Approximate Algorithms for Data-driven Influence Limitation

Sourav Medya, Arlei Silva, and Ambuj Singh

Abstract—Online social networks have become major battlegrounds for political campaigns, viral marketing, and the dissemination of news. As a consequence, "bad actors" are increasingly exploiting these platforms, which is a key challenge for their administrators, businesses and society in general. The spread of fake news is a classical example of the abuse of social networks by these bad actors. While some have advocated for stricter policies to control the spread of misinformation in social networks, this often happens in detriment of their democratic and organic structure. In this paper, we aim to limit the influence of a target group in a social network via the removal of a few users/links. We formulate the influence limitation problem in a data-driven fashion, by taking into account past propagation traces. More specifically, our algorithms find critical edges to be removed in order to decrease the influence of a target group based on past data. The idea is to control the diffusion processes while minimizing the amount of disturbance in the network structure. Moreover, we consider two types of constraints over edge removals, a budget constraint and also a, more general, set of matroid constraints. These problems lead to interesting challenges in terms of algorithm design. For instance, we are able to show that influence limitation is APX-hard and propose deterministic and probabilistic approximation algorithms for the budgeted and the matroid version of the problem, respectively. Experiments show that the proposed approaches outperform several baselines.

Index Terms—Influence Control, Network Design, Combinatorial Optimization, Approximation

1 Introduction

Online social networks, such as Facebook and Twitter, were popularized as platforms for sharing entertaining content and maintaining friendship. However, they have been quickly transformed into major battlegrounds for political campaigns, viral marketing, and the dissemination of news. With this shift, the increase in the number of "bad actors", such as tyrannical governments and spammers, exploiting these platforms has become a key challenge for their administrators, businesses and society in general.

A questionable approach to control the diffusion of misinformation in social platforms is via stricter laws and regulations. This type of control often happens in detriment of the democratic and organic structure that are central to these platforms. Instead, a more sensible approach is to limit the impact of bad actors in the network while minimizing the disruption of its social structure.

In this paper we formalize *the influence limitation problem*. In particular, we focus on modifying a network via the removal (or blocking) of a few edges or nodes. These modifications can be implemented by social network administrators or induced by other organizations or governments via advertising campaigns. Although we focus on influence limitation, our problem is also relevant from the perspective of an agent that aims to maintain the influence of a set of users. Nodes/edges discovered by our algorithm are those that should be protected by such an agent. Similarly, while we focus on the edge version of the problem, our proposed techniques also apply to the node version.

A well-known major challenge in studying influence is its causal nature. We say that person A influenced person B if A induced B's future behavior [1], [2]. Thus, limiting the influence of A is to reduce the extent in which she can affect the behaviors of others around her. While the most accurate assessment of influence should be based on controlled experiments, running them is often expensive and even controversial [3], [4], [5]. Instead, existing work on influence limitation assume that the diffusion process follows classical models—e.g., Independent Cascade (IC), Linear Threshold (LT), and Susceptible-Infected-Recovered (SIR)—that are based on strong premises and require computationally-intensive simulations [6]. Instead, we propose a data-driven approach for influence minimization based on historical data [7].

Another important limitation of existing studies on influence limitation is that they often assume budget constraints [8], [9]—i.e. a fixed number of nodes or edges can be blocked in the network. However, budget constraints have undesired effects in many settings. For instance, they might disconnect or disproportionately target particular communities in the network. Such effects are in conflict with important modern concerns in algorithm design, such as fairness [10]. We address this issue by studying the influence limitation problem not only under a budget constraint but also under a more general set of matroid constraints [11], [12].

This paper demonstrates how the formalization of the influence limitation problem under budget and matroid constraints leads to interesting challenges in terms of algorithm design. Different from the budget version, for which we propose a simple greedy algorithm, the matroid version requires a more sophisticated solution via continuous relaxation and rounding. Yet, we provide a theoretical analysis of the performance of both algorithms that is supported by the fact that the objective function of the influence limitation problem is submodular. Moreover, we provide strong inapproximability results for both versions of the problem.

Our contributions are summarized as follows:

- We investigate the novel data-driven influence limitation problem via node/edge removals. We show that the edge version is more general and covers the node version of the problem.

- *Sourav Medya is at Kellogg School of Management. E-mail: sourav.medya@kellogg.northwestern.edu*
- *The other authors are with the Dept. of Computer Science, University of California, Santa Barbara. E-mails: {arlei,ambuj}@cs.ucsb.edu*

- We study our problem under both budget and matroid constraints, discussing how these affect algorithmic design.
- We prove that the influence limitation problem is APX-hard and propose constant-factor approximations for both versions of the problem—deterministic and probabilistic aproximation for the budget and matroid version, respectively.
- We show that our methods outperform baseline solutions by up to 35% while scaling to large graphs.

Organization: Section 2 introduces the credit distribution model and the influence limitation problems under two different constraints (budget and matroid). In Sections 3 and 4, we prove the submodularity of the objective function and the APX-hardness of both problems. Sections 5.1 and 5.2 introduce efficient algorithms for influence limitation. In Section 6, we demonstrate the effectiveness of the proposed solutions in an experimental evaluation. Most of the proofs are given in the Appendix.

Symbols	Definitions and Descriptions
$G(V, E)$	Given graph (vertex set V and edge set E)
X	Target set of source nodes
C	The set of candidate edges
k	Budget for BIL
$G(a) = (V(a), E(a))$	Action/propagation graph (DAG) for action a
$\Gamma_{v,u}(a)$	Credit of node v for influencing u in $G(a)$
$\Gamma_{X,u}(a)$	Credit given to set X for influencing u in $G(a)$
$\gamma_e(a) = \gamma_{(v,u)}(a)$	Direct credit for v to influence u via $e = (v, u)$
$u \overrightarrow{d} v$	It implies there is a path from u to v in $G(a)$
$\Delta(B)$	The change in credit due to the removal of set B
b	Maximum #edges removed from a node in ILM
$\vec{y}$	The vector with edge membership probabilities
$f(\vec{y})$	The continuous extension of $\Delta(B)$
τ	#iterations in CG algorithm
s	#samples applied by CG algorithm
$DI(B)$	Decrease in influence after removal of set B

TABLE 1: Frequently used symbols

2 Influence Limitation

We start with a description of Credit Distribution Model and formulate the influence limitation problems in Section 2.2.

2.1 Credit Distribution Model

The Credit Distribution Model (CDM) [7] estimates user influence directly from propagation traces. Its main advantages compared to classical influence models (e.g. Independent Cascade and Linear Threshold [13], [14]) is that it does not depend on computationally intensive simulations while also relying less on the strong assumptions made by such models.

Let $G(V, E)$ and $\mathscr{L}(User, Action, Time)$ be a directed social graph and an action log respectively. A tuple (u, a, t) in action log $\mathscr{L}$ indicates that user u has performed action a at time t. Action $a \in \mathscr{A}$ propagates from node u to node v iff u and v are linked in social graph and u performed action a before v. This process defines a *propagation (action) graph* of a as a directed acyclic graph (DAG) $G(a) = (V(a), E(a))$. The action log $\mathscr{L}$ is thus a set of DAGs representing different actions' propagation traces. For a particular action a, a potential influencer of a node or user can be any of its in-neighbours. We denote $N_{in}(u, a) = \{v|(v, u) \in E(a)\}$ as the set of potential influencers of u for action a and $d_{in}(u, a) = |N_{in}(u, a)|$. When a user u performs action a, the *direct influence credit*, denoted by $\gamma_{(v,u)}(a)$, is given to all $v \in N_{in}(u, a)$. Intuitively the CDM distributes the influence credit backwards in the propagation graph $G(a)$ such that not only u gives credit to neighbours, but also in turn the neighbours pass on the credit to their predecessors. The total credit, $\Gamma_{v,u}(a)$ given to a user v for influencing u via action a from v to u in the propagation graph $G(a)$ is:

$$\Gamma_{v,u}(a) = \sum_{w \in N_{in}(u,a)} \Gamma_{v,w}(a).\gamma_{(w,u)}(a) \tag{1}$$

Similarly, one can define the credit for a set of nodes X,

$$\Gamma_{X,u}(a) = \begin{cases} 1 & if\ u \in X \\ \sum_{w \in N_{in}(u,a)} \Gamma_{X,w}(a).\gamma_{(w,u)}(a) & otherwise \end{cases}$$

The influence $\sigma_{cd}(X)$ is the credit given to X by all vertices:

$$\sigma_{cd}(G, X) = \sum_{u \in V} \frac{1}{|\mathscr{A}_u|} \sum_{a \in \mathscr{A}_u} \Gamma_{X,u}(a) \tag{2}$$

Influence probabilities $\gamma_{(u',v')}$ are computed using well-known techniques [15]. However, our theoretical results do not depend on how γ is computed. We study influence minimization in two settings: The first is budget constrained optimization, where a limit on the number of edges to be modified is set as a parameter. The second takes into account a more general class of constraints that can be expressed using independent sets.

2.2 Problem Definitions

Our goal is to remove a few edges $B \subset E$ such that the influence of a target set of users X is minimized according to the CDM. The credit of target user v for influencing user u in $G(a)$ is computed based on Equation 1. Consider $P(v, u)$ to be the set of paths from v to u where each path $p = \{e_1, e_2, ..., e_t\}$ is such that $e_1 = (v, v')$, $e_t = (u', u)$, and $e_i \in E(a)$ for all i and $u', v' \in V(a) - \{v, u\}$. We use $\gamma_{(w',w)}(a)$ or $\gamma_e(a)$ to represent the credit exclusively via edge $e = (w', w)$ for influencing w in $G(a)$. Therefore, Equation 1 can be written as:

$$\Gamma_{v,u}(a) = \sum_{p \in P(v,u)} \prod_{e \in p} \gamma_e(a) \tag{3}$$

A similar expression can be obtained for a target set X:

$$\Gamma_{X,u}(a) = \sum_{p \in P(X,u)} \prod_{e \in p} \gamma_e(a) \tag{4}$$

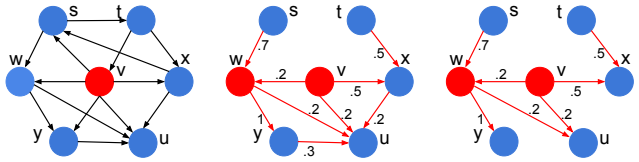
where $P(X, u)$ contains only the minimal paths from $v \in X$ to u—i.e. $\nexists p_i, p_j \in P(X, u)$ such that $p_i \subseteq p_j$.

We apply Equation 4 to quantify the change in credit for a target set of nodes X and a particular action a after the removal of edge e according to the credit distribution model:

$$\delta_a(\{e\}) = \sum_{w \in V} \left(\Gamma_{X,w}(a) - \sum_{\substack{p \in P(X,w) \\ e \notin p}} \prod_{e' \in p} \gamma_{e'}(a) \right) \tag{5}$$

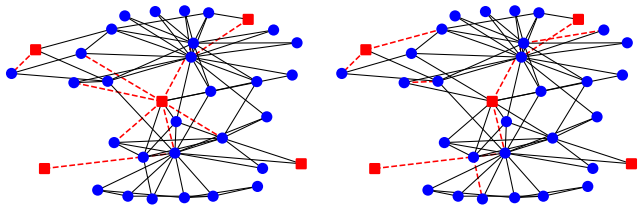
An edge deletion potentially blocks a few paths from v, reducing its credit (or influence). We use $G^m = (V, E - B)$ and $G^m(a)$ to denote the graph and the propagation graph for action a after the removal of edges in B, respectively. The following sections introduce the budget and matroid constrained versions of the influence limitation problem.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2020.3016293, IEEE Transactions on Knowledge and Data Engineering



(a) Social Graph, G (b) $G(a)$ (c) Modified: $G^m(a)$

Fig. 1: Illustrative example of a social graph and CDM with the corresponding credits over the edges.



(a) BIL (b) ILM

Fig. 2: We perform our methods for (a) BIL (Problem 1) and (b) ILM (Problem 2) on the Zachary's Karate network with $|X| = 5, k = 9$. Square (red) nodes are in the target set, $|X|$, and dotted (red) edges are in the solution set. The edges are incident to few nodes in the solution for BIL, being strongly biased towards a small set of nodes. For ILM, we have considered $b = 2$, which leads to a solution with a more uniform set of edges.

2.2.1 Budgeted Influence Limitation (BIL)

We start formalizing the budgeted version of our problem:

Problem 1. ***Budgeted Influence Limitation (BIL):*** *Given a directed graph $G(V, E)$, an action log $\mathscr{L}$, a candidate set of edges C, a given seed set X, and an integer $k < |C|$, find a set $B \subset C \subset E$ of k edges such that $\sigma_{cd}(G^m, X)$ is minimized or, $\Delta(B) = \sigma_{cd}(G, X) - \sigma_{cd}(G^m, X)$ is maximized where $G^m = (V, E \setminus B)$.*

Example 1. *In Figure 1, we assume the candidate set $C = \{(t, x), (y, u), (x, u)\}$, $k = 2$, and $X = \{w, v\}$. In $G(a)$ (Fig. 1b), $\sigma_{cd}(G, X) = 4.21$ and the deletion of $(t, x) \in C$ will not change the influence of X. The removal of (y, u) and (x, u) (Fig. 1c) will make $\sigma_{cd}(G^m(a), X) = \Gamma_{X,v} + \Gamma_{X,w} + \Gamma_{X,x} + \Gamma_{X,u} + \Gamma_{X,y} = 1 + 1 + 0.5 + (0.2 + 0.2) + 1 = 3.9$.*

The BIL problem also generalizes the node version of the influence minimization problem—i.e. where nodes are removed instead of edges. The construction is as follows: Given a node version of the problem, a candidate node u is divided into two nodes u_{in} and u_{out} associated with the incoming and outgoing edges of u, respectively. A directed edge from u_{in} to u_{out} will be added to the candidate edge set in BIL for the corresponding candidate node u in the node version.

Theorem 1. *The BIL problem is NP-hard.*

BIL assumes that any k edges in the candidate set can be removed. As a consequence, an optimal solution for BIL might make the network disconnected or disproportionately affect particular parts of the network. Fig. 2 exemplifies this issue using Zachary's karate[1] network. BIL modifications are strongly biased towards a small set of nodes. To overcome this we define the influence limitation problem under matroid constraints (ILM). Formally it is defined as Problem 2. Next we present a different formulation for influence limitation that addresses some of these challenges.

2.2.2 Influence Limitation under Matroid (ILM)

Matroids are abstract objects that generalize the notion of linear independence to sets [16]. We apply matroids to characterize a class of constraints for influence limitation. To illustrate the expressive power of matroids as a general class of constraints for optimization problems defined over networks, we focus on a particular setting of influence minimization. More specifically, we upper bound the number of incoming edges that can be removed (or blocked) from each node in the network.

Problem 2. ***Influence Limitation under Matroid (ILM):*** *Given a directed social graph $G(V, E)$, an action log $\mathscr{L}$, a candidate set of edges C, a given seed set X, and an integer b, find a set B (where $B \subset C \subset E$) such that at most b edges from B are incident (incoming) on any node in V and $\sigma_{cd}(G^m, X)$ is minimized where $G^m = (V, E - B)$ or, $\Delta(B) = \sigma_{cd}(G, X) - \sigma_{cd}(G^m, X)$ is maximized.*

The effect of ILM enforces network modifications that are more uniformly distributed across the network. Notice that a valid solution for the budget constrained version (BIL) might not necessarily be a valid solution for ILM and vice-versa.

Theorem 2. *The ILM problem is NP-hard.*

It remains to show that ILM follows a matroid—any valid solution is a matroid. We show that ILM follows a *partition matroid*, which is a matroid where the ground set C is partitioned into non-overlapping subsets $C_1, C_2, \cdots, C_l$ with associated integers $b_1, b_2, \cdots, b_l$ such that a set B is independent iff $|B \cap C_i| \leq b_i$.

Observation 1. *ILM follows a partition matroid.*

The key insight for this observation is that, for any incoming edge, the associated node is unique to the edge. As an example, if $e = (u, v)$ then the node v is unique to the edge e. Thus, the ground set C can be partitioned into edge sets $(C_1, C_2, ..., C_{|V|})$ based on the $|V|$ edges associated with them. Any feasible solution B (edge set) is an independent set as $B \cap C_v \leq b$, where $v \in V$. Notice that the more general setting where a constant b_v is defined for each node in the network is also a partition matroid.

The BIL and ILM problems are APX-hard and cannot be approximated a factor greater than $(1 - \frac{1}{e})$. We prove these results in Section 4 using the notion of *curvature* [17].

3 Submodularity

A key feature in the design of efficient algorithms for influence limitation is *submodularity*. Intuitively, submodular functions are defined over sets and have the so called *diminishing returns property*. Besides its more usual application to the budgeted version of our problem, we also demonstrate the power of submodular optimization in the solution of its matroid constrained version.

To prove that the maximization function Δ associated to BIL and ILM is submodular, we analyze the effect of the removal of a candidate edge e over the credit of the target set X. Lemma 3.1 defines the change in credit $(\delta_a(\{e\}))$ in $G(a)$. The notation $u \overrightarrow{a} v$ denotes that there is a path from u to v in $G(a)$.

1. http://www-personal.umich.edu/ mejn/netdata/

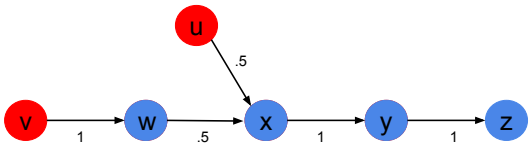


Fig. 3: This illustrates a counter example in Theorem 5.

Lemma 3.1. *For an action a, with corresponding DAG $V(a)$, the change in credit after the removal of $e = (u, v)$ is as follows:* $\delta_a(\{e\}) = \Big(\Gamma_{X,u}(a) \cdot \gamma_{(u,v)}(a)\Big) \cdot \sum_{w \in V} \Gamma_{v,w}(a)$.

We are now able to formalize the change in credit due to a single edge deletion over all the actions in the action set $\mathscr{A}$.

Lemma 3.2. *Let $G^m = (V, E \setminus \{e\})$, then the change in credit $\Delta(\{e\})$ due to the removal of edge e is equal to:*

$$\Delta(\{e\}) = \sum_{a \in \mathscr{A}} \Big(\Gamma_{X,u}(a).\gamma_{(u,v)}(a)\Big).\Big(\sum_{w \in V} \frac{1}{|\mathscr{A}_w|}\Gamma_{v,w}(a)\Big)$$

Lemma 3.2 follows from Lemma 3.1 and Equation 2. Next, we prove the submodularity property of the function Δ. This property helps in designing efficient algorithms for both BIL and ILM.

Theorem 3. *The function Δ is monotone and submodular.*

Proof. The function is monotonic for each action a, as the removal of an edge cannot increase the credit. As a consequence, Δ (sum of credits over all actions) is also monotonic. To prove submodularity, we consider the deletion of sets of edges E_S and E_T where $E_S \subset E_T$, and show that $\Delta(E_S \cup \{e\}) - \Delta(E_S) \geq \Delta(E_T \cup \{e\}) - \Delta(E_T)$ for any edge $e \in C$ such that $e \notin E_S$ and $e \notin E_T$. A non-negative linear combination of submodular functions is also submodular. Thus, it is sufficient to show the property for one action a, as Δ has the following form:

$$\begin{aligned}\Delta(B) &= \sigma_{cd}(G, X) - \sigma_{cd}(G^m, X) \\ &= \frac{1}{|\mathscr{A}_u|} \sum_{a \in \mathscr{A}_u} (\Gamma'_{X,u}(G, a) - \Gamma'_{X,u}(G^m, a))\end{aligned}$$

where $\Gamma'_{X,u}(G, a)$ denotes $\Gamma_{X,u}(a)$ in $G(a)$.

For the same reason, we assume a single node $x \in X$ ($\Gamma_{X,u} = \sum_{s \in X} \Gamma^{V-X+s}_{s,u}$). Edge sets E_S and E_T are removed from the graph and we evaluate $\Delta(\{e\})$ such that $e \notin E_S$ and $e \notin E_T$. Let the credits towards x from node w after removing E_S and E_T edges be $\Gamma'_{x,w}(G^S)$ and $\Gamma'_{x,w}(G^T)$ (omitting a from $\Gamma'(., a)$ for simplicity) respectively. Moreover, we use the notation $u \overrightarrow{a} v$ if there is a path from u to v in $G(a)$. There are two possible cases: (1) If $w \overrightarrow{a} v$ does not hold, then removal of $e = (u, v)$ keeps $\Gamma'_{x,w}(G^S)$ and $\Gamma'_{x,w}(G^T)$ unchanged and the marginal gains for both E_S and E_T are 0. (2) If $w \overrightarrow{a} v$ holds, marginal gains for sets E_S and E_T are equal to $\Gamma'_{x,u}(G^S).\gamma_{(u,v)}.\Gamma'_{v,w}(G^S)$ and $\Gamma'_{x,u}(G^T).\gamma_{(u,v)}.\Gamma'_{v,w}(G^T)$ respectively. Thus,

$$\Gamma'_{x,u}(G^S).\gamma_{(u,v)}.\Gamma'_{v,w}(G^S) \geq \Gamma'_{x,u}(G^T).\gamma_{(u,v)}.\Gamma'_{v,w}(G^T)$$

as $\Gamma'_{x,u}(G^S) \geq \Gamma'_{x,u}(G^T)$ and $\Gamma'_{v,w}(G^S) \geq \Gamma'_{v,w}(G^T)$. These conclude that Δ is a submodular function. □

4 Curvature and APX-hardness

The ILM problem is NP-hard to approximate within a constant greater than $1 - \frac{1}{e}$. We prove the same about the BIL (budget constrained) problem. To show these results, we first describe a parameter named *curvature* that models the dependencies between elements (edges) in maximizing an objective function.

In ILM, the objective is $max\{\Delta(B), B \subset C\}$ where B is an independent set. Before proving APX-hardness, we define the concept of *total curvature* (c_t) [17].

Definition 1. *The total curvature of a monotone and submodular function Δ is defined by:*

$$c_t = 1 - min_{S,e_i} \frac{\Delta(S \cup \{e_i\}) - \Delta(S)}{\Delta(\emptyset \cup \{e_i\}) - \Delta(\emptyset)}$$

The *total curvature* [17] measures how much the marginal gains decrease when an element is added to a set S. Intuitively, it captures the level of dependency between elements in a set S. For instance, if the marginal gains are independent ($c_t = 0$) a simple greedy algorithm will be optimal. Let S^* be the optimal solution set. The *curvature with respect to optimal* (c_o) [17] is as follows:

Definition 2. *Δ has curvature with respect to optimal $c_o \in [0, 1]$ if c_o is the smallest value such that for every T:*

$\Delta(S^* \cup T) - \Delta(S^*) + \sum_{j \in S^* \cap T} \Big(\Delta(S^* \cup T \setminus \{e_i\}) - \Delta(S^* \cup T)\Big) \geq (1 - c_o)\Delta(T)$

Vondrak [17] proves the following theorem:

Theorem 4. *There is no polynomial time algorithm that generates an approximation within a factor larger than $\frac{1}{c_o}(1 - e^{-c_o})$ for maximizing a monotone and submodular function under matroid constraints where c_o is the curvature with respect to optimal.*

We use Theorem 4 to prove the APX-hardness of ILM.

Theorem 5. *ILM is APX-hard and cannot be approximated within a factor greater than $(1 - 1/e)$.*

Proof. ILM is a monotone and submodular optimization problem under a matroid constraint. We prove the inapproximability result by designing a problem instance where the *curvature with respect to optimal* (c_o) is 1. Consider the example in Figure 3, the candidate set $C = \{(w, x), (x, y), (y, z)\}$, $b = 1$ and the target set $X = \{u, v\}$. In this setting, one of the optimal sets $S^* = (w, x), (x, y)$. Assuming $T = (y, z)$ will imply $S^* \cap T = \emptyset$. If $\Delta(S^* \cup T) - \Delta(S^*) = 0$, then c_o has to be 1. Note that, $\Delta(S^* \cup T) = \Delta(S^*) = 2.5$, which leads to $c_o = 1$. Therefore, ILM cannot be approximated within a factor greater than $\frac{1}{1}(1 - e^{-1})$ and our claim is proved. □

Theorem 6. *BIL is APX-hard and cannot be approximated within a factor greater than $(1 - 1/e)$.*

Proof. We reduce the BIL problem from a problem similar to ILM and has matroid constraints with *curvature with respect to optimal* as 1. First, we define this problem as ILMO where maximum b outgoing edges can be deleted form a node (ILM has a limit on incoming edges). However, ILMO is NP-hard, follows matroid constraints and has curvature 1 (the proofs are straightforward and similar to the proofs for ILM). Thus ILMO cannot be approximated within a factor greater than $(1 - \frac{1}{e})$.

Now we give an L-reduction [18] from the ILMO problem. The following two equations are satisfied in our reduction:

$$OPT(I_{BIL}) \leq c_1 \cdot OPT(I_{ILMO})$$
$$OPT(I_{ILMO}) - s(T^S) \leq c_2 \cdot (OPT(I_{BIL}) - s(T^B))$$

where I_{ILMO} and I_{BIL} are problem instances, and $OPT(Y)$ is the optimal value for Y. $s(T^S)$ and $s(T^B)$ denote any solution of the ILMO and BIL instances, respectively. If the conditions hold and BIL has an α approximation, then ILMO has an $(1-c_1c_2(1-\alpha))$ approximation. It is NP-hard to approximate ILMO within a factor greater than $(1-\frac{1}{e})$. Now, $(1-c_1c_2(1-\alpha)) \leq (1-\frac{1}{e})$, or, $\alpha \leq (1-\frac{1}{c_1c_2e})$. So, if the conditions are satisfied, it is NP-hard to approximate BIL within a factor greater than $(1-\frac{1}{c_1c_2e})$.

Consider a problem instance I_{ILMO}, where graph $G = (V, E)$, $|V| = n, |E| = m$ and integer b and the target set $X = \{x\}$ are given. This problem becomes a BIL instance when $b = k$ where k is the budget (in BIL). If the solution of I_{ILMO} is $s(T^S)$ then the influence of node x will decrease by $s(T^S)$. Note that $s(T^B) = s(T^S)$ from the construction. The conditions are satisfied when $c_1 = 1$ and $c_2 = 1$. So, BIL is NP-hard to approximate within a factor grater than $(1-\frac{1}{e})$. □

BIL and ILM are APX-hard and cannot be approximated within a constant greater than $1-\frac{1}{e}$. Given this inaproximability results, we now turn our focus to the design of efficient algorithms for influence limitation, which is the focus of the next section.

5 Algorithms for BIL and ILM

The next two sections describe efficient approximate algorithms for influence limitation based on submodularity.

5.1 An Algorithm for BIL

In Section 4, we have proven that the BIL problem is APX-hard (Theorem 6) and cannot be approximated within a factor greater than $(1-1/e)$ with a polynomial-time algorithm. However, according to Theorem 3, BIL is a monotone submodular maximization problem under a budget constraint. As a consequence, a simple greedy algorithm produces a (tight) constant factor approximation of $(1-1/e)$ [11] for the problem.

We introduce an efficient version of the greedy algorithm based on properties of the credit distribution model. The greedy algorithm removes the edge that minimizes the credit of the target set, one at a time. After each removal, the credit $(\Gamma_{u,v})$ of node u for influencing v has to be updated. However, as only one edge e is removed, intuitively, nodes in the entire network should not be affected but only some in the neighborhood of e. We formalize these observations in an efficient algorithm for BIL. The notation $u \overrightarrow{a} v$ denotes that there is a path from u to v in $G(a)$.

Observation 2. *For a given action a and DAG $G(a)$, the removal of $e = (u, v)$ changes $\Gamma_{z,w}$ iff $z \overrightarrow{a} u$ and $v \overrightarrow{a} w$.*

Let $G(a)$ and $\{z, w\}$ be an arbitrary DAG and node pair, respectively. Deleting $e = (u, v)$ only affects the credit $\Gamma_{z,w}$ i.e. credit of node z for influencing w, if e is on a path from z to w in $G(a)$. The edge e is on one of such paths if and only if $z \overrightarrow{a} u$ and $v \overrightarrow{a} w$. The following observations are derived from Lemma 3.2.

Observation 3. *For given DAG $G(a)$, the removal of $e = (u, v)$ reduces $\Gamma_{z,w}$ by $(\Gamma_{z,u} \cdot \gamma_{(u,v)}) \cdot \Gamma_{v,w}$ iff $z \overrightarrow{a} u$ and $v \overrightarrow{a} w$.*

Algorithm 1: Greedy

Require: X, C, k
Ensure: A solution set B of k edges
1: $B \leftarrow \emptyset$
2: **while** $|B| \leq k$ **do**
3: **for** $e \in C \setminus B$ **do**
4: *e.MC*← computeMC(e)
5: $e^* \leftarrow \arg\max_{e \in C \setminus B}\{e.MC\}$
6: $B \leftarrow B \cup \{e^*\}$ and $E \leftarrow E \setminus \{e^*\}$
7: updateUC(e, EP, UC, SC)
8: updateSC(e, EP, UC, SC)
9: **return** B

Algorithm 2: computeMC

Require: $e = (u, v)$, X, UC, SC
Ensure: mc
1: $mc = 0$
2: **for** $a \in \mathscr{A}$ such that $SC[u][a] > 0$ & $EP[u][v][a] > 0$ **do**
3: $mc_a = 0$
4: **for** each user w such that $UC[v][w][a] > 0$ **do**
5: $mc_a = mc_a + UC[v][w][a]/\mathscr{A}_w$
6: $mc = mc + (SC[u][a] \cdot EP[u][v][a]) \cdot mc_a$

Observation 4. *For given target set X, an action a and DAG $G(a)$, the removal of $e = (u, v)$ reduces $\Gamma_{X,w}$ by $(\Gamma_{X,u} \cdot \gamma_{(u,v)}) \cdot \Gamma_{v,w}$ iff $z \overrightarrow{a} u$ and $v \overrightarrow{a} w$ where $z \in X$.*

Algorithm 1 scans the action log $\mathscr{L}$ to collect information for comparing the effect of each candidate edge. This information is maintained in data structures EP, EC, and SC. More specifically, $EP[u][v][a]$ denotes the edge credit $(\gamma_{(u,v)}(a))$ of u for influencing v when (u, v) exists, $UC[u][v][a]$ is the credit $(\Gamma_{u,v}(a))$ given to u for influencing v, and $SC[u][a]$ is the credit $(\Gamma_{X,u}(a))$ given to X for influencing u, all for an action a.

The contribution of each edge (see Lemma 3.2), given the current solution B, is computed using *computeMC*. Method *updateUC* (Algorithm 4) identifies the credits that have been changed upon an edge removal and does so by updating the data structure UC following Observation 3. Method *updateSC* does the same for the credits of target set of nodes X by updating the data structure SC following Observation 4. The methods *updateUC* and *updateSC* are described in the Appendix.

The expensive steps of Algorithm 1 are steps 4, 7 and 8. The corresponding methods *computeMC*, *updateUC*, and *updateSC*, take $O(\sum_{a \in \mathscr{A}} |V(a)|), O(\sum_{a \in \mathscr{A}} |V(a)|^2)$, and $O(\sum_{a \in \mathscr{A}} |V(a)|)$ time respectively. Thus, the total running time of Greedy is $O(k \cdot |C| \cdot \sum_{a \in \mathscr{A}} |V(a)| + k \cdot \sum_{a \in \mathscr{A}} |V(a)|^2)$. Notice that the total time complexity does not depend on the number of nodes $(|V|)$ in the graph, but on the sizes of the action graphs, the budget and the candidate edge set.

5.2 Method: ILM

Theorems 2 and 5 show that the ILM problem is NP-hard (Section 2) and APX-hard (Section 4), respectively. As was the case for BIL, these inapproximability results imply that there does not exist a polynomial-time algorithm for ILM that generates results that are within a factor greater than $(1-1/e)$ of the optimal. Moreover, notice that the greedy algorithm (Algorithm 1) for BIL might

Algorithm 3: Continuous Greedy (CG)

Require: X, C, b

Ensure: A vector $\vec{y}$ satisfying constraints **(S1)** and **(S2)**

1: Start $\vec{y}$ as a null vector, $t = 0$
2: **while** $t \leq \tau$ **do**
3: Generate s samples $B_1, B_2, ..., B_s$ where e_i belongs to B_j ($\forall j \in [s]$) with probability y_i
4: Set weight of an edge, e_i as $w_i = \frac{\sum_{j=1}^{s} \Delta(B_j \cup \{e_i\}) - \Delta(B_j)}{s}$
5: Compute an edge set E^Y maintaining the constraint **(S2)** and maximizes $\sum_{e_i \in E^Y} w_i$
6: For all $e_i \in E^Y$, set $y_i = y_i + 1/\tau$
7: $t = t + 1$
8: **return** $\vec{y}$

not provide a valid solution for the matroid constrained problem. Based on Observation 1, we apply continuous relaxation and adopt the continuous greedy technique to design our algorithm. We also propose a fast randomized scheme for rounding the relaxed solution that works well in practice.

5.2.1 Continuous Relaxation

Let $\vec{y} = (y_1, y_2, ...y_c)$ be the vector with membership probabilities for edges in the candidate set C ($|C| = c$). Moreover, let B be a random subset where $e_i \in C$ is included in B with probability y_i. From [19], if f is the continuous extension of Δ, then:

$$f(\vec{y}) = \mathbf{E}_{B \sim \vec{y}}[\Delta(B)] = \sum_{B \subseteq C} \Delta(B) \prod_{e_i \in B} y_i \prod_{e_i \in C \setminus B} (1 - y_i) \tag{6}$$

Let $E_{in}(v)$ be the edges incoming to v. Our objective is to find a $\vec{y}$ that maximizes $f(\vec{y})$ with the following constraints:

$$y_i \in [0, 1] \tag{7}$$

$$\sum_{e_i \in E_{in}(v)} y_i \leq b \quad \forall v \in V \tag{8}$$

While Equation 7 (constraint **S1**) maintains the fractional values as probabilities, Equation 8 (constraint **S2**) enforces the maximum number of edges incident to each node to be bounded by b. Because the relaxation of Δ as f is continuous, the optimal value for f is an upper bound on Δ (the discrete version). Let B^* and Y^* be the optimal edge sets for Δ and f, respectively. Also, let Z be a vector defined as follows: $z_i = 1$ if $e_i \in B^*$ and $z_i = 0$, otherwise. Then, $\Delta(B^*) = f(Z)$ and Z maintains the constraints. As $f(Y^*)$ is maximum, $\Delta(B^*) = f(Z) \leq f(Y^*)$.

We show that the function f is smooth (it has a second derivative), monotone and submodular. Based on these properties, we design a continuous greedy algorithm that produces a relaxed solution for ILM with a constant-factor approximation [19].

Theorem 7. *The function f is smooth, monotone and submodular.*

Continuous Greedy (CG): The continuous greedy algorithm (Algorithm 3) provides a solution set $\vec{y}$ such that $f(\vec{y}) \geq (1 - \frac{1}{e}) f(Y^*) \geq (1 - \frac{1}{e})\Delta(B^*)$ with high probability. The guarantee exploits the facts that Δ is submodular (Theorem 7) and ILM follows a matroid constraint (Observation 1). CG is similar to the well-known Frank-Wolfe algorithm [20]. It iteratively increases the coordinates (edge probabilities) towards the direction of the best possible solution with small step-sizes while staying within the feasible region. In [19], the author proves the following:

Theorem 8. *The Continuous Greedy (Algorithm 3) returns a vector $\vec{y}$ that satisfies constraints S1 and S2 and such that $f(\vec{y}) \geq (1 - \frac{1}{e})\Delta(B^*)$ when $\tau = c^2$ and $s = c^5$.*

The values τ and s correspond to the number of iterations and samples applied by CG. The costliest operations of the algorithm are steps 3, 4 and 5. Step 3 takes $O(c.s)$ time, as it visits each edge in the candidate set C ($|C| = c$). Step 4 computes the contribution of edges, having worst case time complexity $O(s.c \sum_{a \in \mathscr{A}} |V(a)|)$. Step 5 greedily selects the best edges, according to the weights. Therefore, the total running time of the algorithm is $O(\tau(s.c \sum_{a \in \mathscr{A}} |V(a)| + c \log c))$. Though Theorem 8 requires high value of τ and s, in practice, the algorithm produces good results with low values of them (see Section 6.2).

5.2.2 Rounding

Algorithm 3 returns a vector $\vec{y}$ satisfying constraints Eq. 7 and Eq. 8 while producing $f(\vec{y}) \geq (1 - \frac{1}{e})\Delta(B^*)$. However, as $\vec{y}$ contains probability values (in the interval $[0, 1]$), a rounding step is still required for obtaining a deterministic set of edges.

There exists a computationally-intensive lossless rounding procedure for matroids known as *swap rounding* [16]. The computation depends on the number of base matroids which can be very large in the solution obtained from Algorithm 3. We address this issue by proposing a simpler and faster randomized procedure. We show that our independent rounding method produces feasible edges with low error and high probability.

We sort the edges according to their weights (probabilities) and round them while maintaining feasibility. This procedure only makes a single pass over the candidate edges in C. In order to analyze this randomized procedure, we assume that it is unaware of the dependency between the edges. Let $\mathcal{B}$ be the edge set produced by rounding, i.e. $f(\vec{y}) = \mathbf{E}[\Delta(\mathcal{B})]$, and let $E_v \subset \mathcal{B}$ be the incoming edges incident on node v. Next we show that the randomized procedure produces a feasible set within error ϵ with (high) probability $1 - \frac{1}{n}$, where $n = |V|$ is the number of nodes.

Theorem 9. *The following holds for the number of edges incoming to v in the rounded set:*

$$Pr(|E_v| < (1 + \epsilon)b) \geq 1 - \frac{1}{n}$$

where $\epsilon = \sqrt{\frac{6 \log n}{b}}$.

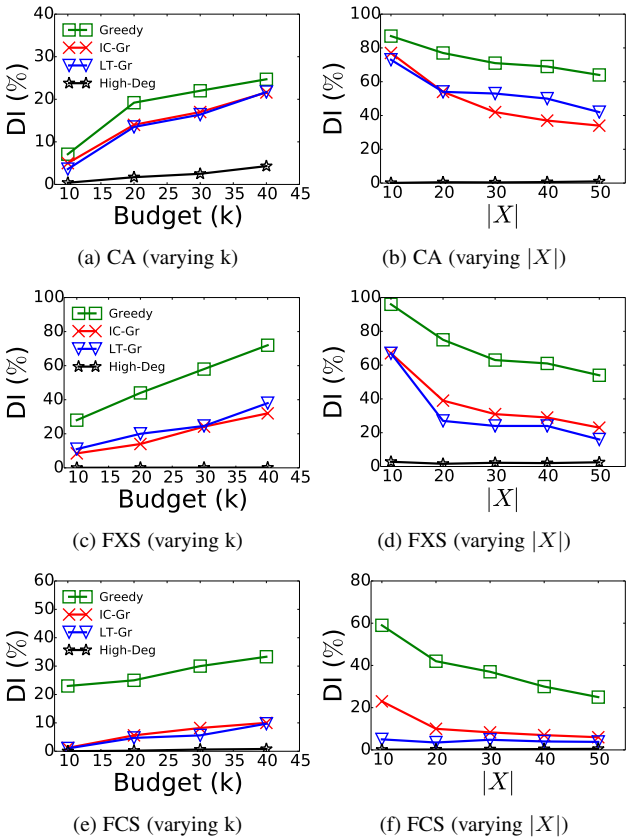
We emphasize two implications of Theorem 9: (1) The probability that the rounded solution is feasible depends on the error ϵ which is small whenever b is large; (2) The rounding procedure has a probabilistic bi-criteria approximation, being lossless if the maximum number of edges to be removed per node is $b' = b(1 + \epsilon)$. The proposed randomized rounding scheme is efficient, as it only performs one pass over the candidate edges C. Additionally, node degrees on average are not very large compared to the constraint b for a node. Moreover, notice that our framework also works with different b's for each node.

5.2.3 Generalizations

Matroids can capture other influence limitation settings, especially when edges in the solution can be naturally divided into partitions. Examples include the limitation of influence in non-overlapping communities [21], [22], disjoint campaigning [23], and problems

Dataset Name	$\|V\|$	$\|E\|$	#Action	#Tuple
ca-AstroPh (CA)	$18K$	$197K$	$1K$	$56K$
email-EuAll (EE)	$265K$	$420K$	–	–
Youtube (CY)	$1.1M$	$2.9M$	–	–
Flixster-small (FXS)	$15K$	$191K$	$1.8K$	$30K$
Flickr-small (FCS)	$15K$	$1.4M$	$1.4K$	$10K$
Flixster (FX)	$1M$	$28M$	$49K$	$8.2M$
Flickr (FC)	$1.3M$	$81M$	$296K$	$36M$

TABLE 2: Statistics of the datasets. We generate synthetic actions via IC model for CA, EE and CY datasets.



(a) CA (varying k) (b) CA (varying $|X|$)

(c) FXS (varying k) (d) FXS (varying $|X|$)

(e) FCS (varying k) (f) FCS (varying $|X|$)

Fig. 4: [BIL] (a, c, e) Decrease in Influence (DI) produced by different algorithms. (b, d, f) DI produced by different algorithms varying the size of the target set, X with $k = 30$.

where issues of fairness arise [24]. Moreover, influence boosting problems via attribute-level modification [25] and edge addition [26] can also be modelled under matroid constraints.

6 Experimental Results

Our solutions were implemented in Java and experiments were conducted on 3.30GHz Intel core with 30 GB RAM.

Datasets: The datasets used in the experiments are the following: 1) **Flixster [7]**: Flixster is an unweighted directed social graph, along with the log of performed actions. The log has triples of (u, a, t) where user u has performed action a at time t. Here, an action for a user is rating a movie. 2) **Flickr [27]**: This is a photo sharing platform. Here, an action would be joining an interest group. 3) **Synthetic**: We use the structure of real datasets that come from different genre (e.g., co-authorship, social). The networks are available online[2]. We synthetically generate the actions and create associated tuples. Synthetic actions are generated assuming the Independent Cascade (IC) [13] model. The

2. https://snap.stanford.edu

	FXS: # (tuples, actions)$\times 10^3$		
Budget	$(30, 1.7)$	$(50, 4.8)$	$(75, 6.9)$
$k = 50$	58	61	68
$k = 75$	73	83	85
$k = 100$	85	88	91
	FCS: # (tuples, actions)$\times 10^3$		
	$(20, 2.6)$	$(30, 3.8)$	$(50, 5.8)$
$k = 50$	208	383	1187
$k = 75$	269	579	1891
$k = 100$	356	780	2551

TABLE 3: [BIL] Running times (in secs.) of Greedy varying number of tuples. Number of tuples and actions are in thousands.

"ca-AstroPh" dataset is a Collaboration network of Arxiv Astro Physics. In the "Youtube" social network, users form friendship with others and can create groups that other users can join. Table 2 shows the statistics of the datasets. We use the small networks (from Flixster and Flickr) for the quality-related experiments as our baselines are not scalable. To show the scalability of our methods, we extract networks of different sizes from the raw large Flixster and Flickr data. For all the networks, we learn the influence probabilities via the method proposed in [15].

Performance Metric: The quality of a solution set B (a set of edges) is the percentage of *Decrease in Influence (DI)* of X:

$$DI(B) = \frac{(\sigma_{cd}(G, X) - \sigma_{cd}(G^m, X))}{\sigma_{cd}(G, X)} \times 100 \qquad (9)$$

The set X is randomly selected from the set of top 150 nodes with highest number of actions. The candidate set C contains edges that appear at least once in any action graph. The number of MC simulations for IC and LT-based baselines is at least 1000.

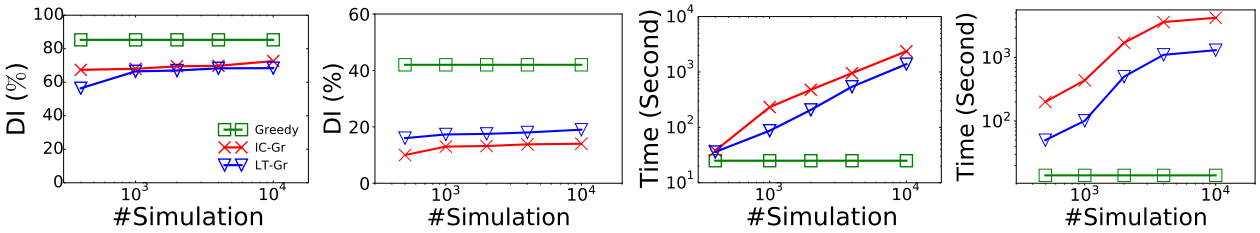
6.1 Experiments: BIL

Baselines: 1) **IC-Gr [28]:** Finds the top k edges based on a greedy algorithm that minimizes influence under the IC model. 2) **LT-Gr [26]:** Identifies the top k edges based on the greedy algorithm proposed in [26], which minimizes the influence of a set of nodes according to the LT model. Note that we also apply the optimization techniques proposed in [26] for both of these baselines. 3) **High-Degree:** Selects edges between the target nodes X and the top-k high degree nodes. Other heuristics (*Friends of a Friend* and random selection) did not produce better results than High-Deg. The mentioned baselines have limitations compared to Algorithm 1. High-degree is a simple heuristic that ignores the combinatorial nature of the BIL objective. IC-Gr [28] and LT-Gr [26] assume theoretical models for influence and do not take the propagation traces into account to compute marginal gains of the edges—i.e. they are not data-driven.

Quality (vs Baselines): We compare our Greedy algorithm (Algorithm 1) against the baselines on three datasets (CA, FXS and FCS) in Figures 4a, 4c and 4e (target size is set as 30). Greedy takes a few seconds to run and significantly outperforms the baselines (by up to 35%) in terms of DI(%). The running time of Greedy is low as it avoids expensive Monte-Carlo simulations. For CA, the action graphs are generated by IC model and hence, IC-Gr produces better results than the other two datasets.

Scalability of Greedy: We show the scalability of our Greedy algorithm (Alg. 1) for increasing number of tuples (#actions) size of the graph. Table 3 shows the results on FXS and FCS. As FCS has higher edge density than FXS, the number of tuples has higher effect on the running time in FCS. Note that we consider

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2020.3016293, IEEE Transactions on Knowledge and Data Engineering



(a) Quality on CA (b) Quality on FXS (c) Time on CA (d) Time on FXS

Fig. 5: [BIL] Comparison of greedy and baselines varying number of simulations: (a-b) Quality and (c-d) Running time.

Dataset	$\|V\|$	Actions	Tuples	$\|C\|$	Time (sec)
EE	265K	5K	326K	4.1K	637
CY	1.1M	5K	313K	6.3K	950
FX	200K	2.6K	200K	51K	4020

TABLE 4: **[BIL]** Running Times of Greedy varying graph size for $|X| = 30$ and $k = 30$.

all the edges that appear in one of the actions in our candidate set of edges. A larger candidate set results in longer running time. However our algorithm only takes around 2 and 43 minutes to run for $75K$ and $50K$ tuples in FXS and FCS, respectively.

Table 4 shows the results varying the graph size. The running times are dominated by the size of both the graphs and the candidate sets. Greedy takes nearly 16 minutes on CY with 1M nodes and 6K candidate edges, whereas, it takes 67 minutes on FX with 200K nodes and 51K candidate edges.

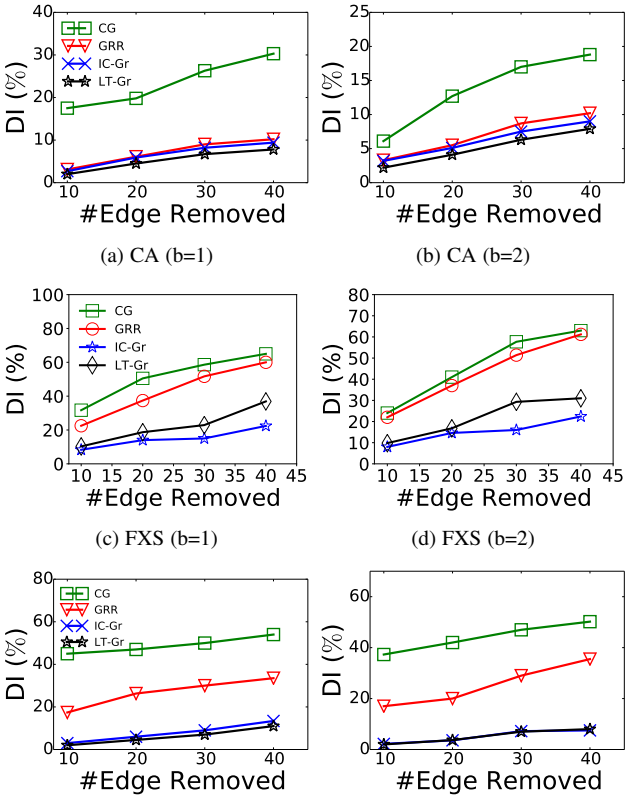
Parameter variations: We also analyze the impact of varying the following parameters: the number of target nodes ($|X|$) and the number of simulations for LT-Gr and IC-Gr. First we vary the size of the target set X. Figures 4b, 4d, and 4f show the results for CA, FXS, and FCS respectively where budget, $k = 30$. Greedy provides better DI (by up to 35%) across all $|X|$ and datasets. With the increase in $|X|$, DI decreases for the top three algorithms. A larger $|X|$ would have a higher influence to reduce. Thus, with the same number of edges removed, DI would decrease for larger target set. DI is lower for FCS as it is denser than CA and FXS.

We also evaluate the effect of the number of simulations on LT-Gr and IC-Gr. Figure 5 shows the results for $|X| = 30$ and $k = 20$. Our algorithm produces better results even when the baselines perform 10^4 simulations. By comparing Figures 5a and 5b, it is evident that IC-Gr performs better than LT-Gr in CA as the synthetic actions are generated by IC model. Figure 5d also shows that our method is 1-4 orders faster than the baselines.

6.2 Experiments: ILM

Baselines and other settings: 1) **Greedy with Restriction (GRR):** Finds the feasible edges (respecting the matroid constraint) using a greedy algorithm (BIL). We also apply **IC-Gr** and **LT-Gr** with the edge removal constraint for each node.

Algorithm 3 (CG) applies a continuous relaxation and a matching procedure that respects the matroid constraint while searching for a solution for ILM. On the other hand, GRR selects the best edge in each iteration greedily respecting the constraint, without taking into account future edge selections. **IC-Gr** and **LT-Gr** suffer from the same problem, in addition to the limitations described in Section 6.1. The number of samples and iterations used in CG are set to $s = 20$ and $\tau = 100$, respectively. After obtaining the solution vector from CG, we the run randomized rounding scheme for 50 times and return the best solution found.



(a) CA (b=1) (b) CA (b=2)

(c) FXS (b=1) (d) FXS (b=2)

(e) FCS (b=1) (f) FCS (b=2)

Fig. 6: **[ILM]** Decrease in Influence produced by different algorithms on (a-b) CA, (c-d) FXS, and (e-f) FCS. Our algorithm, CG outperforms the baselines by up to 20%.

	FCS: # (tuples, actions)$\times 10^3$		
	(20, 2.6)	(30, 3.8)	(50, 5.8)
20	28.1	64.7	180
40	29.1	64.6	181
60	29.2	63.1	167

TABLE 5: **[ILM]** Running Times (in min.) of CG varying number of tuples for $|X| = 20$ and $b = 2$.

Quality (vs Baselines): We compare the Continuous Greedy (CG) algorithm against the baselines on CA, FXS and FCS where $|X| = 30$ varying the number of edge removal. Figure 6 shows the results when $b = 1$ and $b = 2$. CG significantly outperforms the baselines by up to 20%. GRR does not produce good results as it has to select the feasible edge that does not violate the maximum edge removal constraint b. While maintaining feasibility, GRR cannot select the current true best edge.

Scalability: CG (Algorithm 3) is generally slower than GRR. However, unlike for GRR, increasing the budget does not affect

	FCS: # (tuples, actions)$\times 10^3$					
#Edge	$(20, 2.6)$		$(30, 3.8)$		$(50, 5.8)$	
	CG	**GRR**	**CG**	**GRR**	**CG**	**GRR**
20	33	22	50	41	35	20
40	42	32	53	44	45	35
60	44	35	61	54	54	40

TABLE 6: **[ILM]** Decrease in Influence ($\%$) in FCS by Continuous Greedy (CG) vs GRR varying the number of tuples.

the running time of CG. We evaluate the running time of CG while increasing the number of tuples (actions) using FCS in Table 5. Table 6 shows the results on the quality in DI ($\%$) produced by CG and GRR (other baselines are not scalable) on FCS data. CG outperforms GRR by up to 15%.

Table 7 shows results varying the graph size. Running times are dominated by the size of the graphs and the candidate sets (numbers of actions and tuples are in Table 4). CG takes approximately 31 minutes on CY with 1M nodes and 6K candidate edges, whereas it takes approximately 1.5 hours on FX with 200K nodes and 51K candidate edges. CG also outperforms GRR by up to 9%.

Dataset	$\|V\|$	$\|C\|$	CG (Time)	CG (DI)	GRR (DI)
CY	1.1M	6.3K	1858	55.1	47.2
FX	200k	51K	5690	46.2	37.3

TABLE 7: **[ILM]** Running Time (Scalability) in seconds of CG and DI (percentage) by CG and GRR varying graph size for $|X| = 20$, $b = 2$ and the number of edges removed is 20.

7 Previous Work

The influence boosting or limitation problems via network modifications are orthogonal to the classical influence maximization task [13]. In these problems, the objective is to optimize the content spread via structural or attribute-level change in the network.

Previous work addressed the influence limitation problem in the SIR model [8], [29], [30]. The objective is to optimize specific network properties to boost or contain the content/virus spread. For instance, Tong et al. proposed methods to add (delete) edges to maximize (minimize) the eigenvalue of the adjacency matrix. A more recent work [31] proposes measuring the influence of an edge or a node over a rank of nodes (e.g. Pagerank). However, notice that our measure of influence is based on the CDM.

The influence spread optimization problem also has been studied under the IC model via network design [25], [28], [32], [33], [34] and injecting an opposite campaign [35], [36]. Bogunovic [32] addressed the minimization problem via node deletion. On the other hand, the node addition problem was solved via mixed mixed integer programming [33] and later by greedy algorithms [28]. While Chaoji et al. [34] studied the problem of boosting the content spread via edge addition, Lin et al. [25] investigated the same via influencing initially uninfluenced users.

Boosting and controlling the influence via edge addition and deletion were also studied under the Linear Threshold (LT) model by Khalil et al. [26]. They showed the supermodular property for the objective functions and then applied known approximation guarantees. The influence minimization problem was also studied under a few variants of LT model. [9], [37].

In summary, existing approaches for optimizing influence (propagation) are mostly based on theoretical diffusion models such as SIR, LT and IC. However, our work addresses the influence minimization problem based on historical cascade information. Moreover, we frame influence limitation under two different types of constraints—budget and matroid. Another contribution of our work is to show that our algorithms produce tight guarantees based on inapproximability results (see Theorems 5 and 6).

Optimization over matroids: Matroids are powerful mathematical framework for expressing constraints for combinatorial problems [11], [12]. Nemhauser [11] introduced a few optimization problems under matroids. Vondrak [19] addressed matroid optimization with a continuous greedy technique for submodular functions. Calinescu et al. [38] and Chekuri et al. [16] proposed rounding techniques for continuous relaxation of submodular functions under matroids. Here, we apply some of these theoretical results to develop an efficient approximate algorithm for influence limitation, which is a novel problem.

Other network modification problems: Lin et al. [39] addressed a shortest path optimization problem via improving edge weights on undirected graphs. Single-source and all-pair shortest path distance minimization were also studied in [40], [41] along with node versions [42], [43]. Previous literature have also studied optimization of node centrality [44], [45], [46], [47], node similarities [48] and the stability of k-core [49], [50] via network design. However, these optimization problems have objective functions that are different from the one considered in our work.

8 Conclusions

We studied the influence limitation problem via edge deletion. Different from previous work, our formulation is data-driven, taking into account available propagation traces in the selection of edges. Our influence limitation problem was framed under two different types of constraints—budget and matroid. Both versions were shown to be APX-hard and cannot be approximated within a factor greater than $(1 - \frac{1}{e})$. For the budget constrained version, we have developed an efficient greedy algorithm that achieves a good approximation guarantee by exploiting the monotonicity and submodularity of the objective function. The matroid constrained version was solved via continuous relaxation and a continuous greedy technique, achieving a probabilistic approximation guarantee. Experiments showed the effectiveness of our solutions, which outperform the baselines using both real and synthetic datasets.

This work opens several lines for future research. First, while we applied the Credit Distribution Model to measure the effect of edge removals based on propagation traces, it would be interesting to consider probabilistic models for influence (e.g. [51]). Moreover, we would like to investigate influence limitation under theoretical frameworks for causality [52], [53]. Finally, instead of assuming a single actor (influence minimizer), a more realistic and complex setting should take into account also an active influence maximizer as an adversary in a game-theoretic setting [54], [55].

Appendix A

A.1 Proof of Theorem 1

Proof. We show a reduction from the known *Influence Maximization* (IM) problem [7] under the credit distribution model (CDM) to our problem (BIL). Consider a problem instance I_{IM} [7], where graph $G = (V, E)$, $|V| = n, |E| = m$ and integer k are given. We create a corresponding BIL problem instance (I_{BIL}) as follows. The directed social graph is $G' = (V', E')$ where $V' = V \cup \{x\}$, x is an additional node. Let $C = \{(x, v)|v \in V\}$. In I_{BIL}, $E' = E \cup C$. We assume that the edges in the candidate set C are present for every action in IM. Let us assume the set

Algorithm 4: updateUC

Require: $e = (u, v)$, EP, UC, SC

for $a \in \mathscr{A}$ **do**
  $\gamma \leftarrow EP[u][v][a]$
  for each user z such that $UC[z][u][a] > 0$ **do**
    for each user w such that $UC[v][w][a] > 0$ **do**
      $UC[z][w][a] = UC[z][w][a] - (UC[z][u][a] \cdot \gamma) \cdot UC[v][w][a]$

Algorithm 5: updateSC

Require: $e = (u, v)$, EP, UC, SC

for $a \in \mathscr{A}$ such that $SC[u][a] > 0$ & $EP[u][v][a] > 0$ **do**
  $\gamma \leftarrow EP[u][v][a]$
  for each user w s.t. $UC[v][w][a] > 0$ **do**
    $SC[w][a] = SC[w][a] - (SC[u][a] \cdot \gamma) \cdot UC[v][w][a]$

S (of size k) has the maximum influence (σ^*). The maximum reduction of the influence of node x in BIL can be obtained if and only if the edges (k edges) between x and S are removed. □

A.2 Proof of Lemma 3.1

Proof. For $v \overrightarrow{d} w$ we use induction on length l. Let the set of reachable nodes via a path length l from v in $G(a)$ be $R^a(v, l)$. We denote $N_{out}(u, a) = \{v | (u, v) \in E(a)\}$ and the decrease in credit contribution via removing an edge e by any arbitrary node w in $R^a(v, l)$ as $\delta_a^{l,w}(\{e\})$ and by all nodes in $R^a(v, l)$ as $\delta_a^l(\{e\})$.

Base case: When $l = 0$, $\sum_{w \in V} \Gamma_{v,w}(a, 0) = \Gamma_{v,v} = 1$.

Induction step: Assume that the statement is true when restricted to path lengths l, for any arbitrary node w where $w \in R^a(v, l)$, i.e., $\delta_a^{l,w}(\{e\}) = \Big(\Gamma_{X,u}(a).\gamma_{(u,v)}(a)\Big).\Gamma_{v,w}(a, l)$

$$\delta_a^l(\{e\}) = \Big(\Gamma_{X,u}(a).\gamma_{(u,v)}(a)\Big). \sum_{w \in R^a(v,l)} \Gamma_{v,w}(a) = \sum_{w \in R^a(v,l)} \delta_a^{l,w}(\{e\})$$

We will prove that the statement remains true for paths of length $l + 1$ for nodes $w \in R^a(v, l + 1)$. Now in RHS,

$$\begin{aligned} &\sum_{w \in R^a(v,l+1)} \big(\Gamma_{X,u}(a).\gamma_{(u,v)}(a)\big).\Gamma_{v,w}(a, l+1) \\ &= \sum_{w \in R^a(v,l+1)} \big(\Gamma_{X,u}(a).\gamma_{(u,v)}(a)\big). \sum_{y \in N_{in}(w)} \Gamma_{v,y}(a, l).\gamma_{(y,w)}(a) \\ &= \sum_{y \in R^a(v,l)} \big(\Gamma_{X,u}(a).\gamma_{(u,v)}(a)\big).\Gamma_{v,y}(a, l). \sum_{w \in N_{out}(y)} \gamma_{(y,w)}(a) \\ = &\sum_{y \in R^a(v,l)} \delta_a^{l,y}(\{e\}). \sum_{w \in N_{out}(y)} \gamma_{(y,w)}(a) = \sum_{w \in R^a(v,l+1)} \delta_a^{l+1,w}(\{e\}) \end{aligned}$$

□

A.3 Algorithms 4 and 5 (updateUC and updateSC):

UpdateUC (Algorithm 4) identifies the credits (of users) that change upon an edge removal and does so by updating the data structure UC following the Observation 3. Method *updateSC* does the same for the credits of the target set of nodes (X) by updating the data structure SC following the Observation 4.

	FXS: # (tuples, actions)$\times 10^3$					
#Edge Removed	(30, 1.7)		(50, 4.8)		(75, 6.9)	
	CG	**GRR**	**CG**	**GRR**	**CG**	**GRR**
20	50	44	48	42	51	44
40	51	47	53	45	60	56
60	60	54	61	55	63	57

TABLE 8: **[ILM]** Decrease in Influence (%) in FXS by Continuous Greedy (CG) vs GRR varying the number of tuples. The number of tuples and actions are in thousands.

	FXS: # (tuples, actions)$\times 10^3$		
#Edge Removed	(30, 1.7)	(50, 4.8)	(75, 6.9)
20	7.5	20.7	69.4
40	7.1	16.8	69.4
60	7.4	16.7	69.5

TABLE 9: **[ILM]** Running times (in minutes) of CG varying number of tuples for $|X| = 20$ and $b = 2$ on FXS.

A.4 Proof of Theorem 7

Proof. Let $\mathcal{Y} = (y_1, y_2, ...y_c)$ be the vector with membership probabilities for each edge in C ($c = |C|$). Let the set B be a random subset of C where the edge $e_i \in C$ is included in set B with probability y_i. If f is the continuous extension of Δ, then, $f(\mathcal{Y}) = \mathbf{E}_{B \sim \mathcal{Y}}[\Delta(B)] = \sum_{B \subseteq C} \Delta(B) \prod_{e_i \in B} y_i \prod_{e_i \in C \setminus B} (1 - y_i)$. To prove the function $f : [0, 1]^C \rightarrow \mathbb{R}$ is a smooth monotone submodular function, we need the followings: (i) f has second partial derivatives everywhere. (ii) Monotonicity: For each $e_i \in C$, $\frac{\partial f}{\partial y_i} \geq 0$. (iii) Submodularity: For each $e_i, e_j \in C$, $\frac{\partial^2 f}{\partial y_i \partial y_j} \geq 0$. We derive a closed form similar in [19] for the second derivative and thus it always exists. For each $e_i \in C$,

$$\frac{\partial f}{\partial y_i} = \mathbf{E}[\Delta(B)|e_i \in B] - \mathbf{E}[\Delta(B)|e_i \notin B]$$

As Δ is monotone, $\mathbf{E}[\Delta(B)|e_i \in B] - \mathbf{E}[\Delta(B)|e_i \notin B] \geq 0$ and thus, f is also monotone. For each $e_i, e_j \in C, i \neq j$,

$$\begin{aligned} \frac{\partial^2 f}{\partial y_i \partial y_j} = &\mathbf{E}[\Delta(B)|e_i, e_j \in B] - \mathbf{E}[\Delta(B)|e_i \in B, e_j \notin B] - \\ &\mathbf{E}[\Delta(B)|e_i \notin B, e_j \in B] - \mathbf{E}[\Delta(B)|e_i, e_j \notin B] \end{aligned}$$

As Δ is submodular, $\frac{\partial^2 f}{\partial y_i \partial y_j} \geq 0$ from the above expression. Thus, f is submodular. Note that if $i = j$, $\frac{\partial^2 f}{\partial y_i \partial y_j} = 0$. □

A.5 Proof of Thorem 9

Proof. Let $\mathcal{B}$ be the set of edges produced by the rounding procedure. An edge e_i is included in $\mathcal{B}$ with probability y_i. As $\vec{y}$ is a feasible solution, $\sum_{e_i} y_i \leq b \quad \forall v \in V$ (Equation 8) where e_i is incident (incoming) to vertex v. Thus, $\mathbf{E}(|E_v|) \leq b$. Applying the Chernoff's bound:

$$Pr(|E_v| \geq (1 + \epsilon)\mathbf{E}(|E_v|)) < \exp\left(-\frac{\mathbf{E}(|E_v|)\epsilon^2}{3}\right)$$

Applying the union bound, $\forall v \in V$, we get:

$$Pr(|E_v| \geq (1 + \epsilon)b) < n \cdot \exp\left(-\frac{b\epsilon^2}{3}\right)$$

Substituting $\epsilon = \sqrt{\frac{6 \log n}{b}}$, we get:

$$Pr(|E_v| < (1 + \epsilon)b) \geq 1 - \frac{n}{n^2} = 1 - \frac{1}{n}$$

□

A.6 Experimental Results for ILM

Scalability of CG: We evaluate the running time of CG while increasing the number of tuples/actions. Table 9 shows the results on FXS. Because of higher density and thus larger candidate set, CG takes longer in FCS (Table 5). Furthermore, the increment in budget does not affect the running time for CG. These observations validate the running time analysis for CG (Section 5.2.1). Table 8 shows the results on FXS data. CG produces better results than GRR and outperforms it by up to 8%.

Parameter Variation: The size of the target set X is varied and we observe its effect in Figures 7a and 7. We set $b = 2$, and remove 20 edges for these experiments. CG provides better DI consistently across different target sizes. As expected, with the increase of target set size, DI generally decreases for all the algorithms. A larger target size would have a higher influence to be reduced. Thus, with the same number of edges removed, the DI decreases for a larger target set. We compare the CG algorithm against the baseline methods on CA, FCS and FXS varying b (Figure 6). CG significantly outperforms the baselines.

A.7 Case Study on Fake News

To further demonstrate the usefulness of influence limitation, we perform a case study on fake news analysis using Twitter data [56], [57]. More specifically, we compare the set of edges removed to reduce the influence of a set of users that propagate fake tweets versus for those that propagate truthful ones. The dataset has 22.6K nodes, 23K edges, 38 truthful and 25 fake tweets with their propagation traces. We fix the budget to $b=5$ edges and select the 10 top source users in terms of propagation as the target set X.

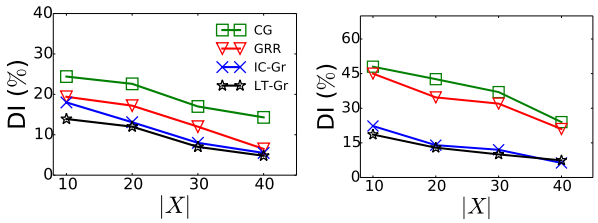
Table 10 shows the number of followers, friends and tweets for users that are endpoints of the edges selected by Algorithm 2 (BIL) using the truthful and fake tweets. Edges for fake tweets tend to connect more active (#tweets) and well-connected (#followers and #friends) users than those for the real ones. This result shows that fake news propagation relies more heavily on connections between a few key users, while truthful tweets spread more organically in the network.

	From			To		
	$\#Fos$	$\#Frs$	$\#Tws$	$\#Fos$	$\#Frs$	$\#Tws$
Fake	30k	15k	476k	29k	559	197k
	170k	12k	78k	8.5k	400	161k
	30k	15k	476k	3k	302k	29k
	170k	12k	78k	36	213	89k
	30k	15k	476k	1.2k	77	49k
Truthful	6k	6.4k	39k	2.6k	1.6k	77k
	6k	6.4k	39k	125	252	6.6k
	283	72	20k	1k	529	33k
	6k	6.4k	39k	186	370	4.7k
	6k	6.4k	39k	347	382	6.5k

TABLE 10: Endpoints (users) of edges selected for budgeted influence minimization (Fos: Followers, Frs: Friends, Tws: Tweets). Some users appear more than once. Edges selected for fake tweets connect more active (#tweets) and well-connected (#followers and #friends) users than those for the real ones.

ACKNOWLEDGEMENTS

Research was funded by National Science Foundation, IIS, Award# 1817046 and Defense Threat Reduction Agency, Award# HDTRA1-19-1-0017.



(a) Quality on CA (b) Quality on FXS

Fig. 7: **[ILM]** Decrease in Influence (DI) produced by different algorithms varying the size of the target set, X when $b = 2$.

REFERENCES

[1] A. Anagnostopoulos, R. Kumar, and M. Mahdian, "Influence and correlation in social networks," in *SIGKDD*, 2008, pp. 7–15.

[2] S. Aral, L. Muchnik, and A. Sundararajan, "Distinguishing influence-based contagion from homophily-driven diffusion in dynamic networks," *Proceedings of the National Academy of Sciences*, vol. 106, no. 51, pp. 21 544–21 549, 2009.

[3] A. D. Kramer, J. E. Guillory, and J. T. Hancock, "Experimental evidence of massive-scale emotional contagion through social networks," *Proceedings of the National Academy of Sciences*, vol. 111, no. 24, pp. 8788–8790, 2014.

[4] J. Metcalf and K. Crawford, "Where are human subjects in big data research? the emerging ethics divide," *Big Data & Society*, vol. 3, no. 1, p. 2053951716650211, 2016.

[5] S. T. Fiskea and R. M. Hauserb, "Protecting human research participants in the age of big data," *PNAS*, vol. 111, no. 38, pp. 13 675–13 676, 2014.

[6] S. Aral and P. S. Dhillon, "Social influence maximization under empirical influence models," *Nature human behaviour*, pp. 375–382, 2018.

[7] A. Goyal, F. Bonchi, and L. V. Lakshmanan, "A data-based approach to social influence maximization," *Proceedings of the VLDB Endowment*, vol. 5, no. 1, pp. 73–84, 2011.

[8] H. Tong, B. A. Prakash, T. Eliassi-Rad, M. Faloutsos, and C. Faloutsos, "Gelling, and melting, large graphs by edge manipulation," in *CIKM*. ACM, 2012, pp. 245–254.

[9] C. J. Kuhlman, G. Tuli, S. Swarup, M. V. Marathe, and S. Ravi, "Blocking simple and complex contagion by edge removal," in *International Conference on Data Mining (ICDM)*. IEEE, 2013, pp. 399–408.

[10] H. Aziz, S. Bouveret, I. Caragiannis, I. Giagkousi, and J. Lang, "Knowledge, fairness, and social constraints." in *AAAI*, 2018.

[11] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher, "Best algorithms for approximating the maximum of a submodular set function," *Math. Oper. Res.*, pp. 177–188, 1978.

[12] C. Chekuri and A. Kumar, "Maximum coverage problem with group budget constraints and applications," in *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*. Springer, 2004, pp. 72–83.

[13] 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*. ACM, 2003, pp. 137–146.

[14] 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*. ACM, 2007, pp. 420–429.

[15] A. Goyal, F. Bonchi, and L. V. Lakshmanan, "Learning influence probabilities in social networks," in *International conference on Web search and data mining (WSDM)*. ACM, 2010, pp. 241–250.

[16] C. Chekuri, J. Vondrak, and R. Zenklusen, "Dependent randomized rounding via exchange properties of combinatorial structures," in *Foundations of Computer Science (FOCS)*. IEEE, 2010, pp. 575–584.

[17] J. Vondrák, "Submodularity and curvature: The optimal algorithm," 2010.

[18] D. P. Williamson and D. B. Shmoys, *The design of approximation algorithms*. Cambridge, 2011.

[19] J. Vondrák, "Optimal approximation for the submodular welfare problem in the value oracle model," in *Proceedings of the 40th Annual ACM Symposium on Theory of Computing (STOC)*, 2008, pp. 67–74.

[20] J. Nocedal and S. J. Wright, "Numerical optimization (second edition)," 2006.

[21] A. Bozorgi, S. Samet, J. Kwisthout, and T. Wareham, "Community-based influence maximization in social networks under a competitive linear threshold model," *Knowledge-Based Systems*, vol. 134, pp. 149–158, 2017.

[22] A. Tsang, B. Wilder, E. Rice, M. Tambe, and Y. Zick, "Group-fairness in influence maximization," in *IJCAI*, 2019.

[23] M. Lake, "A new campaign resource allocation model," in *Applied Game Theory*. Springer, 1979, pp. 118–132.

[24] A. Yadav, B. Wilder, E. Rice, R. Petering, J. Craddock, A. Yoshioka-Maxwell, M. Hemler, L. Onasch-Vera, M. Tambe, and D. Woo, "Bridging the gap between theory and practice in influence maximization: Raising awareness about hiv among homeless youth." in *IJCAI*, 2018, pp. 5399–5403.

[25] Y. Lin, W. Chen, and J. C. Lui, "Boosting information spread: An algorithmic approach," in *International Conference on Data Engineering (ICDE)*. IEEE, 2017, pp. 883–894.

[26] E. B. Khalil, B. Dilkina, and L. Song, "Scalable diffusion-aware optimization of network topology," in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2014, pp. 1226–1235.

[27] A. Mislove, M. Marcon, K. P. Gummadi, P. Druschel, and B. Bhattacharjee, "Measurement and Analysis of Online Social Networks," in *Proceedings of the 5th ACM/Usenix Internet Measurement Conference (IMC'07)*, 2007.

[28] M. Kimura, K. Saito, and H. Motoda, "Minimizing the spread of contamination by blocking links in a network." in *AAAI*, vol. 8, 2008, pp. 1175–1180.

[29] C. Gao, J. Liu, and N. Zhong, "Network immunization and virus propagation in email networks: experimental evaluation and analysis," *Knowledge and Information Systems*, vol. 27, no. 2, pp. 253–279, 2011.

[30] C. M. Schneider, T. Mihaljev, S. Havlin, and H. J. Herrmann, "Suppressing epidemics with a limited amount of immunization units," *Physical Review E*, vol. 84, no. 6, p. 061911, 2011.

[31] M. Wang, J. Kang, C. Nan, Y. Xia, W. Fan, and H. Tong, "Graph ranking auditing: Problem definition and fast solutions," *IEEE Transactions on Knowledge and Data Engineering*, 2020.

[32] I. Bogunovic, "Robust protection of networks against cascading phenomena," Ph.D. dissertation, Master Thesis ETH Zürich, 2012.

[33] D. Sheldon, B. Dilkina, A. N. Elmachtoub, R. Finseth, A. Sabharwal, J. Conrad, C. P. Gomes, D. Shmoys, W. Allen, O. Amundsen *et al.*, "Maximizing the spread of cascades using network design," 2012.

[34] V. Chaoji, S. Ranu, R. Rastogi, and R. Bhatt, "Recommendations to boost content spread in social networks," in *International conference on World Wide Web (WWW)*. ACM, 2012, pp. 529–538.

[35] C. Budak, D. Agrawal, and A. El Abbadi, "Limiting the spread of misinformation in social networks," in *Proceedings of the 20th international conference on World wide web*. ACM, 2011, pp. 665–674.

[36] N. P. Nguyen, G. Yan, M. T. Thai, and S. Eidenbenz, "Containment of misinformation spread in online social networks," in *Proceedings of the 4th Annual ACM Web Science Conference*. ACM, 2012, pp. 213–222.

[37] X. He, G. Song, W. Chen, and Q. Jiang, "Influence blocking maximization in social networks under the competitive linear threshold model," in *SDM*. SIAM, 2012, pp. 463–474.

[38] G. Calinescu, C. Chekuri, M. Pál, and J. Vondrák, "Maximizing a monotone submodular function subject to a matroid constraint," *SIAM Journal on Computing*, vol. 40, no. 6, pp. 1740–1766, 2011.

[39] Y. Lin and K. Mouratidis, "Best upgrade plans for single and multiple source-destination pairs," *GeoInformatica*, pp. 365–404, 2015.

[40] A. Meyerson and B. Tagiku, "Minimizing average shortest path distances via shortcut edge addition," in *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX-RANDOM)*. Springer, 2009, pp. 272–285.

[41] M. Papagelis, F. Bonchi, and A. Gionis, "Suggesting ghost edges for a smaller world," in *International conference on Information and knowledge management (CIKM)*, 2011, pp. 2305–2308.

[42] B. Dilkina, K. J. Lai, and C. P. Gomes, "Upgrading shortest paths in networks," in *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems*. Springer, 2011, pp. 76–91.

[43] S. Medya, J. Vachery, S. Ranu, and A. Singh, "Noticeable network delay minimization via node upgrades," *Proceedings of the VLDB Endowment*, 2018.

[44] P. Crescenzi, G. DAngelo, L. Severini, and Y. Velaj, "Greedily improving our own centrality in a network," in *International Symposium on Experimental Algorithms*. Springer, 2015, pp. 43–55.

[45] V. Ishakian, D. Erdos, E. Terzi, and A. Bestavros, "A framework for the evaluation and management of network centrality," in *SIAM International Conference on Data Mining (SDM)*. SIAM, 2012, pp. 427–438.

[46] S. Medya, A. Silva, A. Singh, P. Basu, and A. Swami, "Group centrality maximization via network design," in *SIAM International Conference on Data Mining (SDM)*, 2018.

[47] V. Amelkin and A. K. Singh, "Fighting opinion control in social networks via link recommendation," in *KDD*, 2019, pp. 677–685.

[48] P. Dey and S. Medya, "Manipulating node similarity measures in network," in *AAMAS*, 2020.

[49] S. Medya, T. Ma, A. Silva, and A. Singh, "A game theoretic approach for core resilience," in *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, 2020.

[50] Z. Zhou, F. Zhang, X. Lin, W. Zhang, and C. Chen, "K-core maximization: An edge addition approach." in *IJCAI*, 2019, pp. 4867–4873.

[51] M. Farajtabar, Y. Wang, M. Gomez-Rodriguez, S. Li, H. Zha, and L. Song, "Coevolve: A joint point process model for information diffusion and network evolution," *The Journal of Machine Learning Research*, vol. 18, no. 1, pp. 1305–1353, 2017.

[52] J. Pearl, *Causality*. Cambridge university press, 2009.

[53] S. L. Morgan and C. Winship, *Counterfactuals and causal inference*. Cambridge University Press, 2015.

[54] S. Goyal, H. Heidari, and M. Kearns, "Competitive contagion in networks," *Games and Economic Behavior*, vol. 113, pp. 58–79, 2019.

[55] J. Tsai, T. H. Nguyen, and M. Tambe, "Security games for controlling contagion," in *AAAI*, 2012.

[56] J. Ma, W. Gao, P. Mitra, S. Kwon, B. J. Jansen, K.-F. Wong, and C. Meeyoung, "Detecting rumors from microblogs with recurrent neural networks," in *The 25th International Joint Conference on Artificial Intelligence*, 2016.

[57] J. Ma, W. Gao, and K.-F. Wong, "Detect rumors in microblog posts using propagation structure via kernel learning," in *The 55th annual meeting of the Association for Computational Linguistics*, 2017.

Sourav medya is currently a postdoctoral fellow at Kellogg School of Management and Northwestern Institute of Complex Systems (NICO). He obtained his PhD degree in Computer Science from University of California, Santa Barbara. He has also been a research intern at the HP Labs and Qatar Computing Research Institute. His research interests include network science, data science, and machine learning.

Arlei Silva is currently a postdoctoral fellow in the department of Computer Science at University of California, Santa Barbara. He received his PhD degree in Computer Science from University of California, Santa Barbara. He has also been a research intern and scholar at Rensselaer Polytechnic Institute, IBM Research, HP Labs, and NEC Labs. His research interests include graphs, data mining and machine learning.

Ambuj Singh Ambuj K. Singh is a Professor of Computer Science at the University of California, Santa Barbara. He joined computer science department at UCSB in 1989. He has served on the editorial boards of journals and program committees of several conferences, workshops and international meetings. His current research interests include network science, machine learning, bioinformatics, and graph mining.