Conquering the Worst Case of Infections in Networks

WEN YAN1,(Student Member, IEEE), PO-LING LOH2, CHUNGUO LI1,(Senior Member, IEEE), YONGMING HUANG1, (Senior Member, IEEE), and LUXI YANG1, (Senior Member, IEEE)

1School of Information Science and Engineering, Southeast University, Nanjing 210096, China (e-mail: {wyan, chunguoli, huangym, lxyang}@seu.edu.cn)
2Department of Statistics, University of Wisconsin-Madison, WI 53706, USA (e-mail: ploh@stat.wisc.edu)

Corresponding author: Luxi Yang (e-mail: lxyang@seu.edu.cn).

This work was supported in part by the State Key Program of National Natural Science Foundation of China under Grant U1936201, in part by the National Natural Science Foundation of China under Grant 61971128, in part by the National Science Foundation under Grant DMS-1749857, and in part by the China Scholarship Council (CSC) under Grant 201706090055.

ABSTRACT We develop algorithms to control the scope of an infection spread on a network by allocating a fixed immunization budget to edges of the graph. We assume that the infection propagates according to an independent cascade model and interventions operate by reducing the propensities of edges to transmit the infection. We formulate this problem as a constrained min-max optimization problem with respect to the placements of interventions and the location of the worst-case seed nodes. However, the result is a challenging bilevel mixed integer optimization problem. Furthermore, gradients of the objective function with respect to the continuous variables are unavailable in closed form. We employ tools from derivative-free optimization and stochastic optimization to optimize the objective by iterating between the outer minimization and inner maximization problems. In the inner loop, we use a weighted degree discount (WDD) method to select the seed set of the influence maximization problem. In the outer loop, we utilize two methods: a sample-based simultaneous perturbation Nelder-Mead (SBSP-NM) algorithm and a simultaneous perturbation stochastic approximation (SPSA) algorithm. We perform simulations on synthetic graphs and three larger-scale real-world datasets and illustrate the computational feasibility of our algorithms and their efficacy at controlling epidemic spreads.

INDEX TERMS Immunization, bilevel optimization, derivative-free optimization, influence maximization, mixed integer program, networks

I. INTRODUCTION

IMMUNIZATION plays an important role in real-world applications, such as rumor control in social networks [1], epidemic control in public health [2], and the control of terrorist attacks for public safety [3]. However, the problem of choosing optimal interventions, subject to a limited budget, is somewhat challenging. First, the interventions must often be selected under uncertainty on the properties of the epidemic spread. For instance, the seed nodes at which the epidemic originates may be unknown, and the epidemic may propagate in a stochastic fashion. Second, the influence of different nodes on the spread of the disease may be highly coupled due to various paths connecting nodes in the network.

In our work, we adopt the approach of finding the optimal allocation of interventions with respect to the worst-case choice of seed nodes for an infection. We assume that the network and the probabilities of transmitting the infection over each edge are known or could be estimated. However, the set of seed nodes is unknown a priori. For instance, to prevent a terrorist attack, the police may already intercept the communication frequencies between terrorists but need to allocate limited police resources to control the order spread and the potential attack as much as possible without knowing which terrorists may issue the attack orders. To go one step further, we formulate the immunization problem as a min-max problem with constraints. The goal is to minimize the maximal expected number of infected nodes in a network, subject to the worst-case seed set, by reducing the propagation probabilities of an optimally chosen subset of edges. Note that unlike minimizing undirect objectives like the leading eigenvalue of the adjacent matrix, optimizing the expected number of infected nodes directly is much more practical but more challenging. What’s more, instead of adopting the node removal or edge removal strategies in complete immunization, we consider a more practically
fractional immunization scenario in this paper.

Nevertheless, the min-max immunization problem brings several difficulties, as well: (i) the inner maximization problem, known as a classical influence maximization problem, is an NP-hard integer optimization problem; (ii) the outer minimization problem is a continuous optimization problem with no convenient closed form for the objective function; and (iii) the two layers of the min-max problem are highly coupled with respect to the discrete solution space of the seed set A and the continuous solution space of the edge intervention vector θ. To deal with these difficulties, we adopt an iterative strategy to alternately optimize over A and θ separately. This produces an overall upper bound on the worst-case infection spread under the optimal intervention. In order to perform the outer minimization over θ, we utilize two methods from derivative-free optimization and stochastic optimization, respectively: the Nelder-Mead (NM) and simultaneous perturbation stochastic approximation (SPSA) algorithms. Furthermore, to make the procedure computationally feasible for large networks, we modify the traditional NM algorithm by combining sampling and simultaneous perturbation ideas, leading to a sample-based simultaneous-perturbation NM (SBSP-NM) method. For the inner maximization problem, we introduce a weighted degree discount (WDD) algorithm, which greedily selects seed nodes according to their discounted weighted degrees and is inspired by the DegreeDiscountIC algorithm [4]. In summary, the main contributions of this paper are as follows:

- **Problem formulation of finding the optimal allocation of interventions with respect to the worst-case choice of seed nodes for an infection**: A min-max problem with constraints is formulated to minimize the maximal expected number of infected nodes in a network for the first time, subject to the worst-case seed set, by reducing the propagation probabilities of an optimally chosen subset of edges.

- **Optimization to solve the min-max immunization problem**: The min-max immunization problem brings multiple difficulties, thus, an alternating optimization procedure is applied to solve the min-max problem, especially where SBSP-NM and SPSA from derivative-free optimization and stochastic optimization respectively are proposed to solve the outer continuous minimization problem with no convenient closed form for the objective function.

- **Simulations on synthetic graphs and three larger-scale real-world datasets**: Experiments on six synthetic graphs are performed to test the effectiveness of our algorithm on different types of graphs, and further on three larger-scale public real-world social network datasets, NetHEPT, ego-Facebook, and wiki-Vote, to explore the effectiveness and computational feasibility of our immunization algorithms.

**Related work**: Several other authors have proposed methods for developing optimal immunization strategies on networks. Tong et al. [5] propose an effective algorithm to optimize the leading eigenvalue based on the edge removal strategy to solve the NetMelt Problem. Prakash et al. [6] formulate a minimization problem involving the top eigenvalue of the adjacency matrix, and propose a linear-time algorithm to batch-allocate resources greedily. Chen et al. [7] study immunization strategies that operate on nodes and propose scalable algorithms to find an optimal set of nodes to be immunized based on a novel score definition. Zhang et al. [8] propose algorithms based on greedily reducing the infection count on a graph based on sampled cascades, using additional information from data logs. From the viewpoint of control theory, Nowzari et al. [9] consider methods for controlling epidemics using a generalized susceptible-exposed-infected-vigilant model and propose several cost-optimal strategies based on geometric programming. Preciado et al. [10] study the problem of finding the optimal distribution of vaccines based on an optimization framework under semidefinite programming constraint. Besides, several studies take network measures in complex networks as the objective of immunization, such as both Chan et al. [11] and Gunasekara et al. [12] study the immunization problem from the network robustness viewpoint by edge deletions and additions, and Shen and Cao [13] aim to maintain the network efficiency by reducing the edge weights on weighted networks. However, none of these papers take the approach of controlling the worst-case spread with respect to adversarially chosen seed nodes, which is the method we adopt in this paper.

We also discuss several papers related to the influence maximization problem, which arises as the inner maximization problem in our iterative optimization approach. Influence maximization has long been a popular topic of research in graph mining. In a seminal paper, Kempe et al. [14] formulated the influence maximization problem as a submodular problem and proved that greedy hill-climbing algorithms achieve a \((1 - 1/e)\)-approximation of the optimal solution when the infection spreads according to a linear threshold or independent cascade model. Liontis and Pitoura [15] aim to increase the influence of a seed set by boosting the activation probabilities of edges adjacent to an optimally chosen set of nodes. However, unlike our approach, the authors assume that the initial seed set is known. Khim et al. [16] propose algorithms for influence maximization in stochastic and adversarial contagion settings and establish related upper and lower bounds. Chen et al. [17] address the scalability issue in influence maximization by restricting computations on local influence regions of nodes. Other popular state-of-the-art methods on influence maximization include the Cost-Effective Lazy Forward (CELF) algorithm of Leskovec et al. [18], the CELF++ algorithm of Goyal et al. [19], the Reverse Influence Sampling (RIS) algorithm of Borgs et al. [20], etc. However, to the best of our knowledge, we are the first to combine influence maximization and immunization in a min-max bilevel optimization framework.

The remainder of our paper is organized as follows: Section II provides the mathematical formulation of our problem,
as well as related notation. Section III contains our proposed algorithms for solving the immunization problem. Section IV provides experimental assessments of our method on different types of synthetic and real-world datasets. We conclude the paper in Section V.

II. PROBLEM SETUP

Consider an undirected graph \( G := (V, E) \), where \( V = \{1, \ldots, N\} \) denotes the set of nodes and \( E \subseteq V \times V \) denotes the set of edges. If nodes \( i \) and \( j \) are connected by an edge, we include both \((i, j)\) and \((j, i)\) in \( E \). Let \( p := \{p_{ij}, (i, j) \in E\} \subseteq \mathbb{R}^{N \times N} \) denote the matrix of edge weights, where \( 0 \leq p_{ij} = p_{ji} \leq 1 \) for each \((i, j) \in E\).

We assume that given a set of seed nodes \( A \subseteq V \), an infection spreads in discrete time according to the independent cascade (IC) model: if \( A_t \) is the set of newly infected nodes at time \( t \), the probability of an uninfected node \( i \) becoming infected at time \( t+1 \) is \( 1 - \prod_{j: (i, j) \in E, j \in A_t} (1 - p_{ij}) \). In other words, each newly infected neighbor \( j \) has probability \( p_{ij} \) of independently infecting node \( i \). The influence of a seed set \( A \) is the expected number of infected nodes at the conclusion of the epidemic, assuming \( A_0 = A \). Let \( f_0(G, A, p) \) denote this expected quantity, assuming the epidemic spreads on the graph \( G \) with edge weights \( p \). Note that since we focus on the operations on the graph based directly on \( f_0(G, A, p) \) other than how the infection spreads, our work is very general and not constrained to fit the IC model, which we adopt here for demonstration and simulation.

The influence maximization problem aims to solve the problem \( \max_{A \subseteq V, |A| \leq k} f_0(G, A, p) \), where \( G \) and \( p \) are fixed and assumed to be known as the prior information. In our setting, we assume we have a fixed budget \( B \) that can be used to immunize edges in the graph, thus decreasing the corresponding edge weights. Thus, we arrive at the following problem formulation:

\[
\min_{\theta} \max_A f_0(G, A, p - \theta)
\]  

(1)

\[
s.t. \sum_{(i, j) \in E} \theta_{ij} \leq B \tag{1a}
\]

\[
0 \leq \theta_{ij} \leq p_{ij}, \forall (i, j) \in E \tag{1b}
\]

\[
|A| \leq k. \tag{1c}
\]

Here, \( \theta := \{\theta_{ij}, (i, j) \in E\} \) denotes the matrix of interventions, so the expected number of infected nodes is \( f_0(G, A, p - \theta) \) after the intervention is applied. Moreover, \( (1a) \) and \( (1b) \) are constraints on \( \theta \) to characterize the budget of immunization and the value range, respectively, and \( (1c) \) is a widely adopted condition to constrain the number of seed nodes. Note that the interventions here belong to fractional immunizations other than complete immunizations by edge removal or node removal, which are more practical since there is no need to change the topology of the network.

For better demonstration, we take the epidemic control network for example. We assume that we can get the sets of the concerned persons (nodes) and their connections (edges), and estimate the propagation probabilities as the edge weights \( (p) \). But we don’t know the set of the sudden infected persons \((A)\), and they will spread to infect their neighbors. We aim to minimize the number of infected persons (denoted as \( f_0 \)) by reducing the propensities of edges to transmit the infection \((\theta)\), namely assigning a constrained number of disinfectants to decrease the infected probabilities. However, the explicit form of \( f_0 \) is unavailable, let alone the properties like continuity, convexity, and derivability, which makes the problem \( (1) \) extremely difficult.

In order to make the operations of our algorithms on each dimension of \( \theta \) explicit, we will use the vector notation \( \theta = [\theta_i] \), with index \( i \) ranging from 1 to \( M := |E| \), and the corresponding notation \( p = [p_{ij}] \). What’s more, it’s trivial to consider \( k \) as an upper bound on the size of the infection seed set.

III. ALGORITHMS

In this section, we describe our proposed algorithms for solving the min-max problem \( (1) \). We apply an alternating optimization procedure \([21] [22]\), where we iteratively maximize with respect to the discrete seed set \( A \) and minimize with respect to the continuous intervention vector \( \theta \). What’s more, for better reproducibility, we state all the algorithms as detailed as possible.

A. SOLVING INNER OPTIMIZATION WITH FIXED \( \theta \)

To optimize over \( A \) with respect to a fixed vector \( \theta \), problem \( (1) \) simply reduces to the conventional influence maximization problem \( (2) \), with fixed edge weights \( p - \theta \).

\[
\max_A f_0(G, A, p - \theta)
\]

(2a)

\[
s.t. |A| \leq k. \tag{2b}
\]

One popular method for solving the influence maximization problem is the general greedy algorithm \([14]\), which greedily adds nodes to the selected seed set to successively maximize the overall influence. However, the simple greedy algorithm can still be computationally expensive on large graphs, especially since the influence function \( f_0(G, A, p - \theta) \) can only be approximated for each candidate set \( A \) by simulating multiple runs of the epidemic starting from the seed set \( A \), and taking an average over the number of infected nodes on each run. For demonstration, we assume all the evaluations of the influence function are estimated by \( R_{cas} \) runs of the epidemic according to the independent cascade model in the following paper.

Inspired by the DegreeDiscountIC algorithm \([4]\), we propose a weighted degree discount (WDD) heuristics algorithm. The main differences between WDD and DegreeDiscountIC are that (i) WDD greedily selects nodes according to their discounted weighted degrees, which is much more
suitable for cases where edge weights matter; and (ii) we filter out the nodes that still have strong connections with seeds even after the immunization. The WDD algorithm is stated in Algorithm 1. Here, we set strong_connection_thres to be a predetermined threshold to decide whether the nodes have strong connections to seed nodes after immunization. The value of $\epsilon$ is chosen to be a suitably small number. Since we pay more attention to the outer minimization problem, further improvements could be to test the performance under different combinations including like RIS [20]. It’s also reasonable to adopt WDD to solve the inner maximization problem for demonstration, because it could provide comparisons under different outer solvers and always lower the adversarial influence spread compared to the optimal solutions.

**Algorithm 1** Weighted degree discount (WDD) algorithm to optimize over $A$ with fixed $\theta$, $WDD(G,k,\theta)$

1. **input** graph $G$, seed size $k$, edge weight modification $\theta$
2. **initialize** $A = \emptyset$, strong_connection_thres, $\epsilon$
3. **for** each node $v \in V$ **do**
   4. compute its weighted degree $wd_v$ based on $p - \theta$
   5. **initialize** its weighted degree discount $wdd_v = wd_v$
4. **end for**
7. **for** $i = 1$ to $k$ **do**
8. select $u = \arg\max_{v \in V/S} \{ wdd_v \}$
9. $A = A \cup \{u\}$
10. **for** each neighbor $v$ of $u$ and $v \in V/S$ **do**
11. if $p_{uv} - \theta_{uv} > $ strong_connection_thres
12. $wdd_v = \epsilon$
13. else
14. $wdd_v = wdd_v - (p_{uv} - \theta_{uv})$
15. **end if**
16. **end for**
17. **end for**
18. **output** $A$

**B. SOLVING OUTER OPTIMIZATION WITH FIXED $A$**

Turning to the problem of maximizing over $\theta$ for a fixed seed set $A$, we obtain a continuous constrained problem (3) as follows:

\[
\min_{\theta} f(\theta) := f_0(G,A,p - \theta) \tag{3}
\]

\[
\text{s.t. } \sum_{i=1}^{M} \theta_i \leq B \tag{3a}
\]

\[
0 \leq \theta_i \leq p_i, \forall i \in [1,M]. \tag{3b}
\]

However, computing the influence spread $f$ given a fixed seed set $A$ is #P-hard [17], let alone the explicit form of $f$ as a function of $\theta$, even if $G$ and $p - \theta$ are known explicitly. To overcome the challenge, we propose to use two methods from derivative-free [23] and stochastic optimization [24]. The first method is based on the Nelder-Mead (NM) method, which proceeds by evaluating the value of the objective function on vertices of a successively changing simplex. Hence, we propose an adaptation of the NM algorithm that is more suitable for large-scale problems and can also incorporate constraints on the parameters. The second method is to use the simultaneous perturbation stochastic approximation (SPSA) algorithm, which obtains successive iterates by approximating the gradient of the objective function in a stochastic manner using simultaneous perturbations in all dimensions.

For both methods, note that the objective function $f$ also needs to be approximated for each value of $\theta$ in the solution path. This is conducted by simulating the infection spread with seed set $A$ and edge weights $p - \theta$ and taking an average over the infection count after a suitably large number of simulations. This simulation-based approach has the advantages of model generality and a good theoretical property that it commonly returns a solution with a constant bounded approximation ratio, while suffering significant computational overheads [25].

1) Nelder-Mead method

The NM algorithm was proposed in 1965 [26], and soon became one of the most popular algorithms in derivative-free optimization, due to its easy understanding and implementation. What’s more, its “best-case efficiency” frequently outweighs its lack of convergence theory [27]. The algorithm evaluates the value of the objective function at the vertices of a simplex and performs a direct search over simplex transformations until convergence [27]. For an $M$-dimensional problem, the NM algorithm first generates a simplex with $M + 1$ points and then performs successive simplex transformations using five operations: reflection, expansion, outside contraction, inside contraction, and shrinkage.

However, applying the NM algorithm to large graphs with tens of thousands of edges is not computationally feasible. Furthermore, the function evaluations must all be approximated via simulations of the infection. Instead, we modify the traditional NM method by combining sampling and simultaneous perturbation, leading to a sample-based simultaneous perturbation NM (SBSP-NM) method. The difference between SBSP-NM and the traditional NM method is that our initial simplex lies in a much smaller predetermined $n$-dimension subspace. To form the initial simplex, we first divide the $M$ coordinate dimensions into $m$ blocks and then generate $n$ points by randomly perturbing the corresponding $m$ blocks by a fixed step size around the origin. This forms an initial simplex using the $n$ points and the origin. As a result, our SBSP-NM method can largely reduce the number of function evaluations from $M + 1$ to $n + 1$ on simplex initialization and shrinkage steps, while revealing $m$-dimensional information with each initial vertex of the simplex. In general, we can choose $m$ such that $nm$ is several times larger than $M$.

Moreover, the traditional NM method is only suitable for unconstrained problems. Thus, we adopt a box constraint
method to trim $\theta$ with its lower and upper bounds as used in Cauchy point computation [28], and use a normalized scaling $\theta = (B/|\theta|_1) * \theta$ when $|\theta|_1$ exceeds the sum- constraint B. Our SBSP-NM algorithm is described in Algorithm 2.

Algorithm 2 SBSP-NM algorithm to optimize over $\theta$ with fixed $A$, algNM($G,A,B$)

1: input graph $G$, seed set $A$, modification summation constraint $B$
2: initialize $x^0 = 0$, step, no_improve_thres, no_improve_break,
3: no_improve_num = 0, $\alpha_{nm} = 1$, $\gamma_{nm} = 2$, $\rho = 0.5$,
4: check $\sigma_{nm} = 0.5$, $m$, $n$
5: compute $f(x^0)$, and define $f_{prev\_best} = f(x^0)$
6: block_size = floor($M/m$)
7: for $j = 1$ to $n$ do
8: $y^j = x^0$
9: for $k = 1$ to $m$ do
10: rand $\leftarrow$ Random($0,1$)
11: $kk = \text{int(floor(rand * block_size + (k - 1) * block_size))}$
12: $y^j_{kk} = y^j_{kk} + \text{step}$
13: end for
14: $y^j = \text{transform}(y^j, 0, p, B)$
15: end for
16: generate the initiate simplex $Y = \{x^0, y^1, \ldots, y^n\}$
17: while 1:
18: order the vertices of $Y$ and change notations $Y = \{x^1, \ldots, x^{n+1}\}$, s.t. $f(x^1) < \cdots < f(x^{n+1})$
19: define $f_{best} := f(x^1)$
20: $\theta = x^1$
21: if $f_{best} < f_{prev\_best} - \text{no\_improve\_thres}$ then
22: no_improve_num = 0
23: $f_{prev\_best} = f_{best}$
24: else no_improve_num + 1
25: end if
26: if no_improve_num > no_improve_break then
27: break
28: end if
29: compute the centroid of best $n$ points $\bar{x} = \frac{1}{n} \sum_{i=1}^{n} x^i$
30: (reflection) compute $x^r = \bar{x} + \alpha_{nm}(\bar{x} - x^{n+1})$, then transform it by $x^r = \text{transform}(x^r, 0, p, B)$ and evaluate $f(x^r)$
31: if $f(x^r) \leq f(x^r) < f(x^n)$ then
32: replace $x^{n+1} \in Y$ by $x^r$, continue
33: else if $f(x^r) < f(x^n)$ then
34: (expansion) compute $x^e = \bar{x} + \gamma_{nm}(\bar{x} - x^{n+1})$, then transform it by $x^e = \text{transform}(x^e, 0, p, B)$ and evaluate $f(x^e)$
35: if $f(x^e) < f(x^n)$ then
36: replace $x^{n+1} \in Y$ by $x^e$, continue
37: else replace $x^{n+1} \in Y$ by $x^r$, continue
38: end if
39: else $f(x^r) \geq f(x^n)$
40: (outside contraction) compute $x^{oc} = \bar{x} + \rho_{nm}(\bar{x} - x^{n+1})$, then transform it by $x^{oc} = \text{transform}(x^{oc}, 0, p, B)$ and evaluate $f(x^{oc})$
41: if $f(x^{oc}) \leq f(x^r)$
42: replace $x^{n+1} \in Y$ by $x^{oc}$, continue
43: else go to shrinkage step
44: end if
45: else $f(x^r) \geq f(x^{n+1})$
46: (inside contraction) compute $x^{ic} = \bar{x} - \rho_{nm}(\bar{x} - x^{n+1})$, then transform it by $x^{ic} = \text{transform}(x^{ic}, 0, p, B)$ and evaluate $f(x^{ic})$
47: if $f(x^{ic}) < f(x^{n+1})$ then
48: replace $x^{n+1} \in Y$ by $x^{ic}$, continue
49: else go to shrinkage step
50: end if
51: (shrinkage) define n new vertices $y^i = x^1 + \sigma_{nm}(x^1 - x^i), i = 1, \ldots, n$, then transform them by $y^i = \text{transform}(y^i, 0, p, B), i = 1, \ldots, n$ and new simplex $Y^+ = \{x^1, y^1, \ldots, y^n\}$, continue
52: end if
53: end if
54: output $\theta$

And in detail, the transform function of the box constraint method is stated in Algorithm 3.

Algorithm 3 Box constraint method to deal with the constraints (1a) and (1b), transform($x$, LB, UB, ConstraintSum)

1: input $x$, LB, UB, ConstraintSum
2: for $i = 1$ to $M$ do
3: if $x_i > UB_i$ then
4: $x_i = UB_i$
5: end if
6: if $x_i < LB_i$ then
7: $x_i = LB_i$
8: end if
9: end for
10: if $\|x\|_1 < \text{ConstraintSum}$ then
11: break
12: else $x = (\text{ConstraintSum}/\|x\|_1) * x$
13: end if
14: output $x$

2) SPSA method
We now describe the simultaneous perturbation stochastic approximation (SPSA) algorithm from stochastic optimization [29]. The SPSA algorithm proceeds by approximating the gradient of $f$ based on stochastic function evaluations, and then performs an analog of gradient descent [30] [31]. We also incorporate the constraints on $\theta$ using the box constraint method adopted in the Nelder-Mead method. The SPSA algorithm is described in Algorithm 4, where the parameters $\alpha_{\text{spsa}} = 0.602$ and $\gamma_{\text{spsa}} = 0.101$ are demonstrated to perform well in practical settings [29]. The convergence guarantee of SPSA requires $f$ to be Lipschitz continuous and
differentiable with bounded second-order derivatives [30], which are not satisfied here, while the algorithm could still be workable under its monotone decreasing property with respect to θ.

Algorithm 4 SPSA algorithm to optimize over θ with fixed A, algSPSA(G,A,B)

\begin{algorithm}
\caption{Algorithm 4 SPSA algorithm to optimize over θ with fixed A, algSPSA(G,A,B)}
\begin{algorithmic}[1]
\STATE \textbf{input} graph G, seed set A, modification constraint B
\STATE \textbf{initialize} θ(1) = 0, R_{spsa}, α_{spsa} = 0.602, γ_{spsa} = 0.101, α, c, α_{big}
\FOR {n = 1 to R_{spsa}}
\STATE define \( f_{\text{best}} := f(\theta(1)) \)
\STATE θ = θ(1)
\FOR {i = 1 to M}
\STATE \( \Delta_1(n) \leftarrow \text{Bernoulli}(1/2) \)
\ENDFOR
\STATE \( a(n) = \frac{α}{(n+1)^α} \)
\STATE \( δ(n) = \frac{β}{(n+1)^β} \)
\STATE \( \theta(n)^+ = \text{transform}(θ(n) + δ(n)\Delta(n), 0, p, B) \)
\STATE \( \theta(n)^- = \text{transform}(θ(n) - δ(n)\Delta(n), 0, p, B) \)
\STATE \( f(\theta(n)^+) = f_0(G, A, p - \theta(n)^+) \)
\STATE \( f(\theta(n)^-) = f_0(G, A, p - \theta(n)^-) \)
\IF {f(\theta(n+1)) < f_{\text{best}}}
\STATE \( f_{\text{best}} = f(\theta(n+1)) \)
\STATE θ = θ(n+1)
\ENDIF
\ENDFOR
\STATE θ = θ(n+1)
\ENDFOR
\STATE \textbf{output} θ
\end{algorithmic}
\end{algorithm}

C. SOLVING THE MIN-MAX PROBLEM

To solve the original min-max problem (1), we alternately optimize with respect to A and θ, using WDD and SBSP-NM/SPSA, respectively, up to a predetermined iteration number \( R_{\text{alt}} \). The procedure is summarized in Algorithm 5, where the hyper-parameter \( R_{\text{alt}} \) provides a tradeoff between effectiveness and efficiency. For a better explanation, we also provide a graphical demonstration to show the optimization process of the algorithms more explicitly in Fig. 1.

Next, we do a theoretical analysis on the output of Algorithm 5.

Remark 1. The output \( \hat{A} \) serves as a lower bound on the decrement of the expected infection count in the worst-case epidemic spread under the optimal intervention, even if Algorithm 5 does not converge to the optimum of the min-max problem (1).

In detail, if \( \hat{A} = \arg \max_{|A| \leq k} f_0(G, A, p - \theta) \) and \((A^*, \theta^*)\) is the solution to the problem (1), we have \( f_0(G, A^*, p - \theta^*) \leq f_0(G, \hat{A}, p - \theta) \). Hence, a smaller value of \( f_0(G, \hat{A}, p - \theta) \) in comparison to \( \max_{|A| \leq k} f_0(G, A, p) \) ensures that \( \theta^* \) will also decrease the expected infection count in the worst case by at least the same amount.

Remark 2. It is guaranteed that intervening according to \( \theta \) will control the worst-case epidemic to the level of \( f_0(G, \hat{A}, p - \theta) \), which is in turn lower bounded by \( f_0(G, A, p - \theta) \).

D. COMPLEXITY ANALYSIS OF THE ALGORITHMS

In this subsection, we analyze the computational complexity of the algorithms in sequence.

The computational complexity of Algorithm 1 could be \( O(k\log N + M) \) by using the Fibonacci heap, much smaller than the \( O(kN R_{\text{cas}} M) \) of the general greedy algorithm.

For Algorithm 2, we analyze each part of Algorithm 2.
separately and get $O((n + R_{\text{nm}})R_{\text{cas}}M)$, largely reduced compared to the $O(R_{\text{cas}}M^2 + R_{\text{nm}}R_{\text{cas}}M)$ of the traditional NM method, where $R_{\text{nm}}$ denotes the iteration number of the simplex transformations. Moreover, the adoption of the shrinkage operation is considered to be a rarity, which has been verified by Torczon in [32].

The computational complexity of Algorithm 4 is $O(R_{\text{spua}}R_{\text{cas}}M)$. It's noteworthy that unlike solving the integer optimization problem with submodular property in the inner optimization, the outer optimization works on the dynamic graph with varying edge weights $p - \theta$ and solves the continuous optimization problem with no convenient closed form for the objective function. Thus, it's difficult to avoid the evaluations of influence function, which also are the computational burden both in Algorithm 2 and Algorithm 4. The computational burden of Algorithm 5 is $O(R_{\text{alt}}(\log N + (n + R_{\text{nm}})R_{\text{cas}}M))/O(R_{\text{alt}}(\log N + R_{\text{spua}}R_{\text{cas}}M))$ for the adoption of SBSP-NM/SPSA, respectively.

Finally, they could be summarized in Table. 1. From Table. 1, we could see that Algorithm 2 has comparable computational complexity with Algorithm 4. Note that since our focus is on designing effective algorithms to deal with the min-max problem (1), their efficiencies could further be improved by relieving the heavy computational burden $O(R_{\text{cas}}M)$ resulted from each evaluation of influence function, like replacing this simulation-based approach with proxy-based or sketch-based approaches [25].

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Computational complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Algorithm 1</td>
<td>$O(k\log N + M)$</td>
</tr>
<tr>
<td>Algorithm 2</td>
<td>$O((n + R_{\text{nm}})R_{\text{cas}}M)$</td>
</tr>
<tr>
<td>Algorithm 4</td>
<td>$O(R_{\text{spua}}R_{\text{cas}}M)$</td>
</tr>
<tr>
<td>Algorithm 5(SBSP-NM)</td>
<td>$O(R_{\text{alt}}(\log N + (n + R_{\text{nm}})R_{\text{cas}}M))$</td>
</tr>
<tr>
<td>Algorithm 5(SPSA)</td>
<td>$O(R_{\text{alt}}(\log N + R_{\text{spua}}R_{\text{cas}}M))$</td>
</tr>
</tbody>
</table>

We compare the proposed algorithm with the following four edge removal methods and one fractional immunization method. (a) Degree: selecting the edges with the largest degrees successively (the degree of an edge is the minimum of the two degrees of the endpoints) until the summation constraint of the intervention meets. Here, for all four edge removal baselines, we assume the selected edge could have a 99% edge weight reduction as the intervention, namely $\theta_{ij} = 0.99*p_{ij}$ for each selected edge $(i, j)$, and the degree of a node is computed. (b) Rich-Rich: selecting the edges with the highest scores successively (the score of an edge is the product of the two degrees of the endpoints) until the summation constraint of the intervention meets. (c) Pagerank: selecting the edges with the largest PageRank scores [33] (the PageRank score of an edge is the minimum PageRank score among its two endpoints) until the summation constraint of the intervention meets. (d) NetMelt: selecting the edges with the highest scores that minimize the leading eigenvalue of the network [5]. (e) UNI: uniformly assigning the intervention according to their edge weights, namely $\theta = \frac{B}{|p|}p$. Note that the degrees of nodes in Degree and Rich-Rich are computed based on the corresponding converted directed graphs by replacing each edge with two directed edges, likewise for the PageRank scores in Pagerank.

IV. EXPERIMENTAL EVALUATIONS

Our experimental evaluations can be classified into two parts. First, to test the effectiveness of our algorithm on different types of graphs, we perform experiments on six synthetic graphs. We keep the size of the graphs small so that we can properly compare the output of our WDD algorithm to the exhaustive search method and general greedy algorithm [4] for solving the inner influence maximization problem. We call these graphs G10-related graphs. Second, we test the performance of our algorithms on three larger-scale public real-world social network datasets, NetHEPT, ego-Facebook, and wiki-Vote, to further explore the effectiveness and computational feasibility of our immunization algorithms.

What’s more, though none of the related work considers the min-max immunization formulation, we design a set of experiments on the three public datasets to compare the two proposed methods to the existing state-of-the-art edge removal methods for immunization in networks (since node removal methods for immunization have a conflict with the inner influence maximization problem by node selection).
weights for the G-Facebook graph than the G-NetHEPT graph, due to the relative density of the Facebook network.

wiki-Vote: This dataset contains all the Wikipedia voting data from the inception of Wikipedia till January 2008 [35]. It contains 7,115 nodes and 100,762 undirected edges (converted from 103,689 directed edges), where nodes represent Wikipedia users and each edge indicates a voting action. We randomly generated edge weights uniformly sampled from (0,0.1) to obtain the weighted graph G-wikiVote.

In summary, though the three real-world public graphs relate to different social behaviors like collaboration and voting, the immunization considered here could all be regarded as the control of the influence spread on these networks.

B. RESULTS FOR G10-RELATED GRAPHS

In this subsection, we report the results of our simulations, providing comparisons within two groups: (i) {NM, SPSA} for optimization over $\theta$; and (ii) {WDD, GG, EX} for solving $A$, where GG and EX are the general greedy algorithm [4] and exhaustive search algorithms, respectively. We applied the exhaustive search algorithm only for $k = 2$. Note that we used the traditional NM algorithm for these experiments, rather than the SBSP-NM algorithm, since the graph size is fairly small.

**Varying seed size $k$:** Here, we have four combinations of algorithms (without including EX), and vary the seed size $k$ from 1 to 5. We set the summation constraint $B = 0.2||p||_1$. For GG, we simulated the infection process 10,000 times to approximate the influence function when selecting successive seed nodes. Furthermore, we used 500 simulations of the infection to approximate the value of $f$ in our algorithms. We ran the iterative algorithm with 10 alternating steps. From Figure 2, we can see that WDD generally returns smaller values than GG in the influence maximization task (larger is better for the same outer optimization solver), indicating that WDD is slightly inferior to GG under our configurations. However, considering the substantial reduction in computational complexity (as shown in Section III-D), WDD is still acceptable for large graphs, since GG is usually computationally infeasible for them. What’s more, NM is slightly superior to SPSA on these small graphs (smaller is better for the same inner optimization solver).

**Varying summation constraint $B$:** We have 6 combinations of possible algorithms. We ran simulations with $B$ ranging from 0 to $||p||_1$ and set the seed size $k = 2$. The other parameters were set in the same manner as before. Inspecting Figure 3, we can conclude that both WDD and GG perform worse than EX (the best influence maximization method when the seed size $k = 2$), which fits the intuition well. But WDD also returns acceptable results compared to EX and
GG, especially considering the large computational complexity reduction, consistent with the results in Figure 2. Another interesting phenomenon is that SPSA sometimes outperforms NM when \( B \) is large (like larger than 50% \( |p| \)). This suggests that NM is better than SPSA in weak immunization cases, while SPSA may be better in strong immunization cases.

C. RESULTS FOR NETHEPT, EGO-FACEBOOK AND WIKI-VOTE GRAPHS

In this subsection, we make comparisons between the algorithms in the group \{SBSP-NM, SPSA, Degree, Rich-Rich, Pagerank, NetMelt, UNI\} to optimize over \( \theta \). In all cases, we use WDD to perform the inner optimization over \( A \) for computational complexity concern.

Varying seed size \( k \): To use a unified scale for the three
datasets, we varied the seed size $k$ from 6% of the respective number of nodes to 30%. We set $B = 0.1 ||p||_1$. For G-NetHEPT, G-Facebook, and G-wikiVote, we ran 10, 30, and 10 independent cascade model simulations to approximate the infection counts, and ran the alternating algorithm all with 10 iterations, respectively. From Figure 4, we can see that SPSA and SBSP-NM outperform the other five baselines most of the time. Furthermore, SBSP-NM performs much better than SPSA on all three graphs, under the weak immunization cases where fixed summation constraint $B = 0.1 ||p||_1$ is adopted. This may due to that the simplex transformation in SBSP-NM may work more efficiently in lower-dimensional stochastic search, while SPSA shows its advantage in higher-dimensional stochastic search by the usage of the approximation of the gradient.

**Varying summation constraint $B$:** We varied $B$ from $0.05 ||p||_1$ to $0.25 ||p||_1$. We set the seed size to be 10% of the respective number of nodes and set the other parameters in the same manner as before. Figure 5 shows that SPSA and SBSP-NM perform better than the other baselines in weak immunization cases, while they may get worse results in strong immunization cases, which need to be improved in the future.

**Runtime comparisons:** We also compare the runtime of the algorithms \{SBSP-NM, SPSA, Degree, Rich-Rich, Pagerank, NetMelt, UNI\} with all used WDD to solve inner maximization problem (because the adoption of GG could not make the programs converge in 72 hours). All programs were run on HTCondor [36]. The runtime includes the time required for running the special algorithm and writing output files, and is included as Table 2. We can see that SBSP-NM and SPSA would consume more runtime than the other baselines, which both result from the stochastic search by optimization techniques in the high-dimensional space of $\theta$. Nevertheless, considering the cost and the performance gain, they could still be useful in non-real-time tasks. What’s more, the runtime and the scalability could also further be improved by relieving the heavy burden of the simulation-based function evaluations, like adopting proxy-based or sketch-based approaches [25], which nevertheless goes beyond the scope of our paper. Moreover, SBSP-NM always shows comparable runtime with SPSA, validating the aforementioned analysis of their computational complexities.

**V. CONCLUSION**

We have studied the problem of controlling the worst-case infection in a network. We first formulated the optimal immunization problem as a constrained min-max problem, and then proposed several effective algorithms based on alternating optimization. To solve the inner layer discrete optimization over $A$ with a fixed value of $\theta$, we proposed a WDD method to select the seed set greedily and efficiently. Next and more importantly, to solve the outer layer continuous optimization over $\theta$ with a fixed value of $A$, we proposed two methods: SBSP-NM and SPSA. Through the simulations on different types of small synthetic graphs and larger-scale real-world datasets, we conclude that the WDD algorithm is computationally feasible for large graphs with acceptable performance. Furthermore, both SBSP-NM and SPSA demonstrate better performance than the other five baselines most of the time, and SBSP-NM usually outperforms SPSA under weak immunization conditions. We hope our work could give some helpful inspirations on the immunization in the worst case by the use of the optimization techniques, especially for decision-makers to deal with the unknown seeds. Note that the initials of $\theta$ in all the simulations of SBSP-NM and SPSA are set as an all-zero vector, which could further improve the performance if replaced by the results of the edge removal baselines, thus to combine the advantages of the stochastic search based on optimization techniques and the edge removal methods based on graph theory. Future research directions include establishing theoretical guarantees based on more tractable formulations and extending the algorithms to other epidemic models and intervention tasks.
ACKNOWLEDGMENT
The authors would like to thank the anonymous reviewers for their valuable and constructive comments that greatly contributed to improving the quality of the paper and the editors for their support during the review process.

REFERENCES

TABLE 2: Runtime comparisons of the algorithms (seconds)

<table>
<thead>
<tr>
<th>algorithm/dataset</th>
<th>NetHEPT-k</th>
<th>ego-Facebook-k</th>
<th>wiki-Vote-k</th>
<th>NetHEPT-B</th>
<th>ego-Facebook-B</th>
<th>wiki-Vote-B</th>
</tr>
</thead>
<tbody>
<tr>
<td>SBSP-NM</td>
<td>40,506</td>
<td>26,112</td>
<td>16,626</td>
<td>40,014</td>
<td>26,406</td>
<td>18,725</td>
</tr>
<tr>
<td>SPSPA</td>
<td>26,501</td>
<td>64,178</td>
<td>25,365</td>
<td>52,770</td>
<td>93,079</td>
<td>40,867</td>
</tr>
<tr>
<td>Rich-Rich</td>
<td>4,256</td>
<td>8,224</td>
<td>3,151</td>
<td>4,051</td>
<td>7,963</td>
<td>2,611</td>
</tr>
<tr>
<td>Paperunk</td>
<td>3,095</td>
<td>8,464</td>
<td>6,111</td>
<td>4,481</td>
<td>8,276</td>
<td>6,338</td>
</tr>
<tr>
<td>NetMelt</td>
<td>4,501</td>
<td>6,873</td>
<td>3,443</td>
<td>4,334</td>
<td>9,162</td>
<td>3,379</td>
</tr>
<tr>
<td>UNI</td>
<td>4,465</td>
<td>8,035</td>
<td>2,008</td>
<td>4,298</td>
<td>7,678</td>
<td>2,034</td>
</tr>
<tr>
<td></td>
<td>148</td>
<td>358</td>
<td>113</td>
<td>142</td>
<td>346</td>
<td>110</td>
</tr>
</tbody>
</table>

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/ACCESS.2019.2962197, IEEE Access

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/.