Nonbacktracking Bounds on the Influence in Independent Cascade Models

Emmanuel Abbe* Sanjeev Kulkarni† Eun Jee Lee‡

June 30, 2017

Abstract

This paper develops upper and lower bounds on the influence measure in a network, more precisely, the expected number of nodes that a seed set can influence in the independent cascade model. In particular, our bounds exploit nonbacktracking walks, Fortuin–Kasteleyn–Ginibre (FKG) type inequalities, and are computed by message passing implementation. Nonbacktracking walks have recently allowed for headways in community detection, and this paper shows that their use can also impact the influence computation. Further, we provide a knob to control the trade-off between the efficiency and the accuracy of the bounds. Finally, the tightness of the bounds is illustrated with simulations on various network models.

1 Introduction

Influence propagation is concerned with the diffusion of information (or viruses) from initially influenced (or infected) nodes, called *seeds*, in a network. Understanding how information propagates in networks has become a central problem in a broad range of fields, such as viral marketing [17], sociology [8, 19, 22], communication [12], epidemiology [20], and social network analysis [23].

One of the most fundamental questions on influence propagation is to estimate the *influence*, i.e. the expected number of influenced nodes at the end of the propagation given some seeds. Estimating the influence is central to various research problems related to influence propagation, such as the widely-known influence maximization problem — finding a set of k nodes that maximizes the expected number of influenced nodes.

Recent studies in the influence propagation have proposed heuristic algorithms [11, 18, 3, 7, 21] for the influence maximization problem while using Monte Carlo (MC) simulations to approximate the influence. Despite its simplicity, approximating the influence via MC simulations is far from ideal for large networks; in particular, MC may require a large amount of computations in order to stabilize the approximation.

*Program in Applied and Computational Mathematics, and the Department of Electrical Engineering, Princeton University, Princeton, USA, eabbe@princeton.edu.

†The Department of Electrical Engineering, Princeton University, Princeton, USA, kulkarni@princeton.edu.

‡Program in Applied and Computational Mathematics, Princeton University, Princeton, USA, ejlee@princeton.edu.

To overcome the limitations of Monte Carlo simulations, many researchers have been taking theoretical approaches to approximate the influence of given seeds in a network. Draief et al., [5] introduced an upper bound for the influence by using the spectral radius of the adjacency matrix. Tighter upper bounds were later suggested in [16] which relate the ratio of influenced nodes in a network to the spectral radius of the so-called *Hazard* matrix. Further, improved upper bounds which account for *sensitive* edges were introduced in [15].

In contrast, there has been little work on finding a tight lower bound for the influence. A few exceptions include the work by Khim et al. [13], where the lower bound is obtained by only considering the infections through the *maximal-weighted* paths.

In this paper, we propose both upper and lower bounds on the influence using nonbacktracking walks and Fortuin–Kasteleyn–Ginibre (FKG) type inequalities. The bounds can be efficiently obtained by message passing implementation. This shows that nonbacktracking walks can also impact influence propagation, making another case for the use of nonbacktracking walks in graphical model problems as in [14, 9, 2, 1], discussed later in the paper. Further, we provide a parametrized version of the bounds that can adjust the trade-off between the efficiency and the accuracy of the bounds.

2 Background

We introduce here the independent cascade model and provide background for the main results.

Definition 1 (Independent Cascade Model)**.** *Consider a directed graph $G = (V, \vec{E})$ with $|V| = n$, a transmission probability matrix $\mathcal{P} \in [0,1]^{n\times n}$, and a seed set $S_0 \subseteq V$. For all $u \in V$, let $N(u)$ be the set of neighbors of node u. The independent cascade model $IC(G, \mathcal{P}, S_0)$ sequentially generates the infected set $S_t \subseteq V$ for each discrete time $t \geq 1$ as follows. At time t, S_t is initialized to be an empty set. Then, each node $u \in S_{t-1}$ attempts to infect $v \in N(u) \cap (V \setminus \cup_{i=0}^{t-1} S_i)$ with probability $\mathcal{P}_{uv}$, i.e. node u infects its uninfected neighbor v with probability $\mathcal{P}_{uv}$. If v is infected at time t, add v to S_t. The process stops at T if $S_T = \emptyset$ at the end of the step $t = T$. The set of the infected nodes at the end of propagation is defined as $S = \cup_{i=0}^{T-1} S_t$.*

We often refer as an edge (u, v) being *open* if node u infects node v. The IC model can also be defined on an undirected graph by replacing each edge $e = (u, v)$ which has the transmission probability p_e with two directed edges $\overrightarrow{(u, v)}, \overrightarrow{(v, u)}$ and assigning transmission probabilities $\mathcal{P}_{uv} = \mathcal{P}_{vu} = p_e$.

The IC model is equivalent to the live-arc graph model, where the infection happens at once, rather than sequentially. The live-arc graph model first decides the state of every edge with a Bernoulli trial, i.e. edge (u, v) is open independently with probability $\mathcal{P}_{uv}$ and closed, otherwise. Then, the set of infected nodes is defined as the nodes that are connected to at least one of the seeds by the open edges.

Definition 2 (Influence)**.** *The expected number of nodes that are infected at the end of the propagation process is called the* influence *(rather than the expected influence, with a slight abuse of terminology) of $IC(G, \mathcal{P}, S_0)$, and is defined as*

$$\sigma(S_0) \quad = \quad \sum_{v \in V} \mathbb{P}(v \textit{ is infected}). \tag{1}$$

It is shown in [4] that computing the influence $\sigma(S_0)$ in the independent cascade model $IC(G, \mathcal{P}, S_0)$ is #P-hard, even with a single seed, i.e. $|S_0| = 1$.

Next, we define nonbacktracking (NB) walks on a directed graph. Nonbacktracking walks have already been used for studying the characteristics of networks. To the best of our knowledge, the use of NB walks in the context of epidemics was first introduced in the paper of Karrer et al. [10] and later applied to percolation in [9]. In particular, Karrer et al. reformulate the spread of infections as a message passing process and demonstrate how the resulting equations can be used to calculate the upper bound on the number of nodes that are susceptible at a given time. As we shall see, we take a different approach to the use of the NB walks, which focuses on the effective contribution of a node in infecting another node and accumulates such contributions to obtain upper and lower bounds. More recently, nonbacktracking walks are used for community detection [14, 2, 1].

Definition 3 (Nonbacktracking Walk)**.** *Let $G = (V, E)$ be a directed graph. A nonbacktracking walk of length k is defined as $w^{(k)} = (v_0, v_1, \ldots, v_k)$, where $v_i \in V$ and $(v_{i-1}, v_i) \in E$ for all $i \in [k]$, and $v_{i-1} \neq v_{i+1}$ for $i \in [k-1]$.*

We next recall a key inequality introduced by Fortuin et. al [6].

Theorem 1 (FKG Inequality)**.** *Let Γ be a finite partially ordered set, ordered by $\prec$ with $(\Gamma, \prec)$ a distributive lattice and μ be a positive measure on Γ satisfying the following condition: for all $x, y \in \Gamma$,*

$$\mu(x \wedge y)\mu(x \vee y) \quad \geq \quad \mu(x)\mu(y),$$

where $x \wedge y = \max\{z \in \Gamma : z \preceq x, z \preceq y\}$ and $x \vee y = \min\{z \in \Gamma : y \preceq z, y \preceq z\}$. Let f and g be both increasing (or both decreasing) functions on Γ. Then,

$$(\sum_{x \in \Gamma} \mu(x))(\sum_{x \in \Gamma} f(x)g(x)\mu(x)) \quad \geq \quad (\sum_{x \in \Gamma} f(x)\mu(x))(\sum_{x \in \Gamma} g(x)\mu(x)). \tag{2}$$

FKG inequality is instrumental in studying the influence propagation since the probability that a node is influenced is nondecreasing with respect to the partial order of random variables describing the states, open or closed, of the edges.

3 Nonbacktracking bounds on the influence

In this section, we presents upper and lower bounds on the influence in the independent cascade model and explain the motivations and intuitions of the bounds. The bounds are computed efficiently by algorithms which utilize nonbacktracking walks and FKG inequalities. In particular, the upper bound on a network based on a graph $G(V, E)$ runs in $O(|V|^2+|V||E|)$ and the lower bound runs in $O(|V| + |E|)$, whereas Monte Carlo simulation would require $O(|V|^3 + |V|^2|E|)$ computations without knowing the variance of the influence, which is harder to estimate than the influence. The reason for the large computational complexity of MC is that in order to ensure that the standard error of the estimation does not grow with

respect to $|V|$, MC requires $O(|V|^2)$ computations. Hence, for large networks, where MC may not be feasible, our algorithms can still provide bounds on the influence.

Furthermore, from our computable upper σ^+ and lower bounds σ^-, we can compute an upper bound on the variance given by $(\sigma^+ - \sigma^-)^2/4$. This could be used to estimate the number of computations needed by MC. Computing the upper bound on the variance with the proposed bounds can be done in $O(|V|^2 + |V||E|)$, whereas computing the variance with MC simulation requires $O(|V|^5 + |V|^4|E|)$.

3.1 Nonbacktracking upper bounds (NB-UB)

We start by defining the following for the independent cascade model $IC(G, \mathcal{P}, S_0)$, where $G = (V, E)$ and $|V| = n$.

Definition 4. *For any $v \in V$, we define the set of in-neighbors $N^-(v) = \{u \in V : (u, v) \in E\}$ and the set of out-neighbors $N^+(v) = \{u \in V : (v, u) \in E\}$.*

Definition 5. *For any $v \in V$ and $l \in \{0, \ldots, n-1\}$, the set $P_l(S_0 \to v)$ is defined as the set of all paths with length l from any seed $s \in S_0$ to v. We call a path P is* open *if every edge in P is open.*

Definition 6. *For any $u, v \in V$, $l \in \{0, \ldots, n-1\}$, and $S \subseteq V$, we define the events*

$$\begin{aligned} A(v) &= \{v \text{ is infected}\} && (3)\\ A_l(v) &= \cup_{P \in P_l(S_0 \to v)} \{P \text{ is open}\} && (4)\\ A_l(u \to v) &= \cup_{P \in P_l(S_0 \to u), P \not\ni v} \{P \text{ is open and edge } (u, v) \text{ is open}\} && (5)\\ A_{l,S}(v) &= \cup_{P \in \{P' \in P_l(S_0 \to v) : P' \not\ni w, \forall w \in S\}} \{P \text{ is open}\}, && (6) \end{aligned}$$

and the probabilities

$$\begin{aligned} p(v) &= \mathbb{P}(A(v)) && (7)\\ p_l(v) &= \mathbb{P}(A_l(v)) && (8)\\ p_l(u \to v) &= \mathbb{P}(A_l(u \to v)). && (9) \end{aligned}$$

In other words, $A_l(v)$ is the event that node v is infected by open paths of length l, $A_l(u \to v)$ is the event that v is infected by node u with open paths of length $l + 1$, i.e. there exists an open path of length $l + 1$ from a seed to v that ends with edge (u, v), and $A_{l,S}(v)$ is the event that node v is infected by length l open paths which do not include any node in S.

Lemma 1. *For any $v \in V$,*

$$p(v) \leq 1 - \prod_{l=0}^{n-1} (1 - p_l(v)). \quad (10)$$

For any $v \in V$ and $l \in \{0, \ldots, n-1\}$,

$$p_l(v) \leq 1 - \prod_{u \in N^-(v)} (1 - p_l(u \to v)). \quad (11)$$

Proof. Recall that $p(v) = \mathbb{P}(A(v))$, $p_l(v) = \mathbb{P}(A_l(v))$, and $p_l(u \to v) = \mathbb{P}(A_l(u \to v))$.

$$\begin{aligned} p(v) &= \mathbb{P}(\cup_{l=0}^{n-1} A_l(v)) && (12)\\ &= 1 - \mathbb{P}(\cap_{l=0}^{n-1} A_l(v)^C) && (13)\\ &\leq 1 - \prod_{l=0}^{n-1} \mathbb{P}(A_l(v)^C) && (14)\\ &= 1 - \prod_{l=0}^{n-1} (1 - p_l(v)). && (15) \end{aligned}$$

Equation (14) follows from positive correlation among the events $A_l(v)^C$ for all $v \in V$, which can be proved by FKG inequality. Similarly,

$$\begin{aligned} p_l(v) &= \mathbb{P}(\cup_{u \in N^-(v)} A_l(u \to v)) && (16)\\ &= 1 - \mathbb{P}(\cap_{u \in N^-(v)} A_l(u \to v)^C) && (17)\\ &\leq 1 - \prod_{u \in N^-(v)} \mathbb{P}(A_l(u \to v)^C) && (18)\\ &= 1 - \prod_{u \in N^-(v)} (1 - p_l(u \to v)). && (19) \end{aligned}$$

□

Lemma 1 suggests that given $p_l(u \to v)$, we may compute an upper bound on the influence. Ideally, $p_l(u \to v)$ can be computed by considering all paths with length l. However, this results in exponential complexity $O(n^l)$, as l gets up to $n-1$. Thus, we present an efficient way to compute an upper bound $\mathrm{UB}_l(u \to v)$ on $p_l(u \to v)$, which in turns gives an upper bound $\mathrm{UB}_l(v)$ on $p_l(v)$, with the following recursion formula.

Definition 7. *For all $l \in \{0, \ldots, n-1\}$ and $u, v \in V$ such that $(u, v) \in E$, $\mathrm{UB}_l(u) \in [0, 1]$ and $\mathrm{UB}_l(u \to v) \in [0, 1]$ are defined recursively as follows.*
Initial condition: *For every $s \in S_0$, $s^+ \in N^+(s)$, $u \in V \setminus S_0$, and $v \in N^+(u)$,*

$$\begin{aligned} &\mathrm{UB}_0(s) = 1,\ \mathrm{UB}_0(s \to s^+) = \mathcal{P}_{ss^+} && (20)\\ &\mathrm{UB}_0(u) = 0,\ \mathrm{UB}_0(u \to v) = 0. && (21) \end{aligned}$$

Recursion: *For every $l > 0$, $s \in S_0$, $s^+ \in N^+(s)$, $s^- \in N^-(s)$, $u \in V \setminus S_0$, and $v \in N^+(u) \setminus S_0$,*

$$\begin{aligned} &\mathrm{UB}_l(s) = 0, \mathrm{UB}_l(s \to s^+) = 0, \mathrm{UB}_l(s^- \to s) = 0 && (22)\\ &\mathrm{UB}_l(u) = 1 - \prod_{w \in N^-(u)} (1 - \mathrm{UB}_{l-1}(w \to u)) && (23)\\ &\mathrm{UB}_l(u \to v) = \begin{cases} \mathcal{P}_{uv}(1 - \frac{1 - \mathrm{UB}_l(u)}{1 - \mathrm{UB}_{l-1}(v \to u)}), & \text{if } v \in N^-(u) \\ \mathcal{P}_{uv} \mathrm{UB}_l(u), & \text{otherwise.} \end{cases} && (24) \end{aligned}$$

Equation (22) follows from that for any seed node $s \in S_0$ and for all $l > 0$, $p_l(s) = 0$, $p_l(s \to s^+) = 0$, $p_l(s^- \to s) = 0$. A naive way to compute $\mathrm{UB}_l(u \to v)$ is $\mathrm{UB}_l(u \to v) =$

$\mathcal{P}_{uv}\mathrm{UB}_{l-1}(u)$, but this results in an extremely loose bound due to the backtracking. For a tighter bound, we use nonbacktracking in Equation (24), i.e. when computing $\mathrm{UB}_l(u\to v)$, we ignore the contribution of $\mathrm{UB}_{l-1}(v\to u)$.

Theorem 2. *For any independent cascade model $IC(G,\mathcal{P},S_0)$,*

$$\sigma(S_0) \;\leq\; \sum_{v\in V}(1-\prod_{l=0}^{n-1}(1-\mathrm{UB}_l(v))) =: \sigma^+(S_0), \tag{25}$$

where $\mathrm{UB}_l(v)$ is obtained recursively as in Definition 7.

Proof. We provide a proof by induction. The initial condition, for $l=0$, can be easily checked. For every $s\in S_0,\ s^+\in N^+(s),\ u\in V\setminus S_0,$ and $v\in N^+(u)$,

$$\begin{aligned} p_0(s)=1 &\leq \mathrm{UB}_0(s)=1 && (26)\\ p_0(s\to s^+)=\mathcal{P}_{ss^+} &\leq \mathrm{UB}_0(s\to s^+)=\mathcal{P}_{ss^+} && (27)\\ p_0(u)=0 &\leq \mathrm{UB}_0(u)=0 && (28)\\ p_0(u\to v)=0 &\leq \mathrm{UB}_0(u\to v)=0. && (29)\end{aligned}$$

For each $l\leq L$, assume that $p_l(v)\leq \mathrm{UB}_l(v)$ and $p_l(u\to v)\leq \mathrm{UB}_l(u\to v)$ for all $u,v\in V$.

Since $p_l(s)=p_l(s\to s^+)=p_l(s^-\to s)=0$ for every $l\geq 1,\ s\in S_0,\ s^+\in N^+(s),$ and $s^-\in N^-(s)$, it is sufficient to show $p_{L+1}(v)\leq \mathrm{UB}_{L+1}(v)$ and $p_{L+1}(u\to v)\leq \mathrm{UB}_{L+1}(u\to v)$ for all $u\in V\setminus S_0,$ and $v\in N^+(u)$.

For simplicity, for any pair of events (A,B), use the notation AB for $A\cap B$.

For any $v\in V\setminus S_0$,

$$p_{L+1}(v)=\mathbb{P}(\cup_{u\in N^-(v)}E_{uv}A_{L,\{v\}}(u)), \tag{30}$$

where E_{uv} denotes the event that edge (u,v) is open, i.e. $\mathbb{P}(E_{uv})=\mathcal{P}_{uv}$. Thus,

$$\begin{aligned} p_{L+1}(v) &= 1-\mathbb{P}(\cap_{u\in N^-(v)}(E_{uv}A_{L,\{v\}}(u))^C) && (31)\\ &\leq 1-\prod_{u\in N^-(v)}(1-\mathbb{P}(E_{uv}A_{L,\{v\}}(u))) && (32)\\ &= 1-\prod_{u\in N^-(v)}(1-p_L(u\to v)) && (33)\\ &\leq 1-\prod_{u\in N^-(v)}(1-\mathrm{UB}_L(u\to v))=\mathrm{UB}_{L+1}(v), && (34)\end{aligned}$$

where Equation (32) is obtained by the positive correlation among the events $E_{uv}A_{L,\{v\}}(u)$, and Equation (34) comes from the assumption.

For any $v\in V\setminus S_0$ and $w\in N^+(v)$,

$$\begin{aligned} p_{L+1}(v\to w) &= \mathbb{P}(E_{vw}A_{L+1,\{w\}}(v)). && (35)\\ &= \mathcal{P}_{vw}\mathbb{P}(A_{L+1,\{w\}}(v)) && (36)\end{aligned}$$

Equation (36) follows from the independence between the events E_{vw} and $A_{L+1,\{w\}}(v)$.
If $w \in N^-(v)$,

$$\begin{aligned} p_{L+1}(v \to w) &= \mathcal{P}_{vw}\mathbb{P}(\cup_{u \in N^-(v)\setminus\{w\}} E_{uv} A_{L,\{v,w\}}(u)) && (37)\\ &\leq \mathcal{P}_{vw}\left(1 - \prod_{u \in N^-(v)\setminus\{w\}} (1 - \mathbb{P}(E_{uv} A_{L,\{v,w\}}(u)))\right) && (38)\\ &\leq \mathcal{P}_{vw}\left(1 - \prod_{u \in N^-(v)\setminus\{w\}} (1 - p_L(u \to v))\right) && (39)\\ &\leq \mathcal{P}_{vw}\left(1 - \prod_{u \in N^-(v)\setminus\{w\}} (1 - \mathrm{UB}_L(u \to v))\right), && (40) \end{aligned}$$

Equation (40) holds, since the two events satisfy $E_{uv}A_{L,\{v,w\}}(u) \subseteq E_{uv}A_{L,\{v\}}(u)$.
Recall that, if $w \in N^-(v)$,

$$\begin{aligned} \mathrm{UB}_{L+1}(v \to w) &= \mathcal{P}_{vw}(1 - \frac{1 - \mathrm{UB}_{L+1}(v)}{1 - \mathrm{UB}_L(w \to v)}) && (41)\\ &= \mathcal{P}_{vw}(1 - \prod_{u \in N^-(v)\setminus\{w\}} (1 - \mathrm{UB}_L(u \to v))). && (42) \end{aligned}$$

Thus, $p_{L+1}(v \to w) \leq \mathrm{UB}_{L+1}(v \to w)$, for all $w \in N^+(v) \cap N^-(v)$.
If $w \notin N^-(v)$,

$$\begin{aligned} p_{L+1}(v \to w) &= \mathcal{P}_{vw}\mathbb{P}(\cup_{u \in N^-(v)} E_{uv} A_{L,\{v,w\}}(u)) && (43)\\ &\leq \mathcal{P}_{vw}\left(1 - \prod_{u \in N^-(v)} (1 - \mathrm{UB}_L(u \to v))\right) && (44)\\ &= \mathcal{P}_{vw}\mathrm{UB}_{L+1}(v) = \mathrm{UB}_{L+1}(v \to w), && (45) \end{aligned}$$

Hence, $p_{L+1}(v \to w) \leq \mathrm{UB}_{L+1}(v \to w)$, for all $w \in N^+(v)$, concluding the induction proof.
Finally, by Lemma 1,

$$\begin{aligned} \sigma(S_0) &\leq \sum_{v \in V}(1 - \prod_{l=0}^{n-1}(1 - p_l(v))) && (46)\\ &\leq \sum_{v \in V}(1 - \prod_{l=0}^{n-1}(1 - \mathrm{UB}_l(v))) = \sigma^+(S_0). && (47) \end{aligned}$$

□

3.2 Algorithm for NB-UB

Next, we present Nonbacktracking Upper Bound (NB-UB) algorithm which efficiently computes $\mathrm{UB}_l(v)$ and $\mathrm{UB}_l(u \to v)$ by message passing. At l-th iteration, the variables in NB-UB represent as follows.

- S_l : set of nodes that are processed at l-th iteration.
- $\mathrm{M}_{\mathrm{curr}}(v) = \{(u, \mathrm{UB}_{l-1}(u \to v)) : u \text{ is an in-neighbor of } v, \text{ and } u \in S_{l-1}\}$,
  set of pairs (previously processed in-neighbor u of v, incoming message from u to v).
- $\mathrm{MSrc}(v) = \{u : u \text{ is a in-neighbor of } v, \text{ and } u \in S_{l-1}\}$,
  set of in-neighbor nodes of v that were processed at the previous step.
- $\mathrm{M}_{\mathrm{curr}}(v)[u] = \mathrm{UB}_{l-1}(u \to v)$, the incoming message from u to v.
- $\mathrm{M}_{\mathrm{next}}(v) = \{(u, \mathrm{UB}_{l}(u \to v)) : u \text{ is an in-neighbor of } v, \text{ and } u \in S_{l}\}$,
  set of pairs (currently processed in-neighbor u, next iteration's incoming message from u to v).

Algorithm 1 Nonbacktracking Upper Bound (NB-UB)

Initialize: $\mathrm{UB}_l(v) = 0$ for all $0 \le l \le n-1$ and $v \in V$
Initialize: Insert $(v, 1)$ to $\mathrm{M}_{\mathrm{next}}(v)$ for all $v \in S_0$
for $l = 0$ **to** $n-1$ **do**
  for $u \in S_l$ **do**
    $\mathrm{M}_{\mathrm{curr}}(u) = \mathrm{M}_{\mathrm{next}}(u)$
    Clear $\mathrm{M}_{\mathrm{next}}(u)$
    $\mathrm{UB}_l(u) = \texttt{ProcessIncomingMsg}_{\texttt{UB}}(\mathrm{M}_{\mathrm{curr}}(u))$
    for $v \in N^+(u) \setminus S_0$ **do**
      S_{l+1}.insert(v)
      if $v \in \mathrm{MSrc}(u)$ **then**
        $\mathrm{UB}_l(u \to v) = \texttt{GenerateOutgoingMsg}_{\texttt{UB}}(\mathrm{M}_{\mathrm{curr}}(u)[v], \mathrm{UB}_l(u), \mathcal{P}_{uv})$
        $\mathrm{M}_{\mathrm{next}}(v)$.insert$((u, \mathrm{UB}_l(u \to v)))$.
      else
        $\mathrm{UB}_l(u \to v) = \texttt{GenerateOutgoingMsg}_{\texttt{UB}}(0, \mathrm{UB}_l(u), \mathcal{P}_{uv})$
        $\mathrm{M}_{\mathrm{next}}(v)$.insert$((u, \mathrm{UB}_l(u \to v)))$.
Output: $\mathrm{UB}_l(u)$ for all l, u

At the beginning, every seed node $v \in S_0$ is initialized such that $\mathrm{M}_{\mathrm{curr}}(v) = \{(v, 1)\}$. For each l-th iteration, every node u in S_l is processed as follows. First, $\texttt{ProcessIncomingMsg}_{\texttt{UB}}(\mathrm{M}_{\mathrm{curr}}(u))$ computes $\mathrm{UB}_l(u)$ as in Equation (23). Second, u passes a message to its neighbor $v \in N^+(u) \setminus S_0$ along the edge (u, v), and v stores (inserts) the message in $\mathrm{M}_{\mathrm{next}}(v)$ for the next iteration. The message contains 1) the source of the message, u, and 2) $\mathrm{UB}_l(u \to v)$, which is computed as in Equation (24), by the function $\texttt{GenerateOutgoingMsg}_{\texttt{UB}}$. Finally, the algorithm outputs $\mathrm{UB}_l(u)$ for all $u \in V$ and $l \in \{0, \ldots, n-1\}$, and the upper bound $\sigma^+(S_0)$ is computed by Equation (25).

Next, we illustrate how the algorithm runs on a small network in Figure 1. The independent cascade model $IC(G, \mathcal{P}, S_0)$ is defined on an undirected graph $G = (V, E)$, where $V = \{a, b, c, d\}$, $S_0 = \{b\}$, and every edge has the same transmission probability p. For each l, Table 1 shows the values of the key variables, S_l, $\mathrm{M}_{\mathrm{curr}}$, and UB_l, in the algorithm and $\mathrm{UB}_l(u \to v)$ for every pair u, v such that $u \in S_l$ and $v \in N^+(u) \setminus S_0$.

For example, at $l = 2$, since $S_2 = \{d\}$, node d is processed. Recall that, at $l = 1$, node a sent the message $(a, \mathrm{UB}_1(a \to d))$ to d, and node c sent the message $(c, \mathrm{UB}_1(c \to d))$ to d.

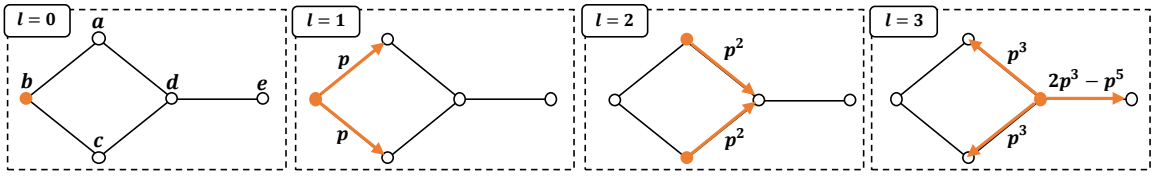


Figure 1: The step-wise illustration of NB-UB algorithm on the example network.

	$l=0$		$l=1$		$l=2$		$l=3$	
S_l	$\{b\}$		$\{a,c\}$		$\{d\}$		$\{a,c,e\}$	
	$\mathrm{M}_{\mathrm{curr}}$	UB_0	$\mathrm{M}_{\mathrm{curr}}$	UB_1	$\mathrm{M}_{\mathrm{curr}}$	UB_2	$\mathrm{M}_{\mathrm{curr}}$	UB_3
a	$\varnothing$	0	$\{(b,p)\}$	p	$\varnothing$	0	$\{(d,p^3)\}$	p^3
b	$\{(b,1)\}$	1	$\varnothing$	0	$\varnothing$	0	$\varnothing$	0
c	$\varnothing$	0	$\{(b,p)\}$	p	$\varnothing$	0	$\{(d,p^3)\}$	p^3
d	$\varnothing$	0	$\varnothing$	0	$\{(a,p^2),$ $(c,p^2)\}$	$2p^2-p^4$	$\varnothing$	0
e	$\varnothing$	0	$\varnothing$	0	$\varnothing$	0	$\{(d,2p^3-p^5)\}$	$2p^3-p^5$
Out -Prob	$\mathrm{UB}_0(b\to a)=p$ $\mathrm{UB}_0(b\to c)=p$		$\mathrm{UB}_1(a\to d)=p^2$ $\mathrm{UB}_1(c\to d)=p^2$		$\mathrm{UB}_2(d\to a)=p^3$ $\mathrm{UB}_2(d\to c)=p^3$ $\mathrm{UB}_2(d\to e)=2p^3-p^5$		$\mathrm{UB}_3(a\to d)=0$ $\mathrm{UB}_3(c\to d)=0$ $\mathrm{UB}_3(e\to d)=0$	

Table 1: The values of the key variables in NB-UB algorithm on the example network in Figure 1.

Thus,

$$\mathrm{M}_{\mathrm{curr}}(d) = \{(a,\mathrm{UB}_1(a\to d)),(c,\mathrm{UB}_1(c\to d))\} = \{(a,p^2),(c,p^2)\} \quad (48)$$
$$\mathrm{MSrc}(d) = \{a,c\}, \quad (49)$$

and node d is processed as follows.
First, compute $\mathrm{UB}_2(d)$ as

$$\mathrm{UB}_2(d) = \texttt{ProcessIncomingMsg}_{\texttt{UB}}(\mathrm{M}_{\mathrm{curr}}(d)) \quad (50)$$
$$= 1-(1-\mathrm{UB}_1(a\to d))(1-\mathrm{UB}_1(c\to d)) = 2p^2-p^4. \quad (51)$$

Next, set $S_3 = N^+(d)\backslash S_0 = \{a,c,e\}$, and compute $\mathrm{UB}_2(d\to a)$, $\mathrm{UB}_2(d\to c)$, and $\mathrm{UB}_2(d\to e)$ as

$$\mathrm{UB}_2(d\to a) = \texttt{GenerateOutgoingMsg}_{\texttt{UB}}(\mathrm{UB}_1(a\to d),\mathrm{UB}_2(d),\mathcal{P}_{da}) \quad (52)$$
$$= \mathcal{P}_{da}\left(1-\frac{1-\mathrm{UB}_2(d)}{1-\mathrm{UB}_1(a\to d)}\right) = p^3 \quad (53)$$
$$\mathrm{UB}_2(d\to c) = \texttt{GenerateOutgoingMsg}_{\texttt{UB}}(\mathrm{UB}_1(c\to d),\mathrm{UB}_2(d),\mathcal{P}_{dc}) \quad (54)$$
$$= \mathcal{P}_{dc}\left(1-\frac{1-\mathrm{UB}_2(d)}{1-\mathrm{UB}_1(c\to d)}\right) = p^3 \quad (55)$$
$$\mathrm{UB}_2(d\to e) = \texttt{GenerateOutgoingMsg}_{\texttt{UB}}(0,\mathrm{UB}_2(d),\mathcal{P}_{de}) \quad (56)$$
$$= \mathcal{P}_{de}\left(1-\frac{1-\mathrm{UB}_2(d)}{1-0}\right) = \mathcal{P}_{de}\mathrm{UB}_2(d) = 2p^3-p^5. \quad (57)$$

Then, node d send messages $(d, \mathrm{UB}_2(d \to a))$ to a, $(d, \mathrm{UB}_2(d \to c))$ to c, and $(d, \mathrm{UB}_2(d \to e))$ to e, concluding the process of the $l = 2$ step.

Computational complexity: Notice that for each iteration $l \in \{0, \ldots, n-1\}$, the algorithm accesses at most n nodes, and for each node, the functions $\texttt{ProcessIncomingMsg}_{\texttt{UB}}$ and $\texttt{GenerateOutgoingMsg}_{\texttt{UB}}$ are computed in $O(\deg(v))$. Therefore, the worst case computational complexity is $O(|V|^2 + |V||E|)$.

3.3 Nonbacktracking lower bounds (NB-LB)

A naive way to compute a lower bound on the influence in a network $IC(G, \mathcal{P}, S_0)$ is to reduce the network to a (spanning) tree network, by removing edges. Then, since there is a unique path from a node to another, we can compute the influence of the tree network, which is a lower bound on the influence in the original network, in $O(|V|)$. We take this approach of generating a subnetwork from the original network, yet we avoid the significant gap between the bound and the influence by considering the following directed acyclic subnetworks, in which there is no backtracking walk.

Definition 8 (Min-distance Directed Acyclic Subnetwork)**.** *Consider an independent cascade model $IC(G, \mathcal{P}, S_0)$ with $G = (V, E)$ and $|V| = n$. Let $d(S_0, v)$ be the minimum distance from a seed in S_0 to v. A minimum-distance directed acyclic subnetwork (MDAS), $IC(G', \mathcal{P}', S_0)$, where $G' = (V', E')$, is obtained as follows.*

- *$V' = \{v_1, ..., v_n\}$ is an ordered set of nodes such that $d(S_0, v_i) \leq d(S_0, v_j)$, for every $i < j$.*
- *$E' = \{(v_i, v_j) \in E : i < j\}$, i.e. remove edges from E whose source node comes later in the order than its destination node to obtain E'.*
- *$\mathcal{P}'_{v_i v_j} = \mathcal{P}_{v_i v_j}$, if $(v_i, v_j) \in E'$, and $\mathcal{P}'_{v_i v_j} = 0$, otherwise.*

If there are multiple ordered sets of vertices that satisfying the condition, we may choose an order arbitrarily.

For any $k \in [n]$, let $p(v_k)$ be the probability that $v_k \in V'$ is infected in the MDAS, $IC(G', \mathcal{P}', S_0)$. Since $p(v_k)$ is equivalent to the probability of the union of the events that an in-neighbor $u_i \in N^-(v_k)$ infects v_k, $p(v_k)$ can be computed by the principle of inclusion and exclusion. Thus, we may compute a lower bound on $p(v_k)$, using Bonferroni inequalities, if we know the probabilities that in-neighbors u and v both infects v_k, for every pair $u, v \in N^-(v_k)$. However, computing such probabilities can take $O(k^k)$. Hence, we present $\mathrm{LB}_l(v_k)$ which efficiently computes a lower bound on $p(v_k)$ by the following recursion.

Definition 9. *For all $v_k \in V'$, $\mathrm{LB}(v_k) \in [0, 1]$ is defined by the recursion on k as follows.* Initial condition: *for every $v_s \in S_0$,*

$$\mathrm{LB}(v_s) = 1. \tag{58}$$

Recursion: *for every $v_k \in V' \setminus S_0$,*

$$\mathrm{LB}(v_k) = \sum_{i=1}^{m^*} \left(\mathcal{P}'_{u_i v_k} \mathrm{LB}(u_i)(1 - \sum_{j=1}^{i-1} \mathcal{P}'_{u_j v_k}) \right), \tag{59}$$

where $N^-(v_k)=\{u_1,\dots,u_m\}$ *is the ordered set of in-neighbors of* v_k *in* $IC(G',\mathcal{P}',S_0)$ *and* $m^*=\max\{m'\le m:\sum_{j=1}^{m'-1}\mathcal{P}'_{u_jv_k}\le 1\}$.

Remark. Since the i-th summand in Equation (59) can utilize $\sum_{j=1}^{i-2}\mathcal{P}'_{u_jv_k}$, which is already computed in $(i-1)$-th summand, to compute $\sum_{j=1}^{i-1}\mathcal{P}'_{u_jv_k}$, the summation takes at most $O(\deg(v_k))$.

Theorem 3. *For any independent cascade model* $IC(G,\mathcal{P},S_0)$ *and its MDAS* $IC(G',\mathcal{P}',S_0)$,

$$\sigma(S_0)\ge\sum_{v_k\in V'}\mathrm{LB}(v_k)=:\sigma^-(S_0), \tag{60}$$

where $\mathrm{LB}(v_k)$ *is obtained recursively as in Definition 9.*

Proof. We provide a proof by induction. For any $v_k\in V'$, let $A(v_k)$ be the event that node v_k is infected in MDAS $IC(G',\mathcal{P}',S_0)$, and for every edge (v_j,v_k), let E_{v_j,v_k} be the event that edge (v_j,v_k) is open, i.e. $\mathbb{P}(E_{v_j,v_k})=\mathcal{P}'_{v_jv_k}$. Recall that $p(v_k)=\mathbb{P}(A(v_k))$.

The initial condition $k=1$ holds, since $p(v_1)=1\ge\mathrm{LB}(v_1)=1$ (v_1 is a seed).

For every $k\le K$, assume $p(v_k)\ge\mathrm{LB}(v_k)$.

For the node v_{K+1},

$$p(v_{K+1}) \;=\; \mathbb{P}(\cup_{v_j\in N^-(v_{K+1})}E_{v_jv_{K+1}}A(v_j)). \tag{61}$$

We re-label vertices in $N^-(v_{K+1})=\{u_1,\dots,u_{m(K+1)}\}$ where $m(K+1)=\text{in-deg}(v_{K+1})$, and let $\mathcal{Q}_{iK+1}=\mathcal{P}'_{u_iv_{K+1}}$. Then, for any integer $m\le m(K+1)$,

$$\begin{aligned}
p(v_{K+1}) &= \mathbb{P}(\cup_{i=1}^{m(K+1)}E_{u_iv_{K+1}}A(u_i)) && (62)\\
&\ge \mathbb{P}(\cup_{i=1}^{m}E_{u_iv_{K+1}}A(u_i)) && (63)\\
&\ge \sum_{i=1}^{m}\mathbb{P}(E_{u_iv_{K+1}}A(u_i))-\sum_{i=1}^{m}\sum_{j=1}^{i-1}\mathbb{P}(E_{u_iv_{K+1}}A(u_i)E_{u_jv_{K+1}}A(u_j)) && (64)\\
&= \sum_{i=1}^{m}\mathcal{Q}_{iK+1}\mathbb{P}(A(u_i))-\sum_{i=1}^{m}\sum_{j=1}^{i-1}\mathcal{Q}_{iK+1}\mathcal{Q}_{jK+1}\mathbb{P}(A(u_i)A(u_j)) && (65)\\
&\ge \sum_{i=1}^{m}\mathcal{Q}_{iK+1}\mathbb{P}(A(u_i))(1-\sum_{j=1}^{i-1}\mathcal{Q}_{jK+1}). && (66)
\end{aligned}$$

Equation (64) follows from the principle of inclusion and exclusion. Equation (65) results from the Independence between the event that an edge ending with v_{K+1} is open and the event that a node v_i is infected where $i<K+1$. Equation (66) holds since $\mathbb{P}(A(u_i))\ge\mathbb{P}(A(u_i)A(u_j))$.

Now, define $m^*=\max\{m'\le m(K+1):\sum_{j=1}^{m'-1}\mathcal{Q}_{jK+1}\le 1\}$. Then,

$$\begin{aligned}
p(v_{K+1}) &\ge \sum_{i=1}^{m^*}\mathcal{Q}_{iK+1}\mathbb{P}(A(u_i))(1-\sum_{j=1}^{i-1}\mathcal{Q}_{jK+1}) && (67)\\
&\ge \sum_{i=1}^{m^*}\mathcal{Q}_{iK+1}\mathrm{LB}(u_i)(1-\sum_{j=1}^{i-1}\mathcal{Q}_{jK+1}) && (68)\\
&= \mathrm{LB}(v_{K+1}). && (69)
\end{aligned}$$

Equation (68) follows since $1 - \sum_{j=1}^{i-1} \mathcal{Q}_{jK+1} \geq 0$ for all $i \leq m^*$ by the definition of m^*. Thus, $p(v_i) \geq \mathrm{LB}(v_i)$ for all $v_i \in V'$, concluding the induction proof.
Finally,

$$\sigma(S_0) \geq \sum_{i=1}^{n} p(v_i) \tag{70}$$

$$\geq \sum_{i=1}^{n} \mathrm{LB}(v_i) = \sigma^-(S_0). \tag{71}$$

Equation (70) holds since its right hand side equals to the influence of the MDAS, $IC(G', \mathcal{P}', S_0)$. □

3.4 Algorithm for NB-LB

Next, we present Nonbacktracking Lower Bound (NB-LB) algorithm which efficiently computes $\mathrm{LB}(v_k)$. At k-th iteration, the variables in NB-LB represent as follow.

- $\mathrm{M}(v_k) = \{(\mathrm{LB}(v_t), \mathcal{P}'_{v_t v_k}) : v_t \text{ is an in-neighbor of } v_k\}$, set of pairs (incoming message from an in-neighbor v_t to v_k, the transmission probability of edge (v_t, v_k)).
- $\mathrm{M}(v_k)_i = (\mathrm{M}(v_k)_{i,1}, \mathrm{M}(v_k)_{i,2})$, the i-th pair in $\mathrm{M}(v_k)$, for $i \geq 1$.

Algorithm 2 Nonbacktracking Lower Bound (NB-LB)

Input: directed acyclic network $IC(G', \mathcal{P}', S_0)$
Initialize: $\sigma^- = 0$
Initialize: Insert $(1, 1)$ to $\mathrm{M}(v_i)$ for all $v_i \in S_0$
for $k = 1$ **to** n **do**
  $\mathrm{LB}(v_k) = \texttt{ProcessIncomingMsg}_{\mathrm{LB}}(\mathrm{M}(v_k))$
  σ^- += $\mathrm{LB}(v_k)$
  for $v_t \in N^+(v_k) \setminus S_0$ **do**
    $\mathrm{M}(v_t)$.insert$((\mathrm{LB}(v_k), \mathcal{P}'_{v_k v_t}))$
Output: σ^-

At the beginning, every seed node $v \in S_0$ is initialized such that $\mathrm{M}(v) = \{(1, 1)\}$, and $\sigma^- = 0$. For each k-th iteration, node v_k is processed as follows. First, $\mathrm{LB}(v_k)$ is computed as in the Equation (59), by the function $\texttt{ProcessIncomingMsg}_{\mathrm{LB}}$, and added to σ^-. Second, v_k passes the message $(\mathrm{LB}(v_k), \mathcal{P}'_{v_k v_t})$ to its neighbor $v_t \in N^+(v_k) \setminus S_0$, and v_t stores (inserts) it in $\mathrm{M}(v_t)$. Finally, the algorithm outputs σ^-, the lower bound on the influence.

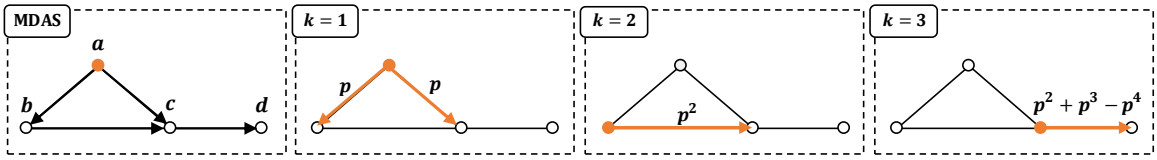


Figure 2: The step-wise illustration of NB-LB on the example network.

In Figure 2, we show an example for the lower bound computation by NB-LB on a small network $IC(G, \mathcal{P}, S_0)$ defined on an undirected graph $G = (V, E)$, where $V = \{a, b, c, d\}$,

$S_0 = \{a\}$, and every edge has the same transmission probability p. For each k, Table 2 shows the values of the key variables, $\mathrm{M}(v_k)$, $\mathrm{LB}(v_k)$, and $(\mathrm{LB}(v_k), \mathcal{P}'_{v_k v_t})$ for the out-neighbors $v_t \in N^+(v_k) \setminus S_0$, and shows the change in σ^-.

	$k=1$	$k=2$	$k=3$	$k=4$
v_k	a	b	c	d
$\mathrm{M}(v_k)$	$\{(1,1)\}$	$\{(1,p)\}$	$\{(1,p),(p,p)\}$	$\varnothing$
$\mathrm{LB}(v_k)$	1	p	$p+p^2-p^3$	$p^2+p^3-p^4$
$N^+(v_k) \setminus S_0$	$\{b,c\}$	$\{c\}$	$\{d\}$	$\varnothing$
$(\mathrm{LB}(v_k), \mathcal{P}'_{v_k v_t})$ to v_t	$(1,p)$ to b and c	(p,p) to c	$(p+p^2-p^3, p)$ to d	
σ^-	1	$1+p$	$1+2p+p^2-p^3$	$1+2p+2p^2-p^4$

Table 2: The values of the key variables in NB-LB on the example network in Figure 2.

From the undirected network, we obtain MDAS in Figure 2 as follows. Since $d(a, S_0) = 0$, $d(b, S_0) = d(c, S_0) = 1$ and $d(d, S_0) = 2$, we order the vertices as $\{v_1 = a, v_2 = b, v_3 = c, v_4 = d\}$ to satisfy that $d(v_i, S_0) \leq d(v_j, S_0)$, for every $i < j$. Then, NB-LB algorithm process the nodes $\{v_1 = a, v_2 = b, v_3 = c, v_4 = d\}$ sequentially.

For example, at $k=3$, node c is processed. Recall that at $k = 1$, node a sent the message $(\mathrm{LB}(a), \mathcal{P}'_{ac})$ to node c; at $k = 2$, node b sent the message $(\mathrm{LB}(b), \mathcal{P}'_{bc})$ to node c. Thus,

$$\mathrm{M}(c) = \{(\mathrm{LB}(a), \mathcal{P}'_{ac}), (\mathrm{LB}(b), \mathcal{P}'_{bc})\} = \{(1,p), (p,p)\}. \tag{72}$$

Then, it computes $\mathrm{LB}(c)$ by the function $\texttt{ProcessIncomingMsg}_{\mathtt{LB}}$.

$$\begin{aligned} \mathrm{LB}(c) &= \texttt{ProcessIncomingMsg}_{\mathtt{LB}}(\mathrm{M}(c)) & (73)\\ &= \mathcal{P}'_{ac}\mathrm{LB}(a) + \mathcal{P}'_{bc}\mathrm{LB}(b)(1 - \mathcal{P}'_{ac}) = p + p^2 - p^3. & (74) \end{aligned}$$

Recall that $\sigma^- = 1 + p$, at the end of iteration $k = 2$, so

$$\sigma^- = 1 + p + \mathrm{LB}(c) = 1 + 2p + p^2 - p^3. \tag{75}$$

Next, since $N^+(c) \setminus S_0 = \{d\}$, node c sends the message $(\mathrm{LB}(c), \mathcal{P}_{cd}) = (p + p^2 - p^3, p)$ to node d, concluding the process of the $k = 3$ step.

Computational complexity: Obtaining an arbitrary directed acyclic subnetwork from the original network takes $O(|V| + |E|)$. Next, the algorithm iterates through the nodes $V' = \{v_1, \ldots, v_n\}$. For each node v_k, $\texttt{ProcessIncomingMsg}_{\mathtt{LB}}$ takes $O(\deg(v_k))$ and v_k sends messages to its out-neighbors in $O(\deg(v_k))$. Hence, the worst case computational complexity is $O(|V| + |E|)$.

3.5 Tunable nonbacktracking bounds

In this section, we introduce the parametrized version of NB-UB and NB-LB which provide control to adjust the trade-off between the efficiency and the accuracy of the bounds.

Tunable nonbacktracking upper bounds (tNB-UB**):** The algorithm inputs the parameter t, which indicates the maximum length of the paths that the algorithm considers to

compute the exact, rather than the upper bound on, probability of infection. For every node $u \in V$, the algorithm computes $p_{\leq t}(u)$ that node u is infected by open paths whose length is less than or equal to t.

$$p_{\leq t}(u) \quad = \quad \mathbb{P}(\cup_{P \in \{\cup_{i=0}^{t} P_i(S_0 \rightarrow u)\}} \{P \text{ is open}\}). \tag{76}$$

Then, we start (non-parameterized) NB-UB algorithm from $l = t + 1$ with the new initial conditions: for all $u \in V$ and $v \in N^+(u)$,

$$\mathrm{UB}_t(u) = p_{\leq t}(u) \tag{77}$$

$$\mathrm{UB}_t(u \rightarrow v) = p_t(u \rightarrow v) \tag{78}$$

Finally, the upper bound by tNB-UB is computed as $\sum_{v \in V}(1 - \prod_{l=t}^{n-1}(1 - \mathrm{UB}_l(v)))$.

For higher values of t, the algorithm results in tighter upper bounds, while the computational complexity may increase exponentially for dense networks. Thus, this method is most applicable in sparse networks, where the degree of each node is bounded.

We present here the parametrized algorithms for NB-UB.

Algorithm 3 Tunable NB-UB (tNB-UB)

parameter: non-negative integer $t \leq n - 1$
Initialize: $\mathrm{UB}_t(v) = 0$ for all $t \leq l \leq n - 1$ and $v \in V$
for $u \in V$ **do**
  $\mathrm{UB}_t(u) = p_{\leq t}(u)$
  for $v \in N^+(u) \setminus S_0$ **do**
    if $p_t(u \rightarrow v) > 0$ **then**
      S_{t+1}.insert(v)
      $\mathrm{M}_{\mathrm{next}}(v)$.insert$(u, p_t(u \rightarrow v))$
for $l = t + 1$ **to** $n - 1$ **do**
  for $u \in S_l$ **do**
    $\mathrm{M}_{\mathrm{curr}}(u) = \mathrm{M}_{\mathrm{next}}(u)$
    Clear $\mathrm{M}_{\mathrm{next}}(u)$
    $\mathrm{UB}_l(u) = \texttt{ProcessIncomingMsg}_{\mathtt{UB}}(\mathrm{M}_{\mathrm{curr}}(u))$
    for $v \in N^+(u) \setminus S_0$ **do**
      S_{l+1}.insert(v)
      if $v \in \mathrm{M}_{\mathrm{curr}}(u)$ **then**
        $\mathrm{UB}_l(u \rightarrow v) = \texttt{GenerateOutgoingMsg}_{\mathtt{UB}}(\mathrm{M}_{\mathrm{curr}}(u)[v], \mathrm{UB}_l(u), \mathcal{P}_{uv})$
        $\mathrm{M}_{\mathrm{next}}(v)$.insert$((u, \mathrm{UB}_l(u \rightarrow v)))$.
      else
        $\mathrm{UB}_l(u \rightarrow v) = \texttt{GenerateOutgoingMsg}_{\mathtt{UB}}(0, \mathrm{UB}_l(u), \mathcal{P}_{uv})$
        $\mathrm{M}_{\mathrm{next}}(v)$.insert$((u, \mathrm{UB}_l(u \rightarrow v)))$.
Output: $\mathrm{UB}_l(u)$ for all $l = \{t, t + 1, \ldots, n - 1\}$, $u \in V$

Tunable nonbacktracking lower bounds (tNB-LB): We first order the vertex set as $V' = \{v_1, \ldots, v_n\}$, which satisfies $d(S_0, v_i) \leq d(S_0, v_j)$, for every $i < j$. Given a non-negative integer parameter $t \leq n$, we obtain a t-size subnetwork $IC(G[V_t], \mathcal{P}[V_t], S_0 \cap V_t)$, where $G[V_t]$ is the vertex-induced subgraph which is induced by the set of nodes $V_t = \{v_1, \ldots, v_t\}$, and

$\mathcal{P}[V_t]$ is the corresponding transmission probability matrix. For each $v_i \in V_t$, we compute the exact probability $p_t(v_i)$ that node v_i is infected in the subnetwork $IC(G[V_t], \mathcal{P}[V_t], S_0 \cap V_t)$. Then, we start (non-parameterized) NB-LB algorithm from $k = t + 1$ with the new initial condition: for all $k \leq t$,

$$\mathrm{LB}(v_k) = p_t(v_k). \tag{79}$$

Finally, tNB-LB computes the lower bound as $\sum_{v_k \in V'} \mathrm{LB}(v_k)$.

Algorithm 4 Tunable NB-LB (tNB-LB)

parameter: non-negative integer $t \leq n$
Initialize: $\sigma^- = 0$
for $k = 1$ **to** t **do**
  $\mathrm{LB}(v_k) = p_t(v_k)$
  σ^- += $\mathrm{LB}(v_k)$
  for $v_i \in \{N^+(v_k) \cap \{v_j : j > t\}\}$ **do**
    $\mathrm{M}(v_i)$.insert$((\mathrm{LB}(v_k), \mathcal{P}'_{v_k v_i}))$
for $k = t + 1$ **to** n **do**
  $\mathrm{LB}(v_k) = \mathtt{ProcessIncomingMsg}_{\mathtt{LB}}(\mathrm{M}(v_k))$
  σ^- += $\mathrm{LB}(v_k)$
  for $v_i \in N^+(v_k) \setminus S_0$ **do**
    $\mathrm{M}(v_i)$.insert$((\mathrm{LB}(v_k), \mathcal{P}'_{v_k v_i}))$
Output: σ^-

For a larger t, the algorithm results in tighter lower bounds. However, the computational complexity may increase exponentially with respect to t, the size of the subnetwork. This algorithm can adopt Monte Carlo simulations on the subnetwork to avoid the large computational complexity. However, this modification results in probabilistic lower bounds, rather than theoretically guaranteed lower bounds. Nonetheless, this can still give a significant improvement, because the Monte Carlo simulations on a smaller size of network require less computations to stabilize the estimation.

4 Experimental results

In this section, we evaluate the NB-UB and NB-LB in independent cascade models on a variety of classical synthetic networks.

Network generation : We consider 4 classical random graph models with the parameters shown as follows: Erdos Renyi random graphs with $ER(n = 1000, p = 3/1000)$, scale-free networks $SF(n = 1000, \alpha = 2.5)$, random regular graphs $Reg(n = 1000, d = 3)$, and random tree graphs with power-law degree distributions $T(n = 1000, \alpha = 3)$. For each graph model, we generate 100 networks $IC(G, pA, \{s\})$ as follows. The graph G is the largest connected component of a graph drawn from the graph model, the seed node s is selected randomly from the vertex set of G, and A is the adjacency matrix of G. Then, the corresponding IC model has the same transmission probability p for every edge.

Evaluation of bounds : For each network generated, we compute the following quantities for each $p \in \{0.1, 0.2, \ldots, 0.9\}$.

· σ_{mc}: the estimation of the influence with 1000000 Monte Carlo simulations.

· σ^{+}: the upper bound obtained by NB-UB.

· σ^{+}_{spec}: the spectral upper bound by [16].

· σ^{-}: the lower bound obtained by NB-LB.

· σ^{-}_{prob}: the probabilistic lower bound obtained by 10 Monte Carlo simulations.

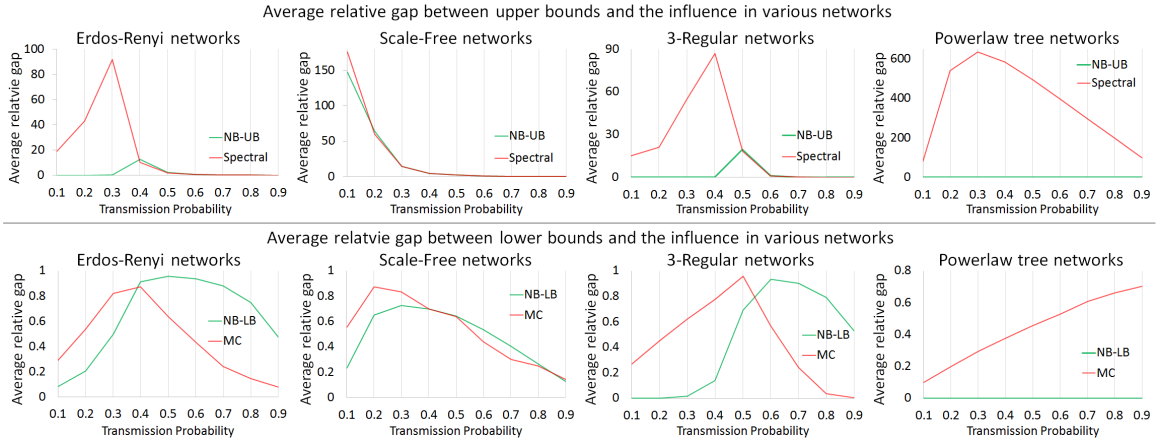


Figure 3: This figure compares the average relative gap of the bounds: NB-UB, the spectral upper bound in [16], NB-LB, and the probabilistic lower bound computed by MC simulations, for various types of networks.

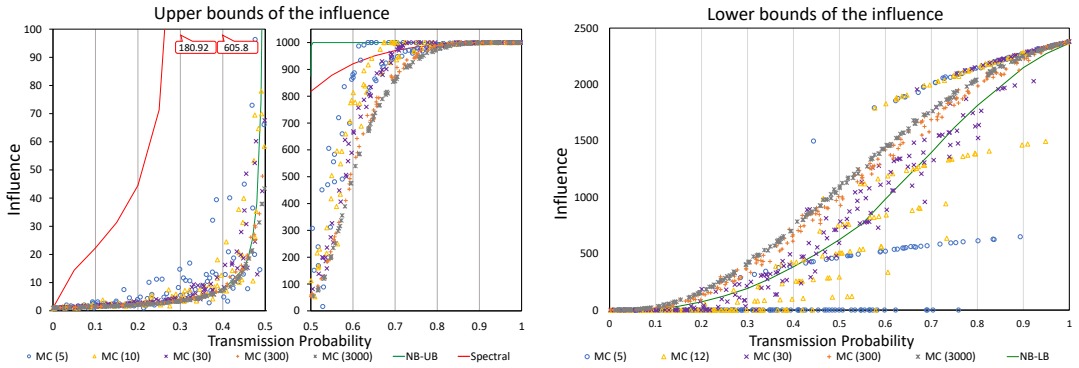
The probabilistic lower bound is chosen for the experiments, since there has not been any tight lower bound. The sample size of 10 is determined to overly match the computational complexity of NB-LB algorithm. In Figure 3, we compare the average relative gap of the bounds for every network model and for each transmission probability, where the true value is assumed to be σ_{mc}. For example, the average relative gap of NB-UB for 100 Erdos Renyi networks $\{\mathcal{N}_i\}_{i=1}^{100}$ with the transmission probability p is computed by $\frac{1}{100}\sum_{i\in[100]} \frac{\sigma^{+}[\mathcal{N}_i]-\sigma_{mc}[\mathcal{N}_i]}{\sigma_{mc}[\mathcal{N}_i]}$, where $\sigma^{+}[\mathcal{N}_i]$ and $\sigma_{mc}[\mathcal{N}_i]$ denote the NB-UB and the MC estimation, respectively, for the network $\mathcal{N}_i$.

Results : Figure 3 shows that NB-UB outperforms the upper bound in [16] for the Erdos-Renyi and random 3-regular networks, and performs comparably for the scale-free networks. Also, NB-LB gives tighter bounds than the MC bounds on the Erdos-Renyi, scale-free, and random regular networks when the transmission probability is small, $p < 0.4$. The NB-UB and NB-LB compute the exact influence for the tree networks since both algorithms avoid backtracking walks.

Next, we show the bounds on exemplary networks.

4.1 Upper bounds

Selection of networks : In order to illustrate a typical behavior of the bounds, we have chosen the network in Figure 4a as follows. First, we generate 100 random 3-regular graphs



(a) (b)

Figure 4: (a) The figure compares various upper bounds on the influence in the 3-regular network in section 4.1. The MC upper bounds are computed with various simulation sizes and shown with the data points indicated with MC(N), where N is the number of simulations. The spectral upper bound in [16] is shown in red line, and NB-UB is shown in green line.
(b) The figure shows lower bounds on the influence of a scale-free network in section 4.2. The probabilistic lower bounds shown with points are obtained from Monte Carlo simulations with various simulation sizes, and the data points indicated with MC(N) are obtained by N number of simulations. NB-LB is shown in green line.

G with 1000 nodes and assign a random seed s. Then, the corresponding IC model is defined as $IC(G, \mathcal{P} = pA, S_0 = \{s\})$, where A is the adjacency matrix, resulting in the same transmission probability p for every edge. For each network, we compute NB-UB and MC estimation of 1000 simulations. We note that with 1000 simulations, the estimation given by MC cannot guarantee stability. Yet, given limited time and resources, it is acceptable for the selection process. Then, we compute the score for each network. The score is defined as the sum of the square differences between the upper bounds and MC estimations over the transmission probability $p \in \{0.1, 0.2, \ldots, 0.9\}$. Finally, a graph whose score is the median from all 100 scores is chosen for Figure 4a.

Results : In figure 4a, we compare 1) the upper bounds introduced [16] and 2) the probabilistic upper bounds obtained by Monte Carlo simulations with 99% confidence level, to NB-UB. The upper bounds from MC simulations are computed with the various sample sizes $N \in \{5, 10, 30, 300, 3000\}$. It is evident from the figure that a larger sample size provides a tighter probabilistic upper bound. NB-UB outperforms the bound by [16] and the probabilistic MC bound when the transmission probability is relatively small. Further, it shows a similar trend as the MC simulations with a large sample size.

4.2 Lower bounds

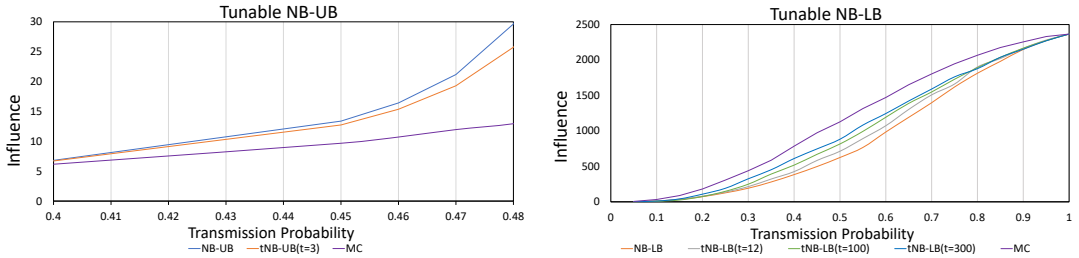
Selection of networks : We adopt a similar selection process as in the selection for upper bounds, but with the scale free networks, with 3000 nodes and $\alpha = 2.5$.

Results : We compare probabilistic lower bounds obtained by MC with 99% confidence level to NB-LB. The lower bounds from Monte Carlo simulations are computed with various sample sizes $N \in \{5, 12, 30, 300, 3000\}$, which accounts for a constant, $\log(n)$, $0.01n$, $0.1n$, and $n = |V|$. NB-LB outperforms the probabilistic bounds by MC with small sample sizes.

Recall that the computational complexity of the lower bound in algorithm 2 is $O(|V| + |E|)$, which is the computational complexity of a constant number of Monte Carlo simulations. In figure 4b, it shows that NB-LB is tighter than the probabilistic lower bounds with the same computational complexity, and it also agrees with the behavior of the MC simulations.

4.3 Tunable bounds

In this section, we present tunable upper and lower bounds on example networks.



(a) (b)

Figure 5: (a) NB-UB, tNB-UB with $t = 3$, and MC estimation with 10000 simulations on a 3-regular network with 100 nodes.
(b) NB-LB, tNB-LB with various $t \in \{12, 100, 300\}$, and MC estimation with 3000000 simulations on a scale-free network with 3000 nodes.

In Figure 5a, we show tNB-UB on a sample network. We consider a 3-regular network with 100 nodes and a single seed. Since the NB-UB gives a tight bound on $p < 0.4$, we plot tNB-UB on $p \in (0.4, 0.5)$ where it shows some improvements with small t.

In Figure 5b, we present tNB-LB on a scale-free network with 3000 nodes, $\alpha = 2.5$, and a single seed. We compare tNB-LB with various choices of $t \in \{1, 12, 100, 300\}$, and tNB-LB approaches the MC estimation as t grows.

5 Conclusion

In this paper, we propose both upper and lower bounds on the influence in the independent cascade models, and provide algorithms to efficiently compute the bounds. We extend the results by proposing tunable bounds which can adjust the trade off between the efficiency and the accuracy. Finally, the tightness and the performance of bounds are shown with experimental results. One can further improve the bounds considering r-nonbacktracking walks, i.e. avoiding cycles of length r rather than just backtracks, and we leave this for future study.

6 Acknowledgements

We would like to thank Colin Sandon for helpful discussions. This research was partly supported by the NSF CAREER Award CCF-1552131 and the ARO grant W911NF-16-1-0051.

References

[1] E. Abbe and C. Sandon. Detection in the stochastic block model with multiple clusters: proof of the achievability conjectures, acyclic bp, and the information-computation gap. *arXiv preprint arXiv:1512.09080*, 2015.

[2] C. Bordenave, M. Lelarge, and L. Massoulié. Non-backtracking spectrum of random graphs: community detection and non-regular ramanujan graphs. In *Foundations of Computer Science (FOCS), 2015 IEEE 56th Annual Symposium on*, pages 1347–1357. IEEE, 2015.

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

[4] W. Chen, Y. Yuan, and L. Zhang. Scalable influence maximization in social networks under the linear threshold model. In *Data Mining (ICDM), 2010 IEEE 10th International Conference on*, pages 88–97. IEEE, 2010.

[5] M. Draief, A. Ganesh, and L. Massoulié. Thresholds for virus spread on networks. In *Proceedings of the 1st international conference on Performance evaluation methodolgies and tools*, page 51. ACM, 2006.

[6] C. M. Fortuin, P. W. Kasteleyn, and J. Ginibre. Correlation inequalities on some partially ordered sets. *Communications in Mathematical Physics*, 22(2):89–103, 1971.

[7] A. Goyal, W. Lu, and L. V. Lakshmanan. Celf++: optimizing the greedy algorithm for influence maximization in social networks. In *Proceedings of the 20th international conference companion on World wide web*, pages 47–48. ACM, 2011.

[8] M. Granovetter. Threshold models of collective behavior. *American journal of sociology*, pages 1420–1443, 1978.

[9] B. Karrer, M. Newman, and L. Zdeborová. Percolation on sparse networks. *Physical review letters*, 113(20):208702, 2014.

[10] B. Karrer and M. E. Newman. Message passing approach for general epidemic models. *Physical Review E*, 82(1):016101, 2010.

[11] D. Kempe, J. Kleinberg, and É. Tardos. Maximizing the spread of influence through a social network. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 137–146. ACM, 2003.

[12] A. Khelil, C. Becker, J. Tian, and K. Rothermel. An epidemic model for information diffusion in manets. In *Proceedings of the 5th ACM international workshop on Modeling analysis and simulation of wireless and mobile systems*, pages 54–60. ACM, 2002.

[13] J. T. Khim, V. Jog, and P.-L. Loh. Computing and maximizing influence in linear threshold and triggering models. In *Advances in Neural Information Processing Systems*, pages 4538–4546, 2016.

[14] F. Krzakala, C. Moore, E. Mossel, J. Neeman, A. Sly, L. Zdeborová, and P. Zhang. Spectral redemption in clustering sparse networks. *Proceedings of the National Academy of Sciences*, 110(52):20935–20940, 2013.

[15] E. J. Lee, S. Kamath, E. Abbe, and S. R. Kulkarni. Spectral bounds for independent cascade model with sensitive edges. In *2016 Annual Conference on Information Science and Systems (CISS)*, pages 649–653, March 2016.

[16] R. Lemonnier, K. Scaman, and N. Vayatis. Tight bounds for influence in diffusion networks and application to bond percolation and epidemiology. In *Advances in Neural Information Processing Systems*, pages 846–854, 2014.

[17] J. Leskovec, L. A. Adamic, and B. A. Huberman. The dynamics of viral marketing. *ACM Transactions on the Web (TWEB)*, 1(1):5, 2007.

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

[19] D. Lopez-Pintado and D. J. Watts. Social influence, binary decisions and collective dynamics. *Rationality and Society*, 20(4):399–443, 2008.

[20] B. Shulgin, L. Stone, and Z. Agur. Pulse vaccination strategy in the sir epidemic model. *Bulletin of Mathematical Biology*, 60(6):1123–1148, 1998.

[21] C. Wang, W. Chen, and Y. Wang. Scalable influence maximization for independent cascade model in large-scale social networks. *Data Mining and Knowledge Discovery*, 25(3):545–576, 2012.

[22] D. J. Watts. A simple model of global cascades on random networks. *Proceedings of the National Academy of Sciences*, 99(9):5766–5771, 2002.

[23] J. Yang and S. Counts. Predicting the speed, scale, and range of information diffusion in twitter. 2010.