($q = 0.002$ and $0.05$) under a link disconnection probability ($p = 2^{-4}$).

m002, r002, m05, and r05. Figures 5(a) and 5(b) are the results for the Enron and Road networks, as before. From Fig. 5, it can be observed that the processing times of the proposed $TCC$ method are quite comparable for different target-nodes settings, especially for the Enron network. This indicates that $TCC$ method is computationally efficient regardless of the target-nodes settings. Note that in case of the Road network, as shown in Fig. 5(b), the processing times are likely to decrease when the disconnection probability $p$ becomes large because of the same reason above (the number of disconnected node pairs increases for a large $p$), which is similar to the experimental result shown in Fig. 4(b).

We further examined the performances of highly ranked links that are identified by the other link centralities, *i.e.*, $CC$ and $TBC$, in case of the disconnection probability $p = 2^{-4}$ for the two networks. $TCC$ values were computed for both medoid and random target settings of the same size. More specifically, let $S$ be one of {m002, r002, m05, r05}, and $L_H$ be $TCC(S)$ value, *i.e.*, $L_H = TCC(S)$. We then evaluated the performance defined by $L_H(e_i^{(C)})$ for each $C$, where $C$ denotes one of $\{CC, TBC, TCC(S)\}$ and $e_i^{(C)}$ stands for the link $e$ with the $i$-th rank in each centrality measure $C$. The results for the other disconnection probabilities are quite similar. Thus, we report the results for $p = 2^{-4}$.



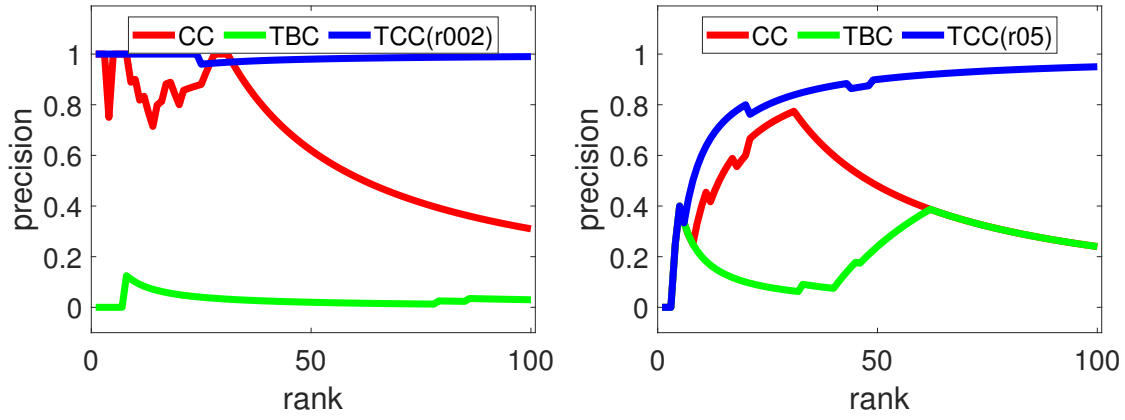

(a) Enron r002 ($q = 0.002$) (b) Enron r05 ($q = 0.05$)

(c) Road r002 ($q = 0.002$) (d) Road r05 ($q = 0.05$)

Fig. 7. $TCC$ value of the top-100 links identified by each centrality measure for two different random target rates ($q = 0.002$ and $0.05$) under a link disconnection probability ($p = 2^{-4}$).

Figure 6 shows the experimental results for the medoid target node setting, where the horizontal and vertical axes denote the ranking of links up to top-100 and the performance evaluated in the $TCC(S)$ measure, respectively. The pairs of Figs. 6(a) and 6(b), and Figs. 6(c) and 6(d) are the results for the Enron and Road networks, respectively. Note that $CC$ does not need target node information. $CC$ and $TCC(\mathrm{r002})$ are hidden underneath $TCC$ almost completely in Fig. 6(a), and $CC$ and $TCC(\mathrm{r05})$ partly in Fig. 6(b). Figure 7 is the results for the random target node settings, where the pairs of Figs. 7(a) and 7(b), and Figs. 7(c) and 7(d) are the results for the Enron and Road networks, respectively. Like in the medoid target node setting, $CC$ and $TCC(\mathrm{m002})$ are hidden mostly underneath $TCC$ in Fig. 7(a), and $CC$ and $TCC(\mathrm{m05})$ partly in Fig. 7(b).

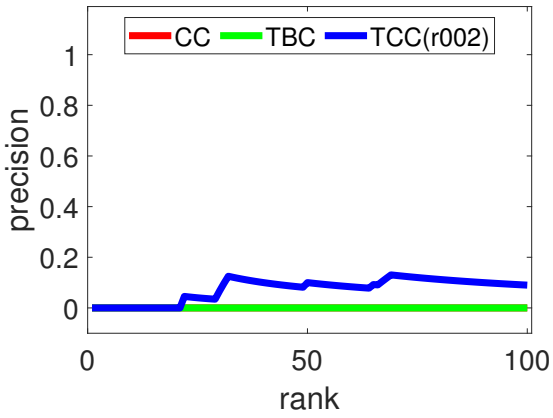
Following observation can be made from these results. Performance behaviors and the degree of their differences are different for different networks and for different target node settings as well as for different link centralities. The difference between $TCC$ and others is much clearer for Road network than Enron network in both target node settings. No other centralities can replace the proposed $TCC$ and the location of the target nodes does matter. To our surprise $TBC$ is not a good measure. We conjecture that



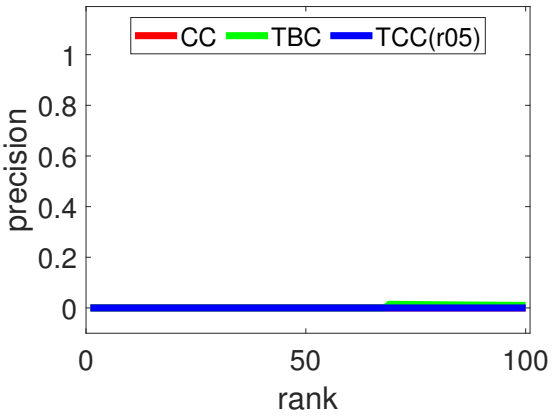

(a) Enron m002 ($q = 0.002$)




(b) Enron m05 ($q = 0.05$)




(c) Road m002 ($q = 0.002$)




(d) Road m05 ($q = 0.05$)

Fig. 8. Precision comparison in the top-100 links obtained by each centrality measure for different medoid target rates ($q = 0.002$ and $0.05$) under a link disconnection probability ($p = 2^{-4}$).

the links that appear on the shortest paths do not happen to be the links needed to maintain connectivity on these networks. In Enron network the location of the target nodes does not seem to matter when the size is very small. However, later we show the precision of the top ranked links and see that overlaps of the curves in Figs. 6(a), 6(b), 7(a) and 7(b) do not necessarily mean that the performance is similar, *i.e.*, the locations of critical links are the same. Putting all the observations together, we can conclude that the performances of highly ranked links strongly depend on the networks and the target node settings, and less strongly on the target node size for some network. In short, $TCC$ measure with a given target node setting has unique properties and no other centrality measure can replace the proposed $TCC$ measure. It is difficult to identify links having high $TCC$ values by means of conventional centrality measures.

Next, in order to more closely examine our experimental results, we further investigated the similarity between the ranking results obtained by $TCC(S)$ measure and those by $CC$, $TBC$ and $TCC(S')$ measures in case of the disconnection probability $p = 2^{-4}$ for the two networks. These correspond to Figs 6 and 7. Again, it should be noted that we obtained quite similar experimental results for the other disconnection probabilities. We measured the similarity between the top $k$ links for the ranking based

on $TCC(S)$, denoted by $\mathcal{A}_k^{(TCC(S))} = \{e_i^{(TCC(S))}\}_{i=1}^k$, and those for the ranking based on the other link centralities, $CC$, $TBC$, and $TCC(S')$, denoted by $\mathcal{A}_k^{(C')} = \{e_i^{(C')}\}_{i=1}^k$, by using the precision $Prec(k)$ defined as follows:

$$Prec(k) = \frac{|\mathcal{A}_k^{(TLC(S))} \cap \mathcal{A}_k^{(C')}|}{k}.$$

where $C'$ and $S'$ stand for one of $\{CC, TBC, TCC(S')\}$, and one of { m002, r002, m05, r05 } \ $\{S\}$, respectively.

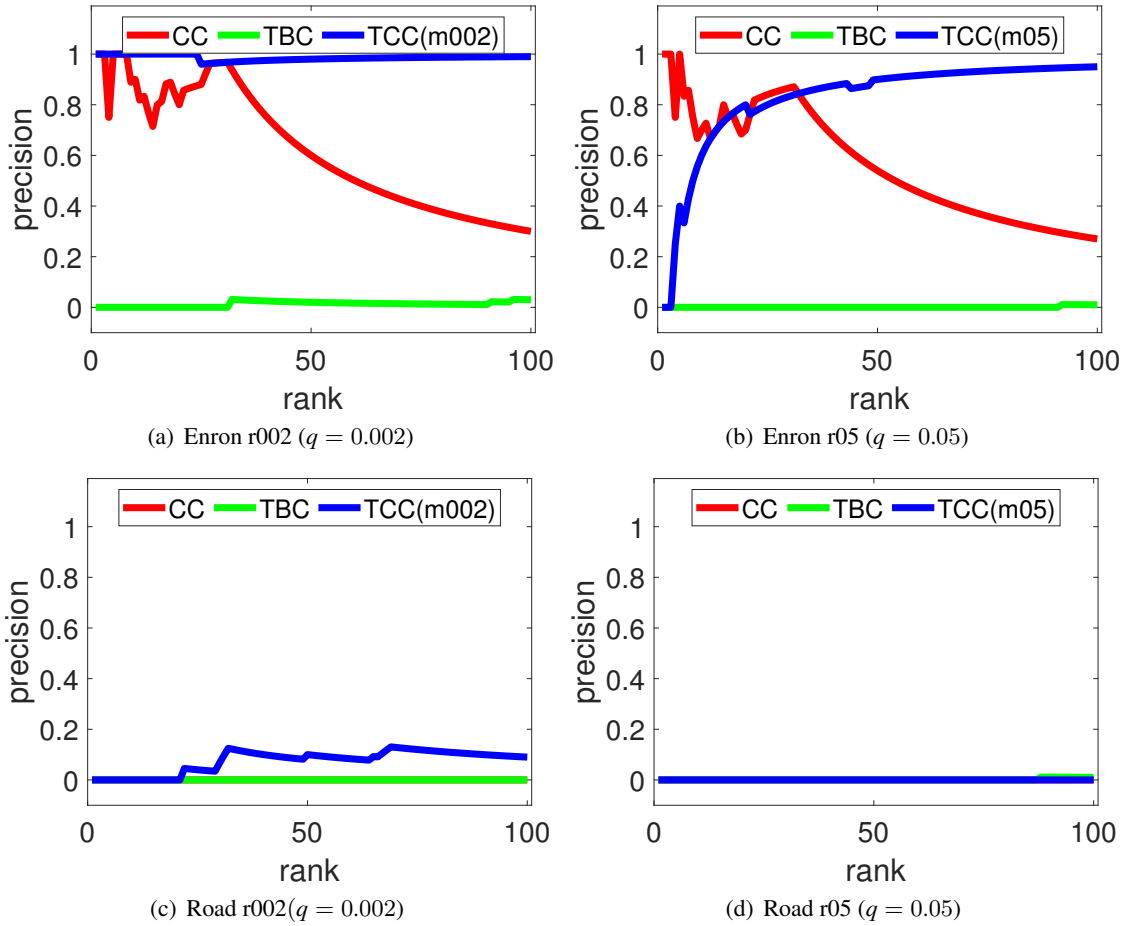
Figure 8 shows the experimental results for the medoid target node settings, where the horizontal and vertical axes denote the rank $k$ up to top-100 and the precision $Prec(k)$, respectively. Again, the pairs of Figs. 8(a) and 8(b), and Figs. 8(c) and 8(d) are the results for the Enron and Road networks, respectively. Figure 9 shows the results for the random target node settings in the same format. As before, some curves are hidden because of the overlaps with the other curves. $CC$ in Figs. 8(c) and 9(c) are hidden beneath $TBC$.

From these figures, we can also see that the experimental results basically coincide with those shown in Figs. 6 and 7. $TBC$ is the worst among all. One notable difference is that the precision scores of $CC$ are substantially poor in comparison to those of $TCC$(r002) in Fig. 8(a), $TCC$(r05) in Fig. 8(b), $TCC$(m002) in Fig. 9(a) and $TCC$(m05) in Fig. 9(b), especially for the ranks from top-50 to top-100. This is because the slight difference of $TCC$ values by $CC$ method results in different links as critical ones. In short, our results confirm that $TBC$ measure is not useful at all for finding links having high $TCC$ values. $CC$ may be helpful to some degree but it heavily depends on network configuration. When the size of the target nodes is small, $TCC$ does not require correct locations, but this again heavily depends on network configuration.

Finally, we evaluated the approximation accuracy by changing $H$. More specifically, we computed the $TCC$ value, $L_H(e)$ , for each link $e$ by setting $H = 1,000,000$, and regarded them as the ground truth by setting $L^*(e) \leftarrow L_H(e)$. Then, we repeated $L_H(e)$ 100 times for a fixed $H$,*i.e.*, $\{L_H^{(i)}(e) \mid 1 \leqslant i \leqslant 100\}$, each of which is obtained by a series of independent Bernoulli trials for $H \in \{10^2, 10^3, 10^4\}$, and evaluated them in terms of the relative error $RE_H(e)$ defined by

$$RE_H(e) = \frac{1}{100} \sum_{i=1}^{100} \left| \frac{L_H^{(i)}(e) - L^*(e)}{L^*(e)} \right|. \tag{8}$$

Figure 10 shows the experimental results of the top-1 link according to $L^*(e)$ for each probability setting of $p \in \{2^{-1}, \cdots, 2^{-9}\}$ for the medoid target node settings. Again, the pairs of Figs. 10(a) and 10(b), and Figs. 10(c) and 10(d) are the results for the Enron and Road networks, respectively. Figure 11 shows the results for the random target node settings in the same format. From these figures, we can see that the experimental results for the case of $q = 0.002$ are relatively more stable than those of $q = 0.05$, where the numbers of target nodes for the former settings are substantially smaller than those for the latter settings. Moreover, we can see that the relative errors are less than 10% for any disconnection probability $p$ even for $H = 10^2$ for Enron network and for Road network with the setting m05, and $H = 10^3$ for the Road network with the setting r002. In short, these experimental results suggest that we can stably compute our $TCC$ centrality without sampling very many graphs.




(a) Enron r002 ($q = 0.002$) (b) Enron r05 ($q = 0.05$)

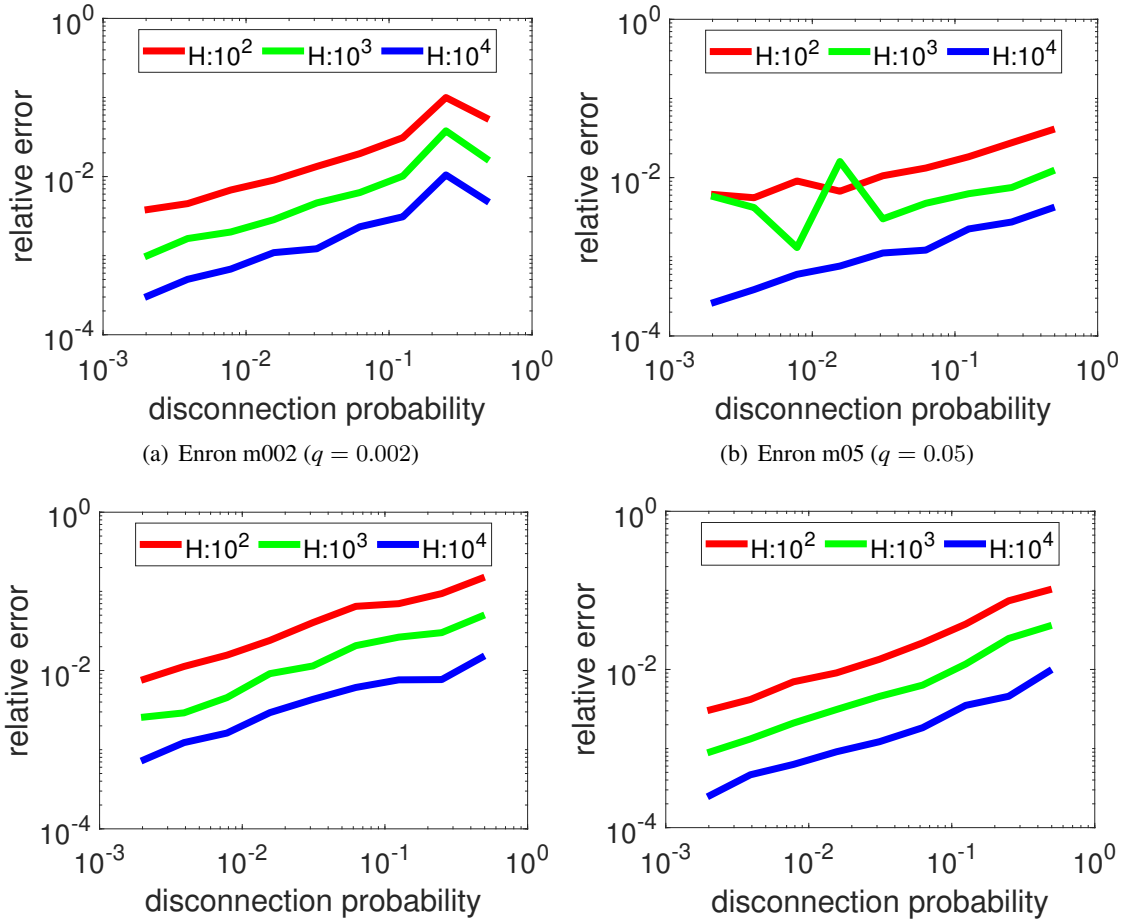
(c) Road r002($q = 0.002$) (d) Road r05 ($q = 0.05$)

Fig. 9. Precision comparison in the top-100 links obtained by each centrality measure for different random target rates ($q = 0.002$ and $0.05$) under a link disconnection probability ($p = 2^{-4}$).

## 6. Conclusion

We addressed the problem of efficiently identifying critical links under a realistic situation where all the links in a network are probabilistically disconnected as is assumed in studies in uncertain graphs. A link is critical if it has a large performance degradation when it is removed. There are many situations in which disconnection can take place probabilistically, *e.g.*, some information paths between two persons may not be open by server down in case of a social network, and some roads between two intersections may be blocked due to traffic accident in case of a road network.

Noting that bridge in graph theory is the key concept to solve this problem, we proposed a novel algorithm that can efficiently identify critical links by incorporating the bridge detection algorithm as the underlying search technique. A bridge is a link in a connected component such that its deletion divides the component into two disjoint ones. To be more precise, we define a new measure, *Target-oriented latent link Criticalness Centrality (TCC)*, which quantifies the performance degradation and is defined in this paper as the marginal loss of the expected number of nodes that can reach, and equivalently can be



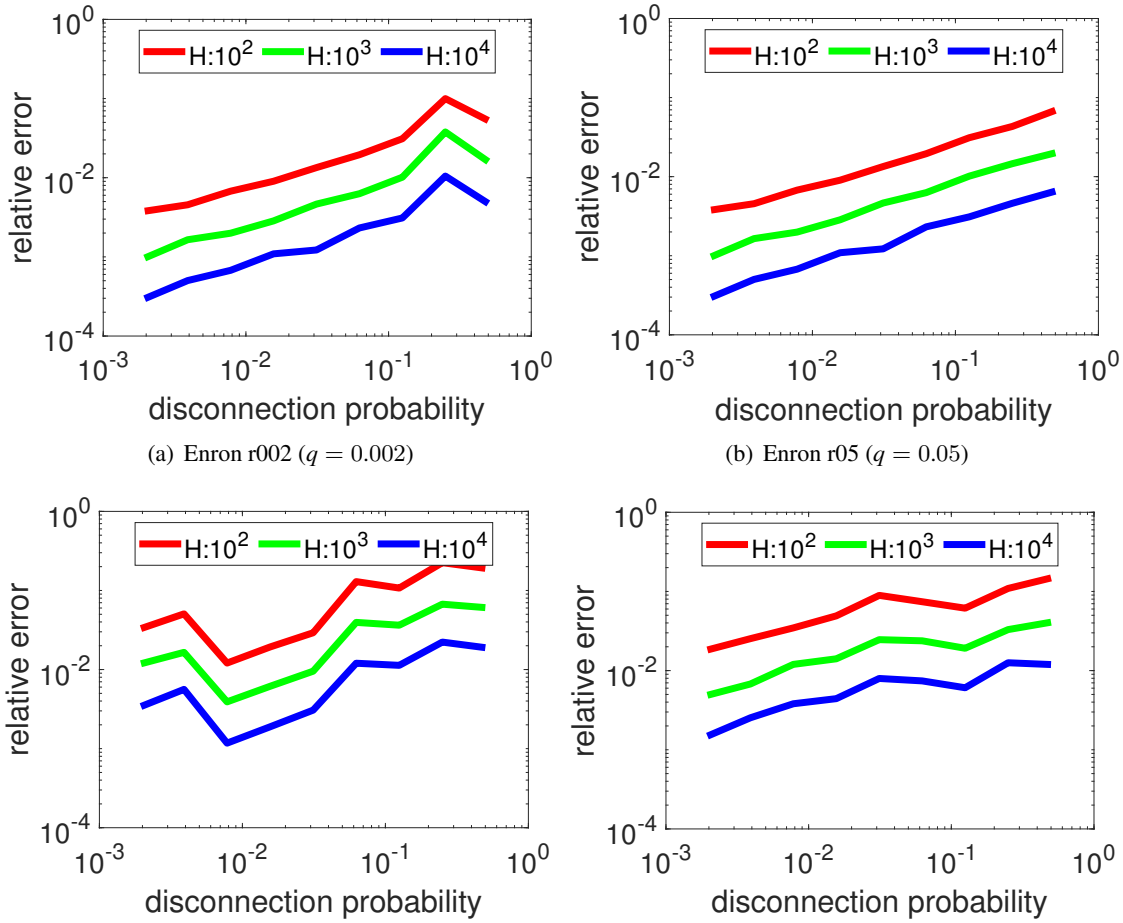

(a) Enron m002 ($q = 0.002$) (b) Enron m05 ($q = 0.05$)

(c) Road m002 ($q = 0.002$) (d) Road m05 ($q = 0.05$)

Fig. 10. Fluctuation of the relative error of $TCC$ values as a function of the link disconnection probability for different number of simulations for two different medoid target rates ($q = 0.002$ and $0.05$).

reached from, one of the target nodes in the network, and compute $TCC$ for each link by use of detected bridges.

We applied the proposed method to two real-world networks, one from social network and the other from spatial network, and investigated how the performance compares with the traditional link centralities with respect to processing time, $TCC$ value, network structure, target node size and allocation. Computational complexity of the proposed algorithm is $O(H \times |\mathcal{E}|)$, where $H$ is the number of networks generated from an original one based on the probabilistic link disconnection model to compute the expectation, and $|\mathcal{E}|$ is the number of links in the network. The experimental results show that our method is much faster than computing the traditional centrality measures, and has a good scalability with respect to the network size. Critical links identified by the proposed method possesses unique properties. The extended version of the traditional link betweenness centrality $TBC$ where one of the pair nodes is constrained to the closest one in the target nodes gives a very poor performance in terms of both $TCC$ measure and computation time and cannot find good critical links. The target nodes and their size do




(a) Enron r002 ($q = 0.002$) (b) Enron r05 ($q = 0.05$)

(c) Road r002 ($q = 0.002$) (d) Road r05 ($q = 0.05$)

Fig. 11. Fluctuation of the relative error of $TCC$ values as a function of the link disconnection probability for different number of simulations for two different random target rates ($q = 0.002$ and $0.05$).

affect the results, but their effect strongly depends on network structure. We conclude that no known link centralities can replace $TCC$.

Our immediate future work is to extend the proposed method to be able to deal with directed networks for which the standard bridge detection algorithm cannot be used. Another direction is to make the model more realistic, *e.g.* by removing the assumption of uniform link disconnection probability by incorporating problem and domain specific knowledge. We used networks in the real world. It would be useful to use artificial networks as well and systematically change the network properties and conduct performance evaluations more systematically to see which factors are crucial in determining the performance.

## Acknowledgments

This work was partly supported by JSPS Grant-in-Aid for Scientific Research (C) (No. 20K11940).

## References


[1] W. Chen, L.V.S. Lakshmanan and C. Castillo, Information and influence propagation in social networks, *Synthesis Lectures on Data Management* **5(4)** (2013), 1–177.

[2] D. Kempe, J. Kleinberg and E. Tardos, Maximizing the spread of influence through a social network, *Theory of Computation* **11** (2015), 105–147.

[3] Y. Shen, N.P. Nguyen, Y. Xuan and M.T. Thai, On the Discovery of Critical Links and Nodes for Assessing Network Vulnerability, *IEEE/ACM Transaction on Networking* **21**(3) (2013), 963–973.

[4] E.L. Oliveira, L.S. Portugal and W.P. Junior, Determining critical links in a road network: vulnerability and congestion indicators, *Procedia - Social and Behavioral Sciences* **162** (2014), 158–167.

[5] K. Saito, M. Kimura, K. Ohara and H. Motoda, Super mediator - A new centrality measure of node importance for information diffusion over social network, *Information Sciences* **329** (2016), 985–1000.

[6] K. Saito, M. Kimura, K. Ohara and H. Motoda, Detecting Critical Links in Complex Network to Maintain Information Flow/Reachability, in: *Proceedings of the 14th Pacific Rim International Conference on Artificial Intelligence (PRICAI2016)*, 2016, pp. 419–432.

[7] K. Ohara, K. Saito, M. Kimura and H. Motoda, Accelerating computation of distance based centrality measures for spatial networks, in: *Proceedings of the 19th International Conference on Discovery Science (DS'16)*, LNCS 9956, 2016, pp. 376–391.

[8] K. Saito, M. Kimura, K. Ohara and H. Motoda, An Accurate and Efficient method to Detect Critical Links to Maintain Information Flow in Network, in: *Proceedings of the 23th International Symposium on Methodologies for Intelligent Systems (ISMIS2017)*, 2017, pp. 116–126.

[9] K. Ohara, K. Saito, M. Kimura and H. Motoda, Maximizing Network Performance based on Group Centrality by Creating Most Effective k-links, in: *Proceedings of the 4th IEEE International Conference on Data Science and Advanced Analytics (DSAA'17)*, 2017, pp. 561–570.

[10] K. Saito, K. Ohara, M. Kimura and H. Motoda, Critical Link Identification based on Bridge Detection for Network with Uncertain Connectivity, in: *Proceedings of The 24th International Symposium on Methodologies of Intelligent Systems (ISMIS2018)*, LNAI 11177, 2018, pp. 89–99.

[11] K. Ohara, K. Saito, M. Kimura and H. Motoda, Critical Node Identification Based on Articulation Point Detection for Network with Uncertain Connectivity, in: *Proceedings of the Sixth International Symposium on Computing and Networking (CANDAR 2018)*, 2018, pp. 146–152.

[12] K. Ohara, K. Saito, M. Kimura and H. Motoda, Critical Node Identification based on Articulation Point Detection for Uncertain Network, *International Journal of Networking and Computing* **9**(2) (2019), 201–216.

[13] L. Freeman, Centrality in social networks: Conceptual clarification, *Social Networks* **1** (1979), 215–239.

[14] S. Brin and L. Page, The anatomy of a large-scale hypertextual web search engine, *Computer Networks and ISDN Systems* **30** (1998), 107–117.

[15] M. Piraveenan, M. Prokopenko and L. Hossein, Percolation Centrality: Quantifying Graph-Theoretic Impact of Nodes during Percolation in Networks, *PLoS ONE* **8**(1) (2013), 1–14.

[16] P. Crucitti, V. Latora and S. Porta, Centrality measures in spatial networks of urban streets, *Physical Review E* **73**(3) (2006), 036125.

[17] T. Opsahl, F. Agneessens and J. Skvoretz, Node centrality in weighted networks: Generalizing degree and shortest paths., *Social Networks* **32**(3) (2010), 245–251.

[18] T.L. Frantz, M. Cataldo and K.M. Carley, Robustness of centrality measures under uncertainty: Examining the role of network topology, *Computational Mathematical Organization Theory* **15**(303) (2009).

[19] J.J. Pfeiffer III and J. Neville, Methods to Determine Node Centrality and Clustering in Graphs with Uncertain Structure, in: *Proceedings of the Fifth International Conference on Weblogs and Social Media*, L.A. Adamic, R. Baeza-Yates and S. Counts, eds, The AAAI Press, 2011.

[20] C. Wang and Z. Lin, An Efficient Approximation of Betweenness Centrality for Uncertain Graphs, *IEEE Access* **7** (2019), 61259–61272.

[21] M. Avella-Medina, F. Parise, M.T. Schaub and S. Segarra, Centrality Measures for Graphons: Accounting for Uncertainty in Networks, *IEEE Transactions on Network Science and Engineering* **7**(1) (2020), 520–537.

[22] M.G. Everett and S.P. Borgatti, Extending centrality, in: *P. J. Carrington, J. Scott and S. Wasserman (Eds.), Models and methods in social network analysis*, 2005, pp. 57–76.

[23] D. Grady, C. Thiemann and D. Brockmann, Robust classification of salient links in complex networks, *Nature Communications* **3**(864) (2012), 1–10.

[24] Y.-P. Fang, N. Pedroni and E. Zio, Comparing Network-Centric and Power Flow Models for the Optimal Allocation of Link Capacities in a Cascade-Resilient Power Transmission Network, *IEEE Systems Journal* **99** (2014), 1–12.

[25] I. Stojmenovic, D. Simplot-Ryl, A. Nayak and Y. Velaj, Toward scalable cut vertex and link detection with applications in wireless ad hoc networks, *IEEE Network* **25**(1) (2011).

[26] V.K. Akram and O. Dagdeviren, Breadth-First Search-Based Single-Phase Algorithms for Bridge Detection in Wireless Sensor Networks, *Sensors* **13**(7) (2013), 8786–8813.
[27] K. Diao, R. Farmani, G. Fu and D. Butler, Toward scalable cut vertex and link detection with applications in wireless ad hoc networks, *Int. J. Optim. Civil Eng.* **5**(3) (2015).
[28] M. Papagelis, Refining Social Graph Connectivity via Shortcut Edge Addition, *ACM Transactions on Knowledge Discovery from Data* **10**(2) (2015).
[29] N. Parotsidis, E. Pitoura and P. Tsaparas, Selecting Shortcuts for a Smaller World, in: *Proceedings of the 2015 SIAM International Conference on Data Mining (SDM 2015)*, 2015, pp. 28–36.
[30] P. Crescenzi, G. D'angelo, L. Severini and Y. Velaj, Greedily Improving Our Own Closeness Centrality in a Network, *ACM Transactions on Knowledge Discovery from Data* **11**(1) (2016).
[31] N. Parotsidis, E. Pitoura and P. Tsaparas, Centrality-Aware Link Recommendations, in: *Proceedings of of the Ninth ACM International Conference on Web Search and Data Mining (WSDM '16)*, 2016, pp. 503–512.
[32] V. Chaoji, S. Ranu, R. Rastogi and R. Bhatt, Recommendations to Boost Content Spread in Social Networks, in: *Proceedings of the 21th International Conference on World Wide Web (WWW '12)*, 2012, pp. 529–538.
[33] H. Tong, B.A. Prakash, T. Eliassi-Rad, M. Faloutsos and C. Faloutsos, Gelling, and Melting, Large Graphs by Edge Manipulation, in: *Proceedings of the 21st ACM international conference on Information and knowledge management (CIKM '12)*, 2012, pp. 245–254.
[34] D. Li, Z. Xu, S. Li, X. Sun, A. Gupta and K.Sycara, Link Recommendation for Promoting Information Diffusion in Social Network D, in: *Proceedings of the 22nd International Conference on World Wide Web (WWW '13)*, 2013, pp. 185–186.
[35] D. Koschützki, K.A. Lehmann, L. Peeters, S. Richter, D.Tenfelde-Podehl and O. Zlotowski, Centrality Indices, in: *Network Analysis: Methodological Foundations*, U. Brandes and T. Erlebach, eds, Springer, 2005, pp. 16–61.
[36] R.E. Tarjan, A note on finding the bridges of a graph, *Information Processing Letters* **2**(6) (1974), 160–161.
[37] A.E. Sariyüce, K. Kaya, E. Saule and U.V. Çatalyürek, Graph Manipulations for Fast Centrality Computation, *ACM Transactions on Knowledge Discovery from Data* **11**(3) (2017), 26:1–26:25.
[38] A. Khan, Y. Ye and C. L, *On Uncertain Graphs*, Morgan & Claypool, 2018.
[39] V. Kassiano, A. Gounaris, A.N. Papadopoulos and K. Tsichlas, Mining Uncertain Graphs: An Overview, in: *Proceedings of the International Workshop on Algorithmic Aspects of Cloud Computing (ALGOCLOUD 2016)*, 2017, pp. 87–116.
[40] M. Potamias, F. Bonchi, A. Gionis and G. Kollios, K-nearest neighbors in uncertain graphs, in: *Proceedings of the VLDB Endowment (PVLDB)*, Vol. 3, 2010, pp. 997–1008.
[41] L. Liu, R. Jin, C. Aggarwal and Y. Shen, Reliable clustering on uncertain graphs, in: *Proceedings of the 2012 IEEE 12th International Conference on Data Mining (ICDM 2012)*, 2012, pp. 459–468.
[42] M. Kimura, K. Saito and H. Motoda, Blocking links to minimize contamination spread in a social network, *ACM Transactions on Knowledge Discovery from Data* **3** (2009), 9:1–9:23.
[43] B. Klimt and Y. Yang, The Enron corpus: A new dataset for email classification research, in: *Proceedings of the 2004 European Conference on Machine Learning (ECML'04)*, 2004, pp. 217–226.