Journal of Artificial Intelligence Research 69 (2020) 847-879 Submitted 05/2019; published 11/2020

Contrasting the Spread of Misinformation in Online Social Networks

Marco Amoruso MARCO.AMORUSO91@GMAIL.COM
Daniele Anello DANIELE.ANELLO@GMAIL.COM
Vincenzo Auletta AULETTA@UNISA.IT
Raffaele Cerulli RAFFAELE@UNISA.IT
Diodato Ferraioli DFERRAIOLI@UNISA.IT
Andrea Raiconi ARAICONI@UNISA.IT
University of Salerno
Via Giovanni Paolo II 132, 84084 Fisciano (SA), Italy

Abstract

Online social networks are nowadays one of the most effective and widespread tools used to share information. In addition to being employed by individuals for communicating with friends and acquaintances, and by brands for marketing and customer service purposes, they constitute a primary source of daily news for a significant number of users. Unfortunately, besides legit news, social networks also allow to effectively spread inaccurate or even entirely fabricated ones. Also due to sensationalist claims, misinformation can spread from the original sources to a large number of users in a very short time, with negative consequences that, in extreme cases, can even put at risk public safety or health.

In this work we discuss and propose methods to limit the spread of misinformation over online social networks. The issue is split in two separate sub-problems. We first aim to identify the most probable sources of the misinformation among the subset of users that have been reached by it. In the second step, assuming to know the misinformation sources, we want to locate a minimum number of monitors (that is, entities able to identify and block false information) in the network in order to prevent that the misinformation campaign reaches some "critical" nodes while maintaining low the number of nodes exposed to the infection.

For each of the two issues, we provide both heuristics and mixed integer programming formulations. To verify the quality and efficiency of our suggested solutions, we conduct experiments on several real-world networks. The results of this extensive experimental phase validate our heuristics as effective tools to contrast the spread of misinformation in online social networks.

Regarding the source identification step, our approach showed success rates above 80% in most of the considered settings, and above 60% in almost all of them.

With respect to the second issue, our heuristic proved to be able to obtain solutions that exceeded (in terms of number of required monitors) the ones obtained through our MILP-based approach of more than 20% in only few test scenarios. Our heuristics for both problems also proved to outperform significantly some previously proposed algorithms.

1. Introduction

In the last years online social networks have indisputably emerged as a phenomenon of ever-growing relevance. Every day, a vast amount of users access, consume and create new content on popular platforms like Facebook or Twitter. According to recent estimates, 2.23

billion people (two-thirds of the world population with access to internet) use Facebook (Frier, 2018).

Online social networks offer a venue to quickly and conveniently exchange information among a single user and other individuals or entities with which the user has established a bond of friendship or trust. The set of users and of these relationships constitute the so-called *social graph*, on which a content considered of high general interest can easily become *viral*, that is, spread from its originating sources to a vast audience and possibly become a subject of public debate. However, the s ame ease of communication and trust relationships also favor the diffusion of inaccurate or entirely false information. Such type of content can be introduced in the network either by mistake or with malicious intents, such as attracting traffic towards specific social profiles or websites in order to increase advertising revenue or influence the public opinion formation.

Misinformation can have very undesirable or even dangerous consequences.

It is well-known, for instance, that misinformation about the side-effects of vaccinations is leading many parents to refuse immunization for their children, putting herd immunity at risk (Hotez, 2016; Lewandowsky et al., 2012). During the Ebola outbreak in 2014, Twitter became a vector for the diffusion of panic and incorrect, harmful medical advice. Indeed, rumors spread about the diffusion of the disease in the US (Luckerson, 2014) and alleged curative powers of salt water (Vijaykumar et al., 2018). Similarly, misinformation has been and is currently being spread online about the COVID-19 pandemic (Mian & Khan, 2020). According to recent surveys, more than 30% U.S. adults claim to often notice entirely false political news on social media, and more than 50% see inaccurate ones (Shin et al., 2018). The exposition to political misinformation has been linked to influence on voting decisions and lack of trust in the institutions (Einstein & Glick, 2015; Weeks & Garrett, 2014; Pogue, 2017). False information coming from social media may also cause instability or manipulations of the financial markets (Ferrara et al., 2016; Jin et al., 2013).

The problem of contrasting the spread of misinformation in an online social network is complex and multi-faceted. We can identify three main steps:

(i) recognize misinformation;

(ii) identify misinformation sources;

(iii) limit their ability to diffuse fake or inaccurate news.

In this paper we focus on the last two points. We consider a scenario where misinformation has already been diffused in the network and administrators have been able to recognize it and find the set of the *infected* users, where we use the term *infected* to refer to a user that has been reached by misinformation, and that possibly contributed (in good faith or not) to disseminate it. We want to identify the sources of misinformation and limit their ability to continue in diffusing misinformation in the network.

Identifying sources is crucial in contrasting misinformation, since it allows network administrators to understand the ultimate goals of its creators, recognize their targets, take actions against the guilty nodes or orchestrate effective strategies for containing its diffusion.

Due to the size of online social networks, recognize misinformation sources can be a very challenging task and in several scenarios it is not possible to identify them for certain. For this reason, social network administrators are reluctant to ban users from the network if they do not have incontrovertible evidence of their misbehavior. A more pragmatic approach is to create a list of “suspects” that can be monitored in order to recognize the fake news that they could inject in the network in an early stage and thus reduce its effect. As will be discussed in Section 4, our proposed strategy for source identification proved to be able to identify most misinformation sources in the considered test scenarios, and an even higher number of them can be found by looking at nodes in their close proximity. This would allow us to obtain highly reliable lists of such suspect users.

The control can be performed through *monitors* placed on users to parse all their activities, recognize misinformation and block it. With monitors, we refer to either human or software-controlled accounts able to detect false information and report it.

Given the huge number of users in online social networks it may be impossible or too expensive to place monitors on all nodes. On the other hand, it could be impossible or undesired to place monitors directly on suspect users, because we cannot have access to them or we do not want to raise their suspicions. Thus, we have to select a set of users to monitor, distinct from the set of suspected sources, such that we can guarantee misinformation injected into the network will be intercepted. Clearly, we cannot guarantee that misinformation is recognized as soon as it is created, but we would like to reduce as much as possible the number of exposed users. Moreover, in several contexts we could have specific users that must be protected from misinformation.

We remark that asking a user to act as monitor would require an explicit consent. Furthermore, in our proposed strategy the network administrator would not need to track or store any user activity, while each monitoring user should just report the detected misinformation, thus mitigating privacy concerns.

1.1 Our Contribution

We now describe in details the problems that we address in this work, and our contribution.

Let $G = (V, E, w)$ be a directed weighted graph, representing the considered social network. For each edge $(i, j) \in E$ among two users, the weight $0 \leq w(i, j) \leq 1$ is interpreted as the probability for user i to transmit a given piece of information to user j.

We model the diffusion of (mis-)information on this network through the *Independent Cascade Model*, that has been first investigated by Goldenberg et al. (2001a, 2001b) and by Kempe et al. (2003). In this model, it is assumed that there is a set $S \subseteq V$ of nodes, denoted *sources* that originated the cascade. The cascade then proceeds in discrete time steps: at step t nodes that have been infected at time $t-1$ will try to infect their neighbors. Formally, let $V_{inf}(t) \subseteq V$ be the subset of users reached by an information (in our case, infected by misinformation) originating from the sources S in a given time period t. Furthermore, let $V_{inf}^{t} = \bigcup_{t' \leq t} V_{inf}(t')$ be the set of all nodes reached within time t. Clearly, $V_{inf}(0) = V_{inf}^{0} = S$. For any given value of t, given a node $i \in V_{inf}(t-1)$ and a node $j \in V \setminus V_{inf}^{t-1}$ such that $(i, j) \in E$, $j \in V_{inf}(t)$ with probability $w(i, j)$. The infection process stops at the earliest time t^* such that $V_{inf}(t^*) = \emptyset$ (no new node is infected). In the following, we refer to $V_{inf}^{(t^*-1)}$ as V_{inf}.

Both the Independent Cascade model and epidemics models (which will be discussed in Section 1.2.1) have been used in the literature to model the spread of information through social media, regardless of its truthfulness. Even assuming that the misinformation sources have malicious intents, which may not always be the case, many users contribute to spread it further because they assume it to be true, therefore the diffusion of true or false information should not be modeled differently. However, as already mentioned and as analyzed by Vosoughi et al. (2018), false information tends to have a greater degree of novelty and to stimulate emotional reactions more than true one, which leads to a higher sharing probability. This would therefore lead to higher values for the weights w in the case of willingly deceitful misinformation. Another difference is that we may expect legit information to be generally verifiable, and possibly include references to its original sources, hence source identification is generally an easier task in this case.

1.1.1 Source Identification

The first problem that we consider in this paper is the *Source Identification (SI) problem.* Here, given the graph G and the set V_{inf} of nodes infected at the end of an Independent Cascade originated by a (possibly unknown) number k of sources, the aim is to return the set S of sources. Clearly, since the diffusion of information is probabilistic, it is impossible to optimally identify these sources for each possible input. However, our objective would be to design an approach able to find the correct sources (or at least a large subset of them) with high probability. Ideally, one may think to solve the following optimization problem: given the sets G and V_{inf} as above, and a bound k on the number of sources, find the most probable set of k sources from which an Independent Cascade can terminate with the set of infected nodes being exactly V_{inf}. Unfortunately, this problem has been showed to be very hard to solve: it is in fact NP-hard even to produce a β-approximate solution to this problem, for every $\beta > 1$ (Lappas et al., 2010).

For this reason, we here propose an heuristic approach to the problem. The main idea behind our work is to reduce our problem to the Maximum Spanning Branching problem, i.e., the problem to find the forest of maximum probability in the subgraph of G defined by nodes in V_{inf}. Note that this problem is known to be solvable in polynomial time (Chu & Liu, 1965; Edmonds, 1967; Camerini et al., 1980).

The idea behind our approach has been partially introduced already by Lappas et al. (2010). However, they did not work the details of the algorithm, and thus they failed in fully appreciating how this approach can be effective for the problem of source identification. Indeed, we run extensive experiments on real-worlds networks, that shows that our approach very often returns a small set of nodes that contains many, if not all, the real sources, and it outperforms the most effective alternatives known in literature.

Unfortunately, despite their polynomial-time running time, the known algorithms for solving the Maximum Spanning Branching problem turn out to require very large running time, and thus they are infeasible to use with very large social networks. To address this issue, we provide a mixed integer linear programming (MILP) formulation, that turns out to be more time-efficient and effective in recognizing most of the misinformation sources.

1.1.2 Monitor Placement

A correct identification of sources enables to implement efficient strategies to limit their ability to diffuse misinformation.

Zhang et al. (2015a) suggest to use monitors to limit the spread of misinformation originated from a set of known (or suspected) sources. The role of these nodes should be to filter the information they receive and block what they recognize as misinformation. Their goal is to use as few monitors as possible, since they may be expensive to deploy and/or maintain, and place them as close as possible to the sources in order to limit the number of nodes reached by misinformation.

Specifically, Zhang et al. (2015a) considered the following problem, named τ-Monitor Placement. Let G be a network, S be a set of misinformation sources and t be a single target node that we have to protect from misinformation. A set of monitors placed in a set $M \subseteq V$ of vertices detects misinformation if there is at least one path from a node in S to a node in M on which there is a successful diffusion. Specifically, we denote as $L_{S,M}$ the set of all paths whose starting endpoints are in S and the final endpoints are in M, and as $D(S, M)$ the probability that monitors in M detect the misinformation originated in S. The τ-Monitor Placement problem then asks for a subset M of vertices chosen among the vertices at distance at most δ from S, such that $t \notin L_{S,M}$ and the misinformation detection probability $D(S, M) \geq 1 - \tau$. Zhang et al. (2015a) proved that the τ-Monitor Placement problem is $\#P$-complete and presented a heuristic to compute a monitor placement.

In this paper we consider a generalization of the τ-Monitor Placement problem, called *Monitor Placement (MP) problem*. Here, given G, the set S of (identified or suspected) misinformation sources, the (possibly empty) set T of target nodes and an integer k, the aim is to identify a minimum cardinality set of nodes $M \subseteq V$ on which monitors should be installed in order to avoid information originating from any element of S to reach any element of T. Furthermore, we require no more than $|S| \leq k \leq |V| - |T|$ nodes to be infected overall. Finally, it is a realistic assumption that in most practical scenarios monitors cannot be placed directly on sources, hence we also require that $M \cap S = \emptyset$.

It is immediate to check that multiple extensions to the τ-Monitor Placement problem are introduced by our formulations. First, we assume to have a set T of target nodes to protect from misinformation (not only a single node). Moreover, we require that whenever misinformation spreads over the network starting from the known set S of sources, then it will be detected and blocked by monitors in M before it reaches nodes in T, i.e. $\tau = 0$. Finally, in order to limit more effectively the spread of misinformation, we put an explicit bound on the number of nodes that can receive misinformation before it is blocked by monitors. Specifically, we require that the number of nodes in $V \setminus (S \cup M)$ that lie on paths in $L_{S,M}$ (these are the only nodes that can be reached by misinformation) is upper bounded by k. This requirement generalizes and strengthens the request of placing monitors in nodes within distance at most δ from S, considered by Zhang et al. (2015a). Indeed, if the number of nodes close to S is small our requirement achieves the same effect as the τ-Monitor Placement problem, but it allows to keep low the number of infected nodes even if there are many nodes around the sources.

Clearly, the hardness result for the τ-Monitor Placement problem shown by Zhang et al. (2015a) extends to our problem. Moreover, since our problem is much more constrained than the τ-Monitor Placement problem, we should expect that more monitors will be required and their placement would be more difficult to compute. Nevertheless, we here propose an heuristic for the MP problem, based on the concept of k–unbalanced cut (Hayrapetyan et al., 2005), and we show that its performances are comparable (and in some cases even better) to the ones of the approach by Zhang et al. (2015a) both in terms of number of monitors and of computation time.

To further evaluate the quality of the solutions provided by the heuristic, we provide a MILP-based method for the problem, and compare the results of our heuristic against this benchmark: the results confirm the quality of our approach.

1.2 Related Works

Here, we resume the main proposed contributions for both problems of source identification and of limiting misinformation diffusion. Several authors have also faced detection of misinformation (Hamidian & Diab, 2016; Wu et al., 2017; Shu et al., 2017; Zubiaga et al., 2018; Ko et al., 2019; Yang et al., 2019), whose discussion is out of the scope of this work.

1.2.1 Source Identification

The first works on the source identification problem used simple epidemics models: that is, they describe the information diffusion process as an infection disease spreading over the population. These works adopted centrality measures to identify the sources of the diffusion process. In particular, Comin and da Fontoura Costa (2011) run several experiments to compare degree, betweenness, closeness, and eigenvector centrality in identifying the sources of the misinformation.

Along the same line of research, Shah and Zaman (2011) proposed a new centrality measure, named *rumor centrality*, and showed that it outperforms all the previously considered centrality measures.

Rumor centrality revealed to be very influential and it has been largely used, and extensions and generalizations have been proposed to identify sources of epidemics spread in several different settings, varying in the number of sources, the topology of the network and the coarseness of information about the set of affected nodes that is known to the algorithm. We refer interested readers to the survey of Jiang et al. (2014) and references therein.

Epidemics models assume that there exists a global parameter that describes the probability that a user is infected by a neighbor. While this assumption simplifies the computational complexity of the model, it fails in describing real-world situations where users are differently bent to accept information from their neighbors. To overcome this difficulty, the Independent Cascade model has been proposed as a generalization of the epidemics model where each edge has its specific activation probability.

Clearly, this generalization makes the problem extremely more complex to deal with. Indeed, as indicated above, Lappas et al. (2010) prove that it is NP-hard even to produce a β-approximation, for every $\beta > 1$, for the Source Identification problem when information diffuses according to an Independent Cascade model.

This hardness result leaves us only two possible research directions: either we focus on special network topologies where the problem is tractable or we consider general heuristics with good experimental performances. Lappas et al. (2010) follow the first direction and study the Source Identification problem on tree networks.

Nguyen et al. (2012a), instead, follow the second direction and propose efficient heuristics for identifying sources of misinformation in general networks. In this work we build upon their contribution. We present a new heuristic approach whose performances turn out to be much better than previously presented algorithms. Moreover, we remark that both algorithms by Lappas et al. (2010) and Nguyen et al. (2012a) need to know in advance the number k of sources to find. Our heuristic, instead, works well even if the number of sources is not known.

1.2.2 Limiting the Diffusion of Misinformation

Two main approaches have been proposed in literature to address this problem. The first one, proposed by Budak et al. (2011), requires that a true information campaign is initiated from a subset of highly influential nodes. In this way, the diffusion of misinformation and true information proceeds in parallel, except that nodes that have received the true information will be immune to the misinformation and will not transmit it. However, in order to have a true information campaign that would be effective in the tentative of limiting misinformation, one must carefully choose the seeds from which the diffusion starts. We remark that this approach requires perfect knowledge of the sources of misinformation in order to correctly selecting the seeds of the contrasting campaign.

Budak et al. (2011) studied the computational complexity of this problem and proposed some preliminary solutions. A similar approach has been taken by Nguyen et al. (2012b) and Li et al. (2013). They introduced the Node Protector problem which aims to find the smallest set of highly influential nodes whose decontamination with good information helps to contrast the viral spread of misinformation. They give inapproximability results and propose greedy approximation algorithms. Heuristics improving on this algorithm have been proposed by Tong et al. (2020). Variants of the problem have been also considered by Fan et al. (2013), that focused on a community-based network, and by Zhang et al. (2015b) and Hosni et al. (2018) that, instead, not only aim to minimize the spread of misinformation, but also to maximize the diffusion of true information.

The second approach consists in individuating nodes or links in the network that would block the diffusion of the misinformation. Many works have been proposed on this topic, varying in the objective to optimize, in the model for information diffusion, and in the technique adopted for computing these blocking nodes or monitors. For example, Aspnes et al. (2006) considered a setting with a single random source, no target, and no threshold on the number of infected nodes, but the objective is to minimize the sum of the number of monitors and the number of infected nodes. Similarly, Kimura et al. (2008) assumed that the source is unique and selected at random, no target exists, and the goal is to remove links in order to minimize the contaminantion degree, i.e., the expectation over the choice of the source of the number of nodes infected after the cascade. A similar problem has been also addressed by Zhang et al. (2016). Habiba et al. (2010) and Kuhlman et al. (2010) proposed heuristics for the problem, based on centrality measures and on covering and potential approaches, respectively. Both these works do not consider the presence of a target to protect. Moreover, Kuhlman et al. (2010) also focus on a simpler deterministic diffusion model.

As indicated above, the setting that is more closely related to the one considered in this work has been proposed by Zhang et al. (2015a). Namely, they propose to place monitors over the network that are able to detect misinformation and block it. However, a good monitor placement should satisfy two requirements: on one side, we would like to place as few monitors as possible, on the other side, we would like that our monitors limit the number of nodes exposed to misinformation. These two discording goals make the problem very difficult. Indeed, Zhang et al. (2015a) proved that the problem is $\#P$-complete, and proposed an heuristic for placing monitors so that misinformation is detected with high probability before it reaches target nodes.

In this work, we strengthen the model proposed by Zhang et al. (2015a), by putting more stringent requirements on the number of nodes exposed to misinformation and requiring that misinformation is always detected (more details in Section 3). Nevertheless, experiments show that our heuristic has performances comparable or even better than the algorithm proposed by Zhang et al. (2015a).

1.3 Organization of the Paper

The rest of the work is organized as follows. Our proposed approaches and formulations are presented and discussed in Sections 2 and 3, respectively. The results of our computational experience are reported in Section 4.

2. Source Identification

This section is devoted to the presentation of the proposed approaches for the Source Identification problem, with either a known or an unknown number of sources. The underlying idea to model the problem and the proposed algorithm are presented in Section 2.1, while a MILP formulation for the same problem is discussed in Section 2.2.

2.1 The Approach

As mentioned, we model source identification as a maximal spanning branching problem. Given the input network $G = (V, E, w)$ and the infected users subset $V_{inf} \subseteq V$, we consider the subgraph $G_{inf} = (V_{inf}, E_{inf}, w_{inf})$ induced in G by V_{inf}. That is, for each $(i, j) \in E$, $(i, j) \in E_{inf}$ if and only if $i \in V_{inf}$ and $j \in V_{inf}$. Furthermore, $w_{inf}(i, j) = w(i, j)$ for each $(i, j) \in E_{inf}$. It is straightforward to understand that infection could only have spread on nodes and edges belonging to G_{inf}, since edges in $V \setminus V_{inf}$ were not exposed to it.

In order to help the reader in understanding how our approach works, we first start by considering the simple case in which misinformation starts from a unique source. Then, we show how the deployed ideas can be extended in order to take in account multiple sources.

2.1.1 Warm-up: Single Source

An arborescence of the graph G_{inf} is a directed subgraph T on a subset $V' \subseteq V_{inf}$ of vertices of G_{inf}, such that there is a distinguished node $r \in V'$, called *root*, and a single directed path from r to every other vertex in V'. A *spanning arborescence* of G_{inf} is an arborescence containing all the vertices of G_{inf}. Roughly speaking, an arborescence is a directed tree and a spanning arborescence is a directed spanning tree.

The weight of an arborescence $T = (V', E')$ is the sum of the weights of the edges in T, i.e., $W(T) = \sum_{(u,v)\in E'} w_{inf}(u, v)$. The *maximum spanning arborescence* is a spanning arborescence of maximum weight.

Notice that the probability that a given node is the source is the sum of the weights of all the arborescences rooted in that node. Unfortunately, computing this probability is not computationally affordable. Still the weight of a spanning arborescence T turns out to be a measure of the probability that the *misinformation spreads according to* T, i.e., that an Independent Cascade starting from the node $r(T)$, i.e., the root of T, has each node $v \in V_{inf}$ infected by its unique predecessor in T. Hence, the root r^* of the maximum spanning arborescence T^* of the subgraph G_{inf} is a very natural candidate for being the source of misinformation.

We note that the problem of computing spanning arborescences is very well-studied and a lot of algorithms are known both for general networks and for specific classes of graphs. In particular, it is possible to efficiently compute the maximum spanning arborescence of a graph through the *Chu-Liu/Edmonds algorithm* independently proposed by Chu and Liu (1965) and by Edmonds (1967).

2.1.2 Multiple Sources

Clearly, the assumption that misinformation originated in only one source is too restrictive and in this paragraph we show how to relax it.

The approach described for the the single source case works by computing the maximum spanning arborescence. However, this does not help when there are multiple sources. Indeed, even if we assume that misinformation diffuses exactly along the edges of that arborescence, it is not clear how to select sources out of the root of the arborescence. For example, if we select nodes that are close to the root, we are implicitly limiting the influence of the root node, but nodes that are far away from the root may be scarcely influential.

However, the arborescence approach can still be fruitful. Suppose that misinformation starts from k different sources and proceeds as in k parallel threads. Then we can model the diffusion process by simply considering multiple arborescences, up to one for each source. Hence, if we can identify these diffusion trees, we can choose their roots as natural candidates for misinformation sources, exactly as done when we have a single arborescence. This motivates us to use branchings in places of arborescences.

A *branching* of the graph G_{inf} is a forest of disjoint arborescences. In a natural way, we can define the *maximum spanning branching* of G_{inf} as a set of disjoint arborescences containing all the vertices of G_{inf} and such that the sum of their edges' weights is maximum.

Our approach consists then of computing the maximum spanning branching of G_{inf}. The roots of the identified arborescences correspond to the identified sources (that is, the nodes that we suspect to belong to S). Additionally, as mentioned, we may require the identification of a predefined number of sources $k' > 0$; in this case, we look for the maximum-weight spanning branching such that $|\mathcal{T}| = k'$.

As for the case of arborescences, algorithms are known to efficiently computing a maximum spanning branching (e.g., Camerini et al., 1980). Nevertheless, the algorithms, despite their polynomial running time, turns out to be very time expensive when run on large social networks. For this reason, in next section we provide a more efficient MILP formulation that we adopted in our experiments.

2.2 MILP Formulation

We first describe how to model the case in which the number of sources to be identified is fixed to an integer value $k' > 0$. We modify the definition of G_{inf} by adding to it a *super-source* node s_0, which is connected to every other node i belonging to V_{inf} through an edge $(s_0, i) \in E_{inf}$. We do not need to define a specific weight for these additional edges, since they will not be considered in the evaluation of any objective function; however, for sake of consistency, we may assign an equal weight c to each of them.

We then look for a maximum weight arborescence of G_{inf} rooted in s_0, constraining the super-source degree to k'. As mentioned, we exclude from the objective function computation the weight of the k' edges outgoing from s_0. It is easy to understand that every node connected to s_0 in the solution corresponds to the root of an arborescence in the maximum spanning branching problem with k' components.

The proposed mixed integer linear programming formulation (called **[SI]**) is the following:

$$\textbf{[SI]} \max_{(i,j)\in E_{inf}: i\neq s_0} w_{inf}(i,j)x_{ij} \tag{1}$$

$$s.t.$$

$$\sum_{(s_0,i)\in E_{inf}} x_{s_0 i} = k' \tag{2}$$

$$\sum_{(s_0,i)\in E_{inf}} f_{s_0 i} = |V_{inf}| - 1 \tag{3}$$

$$\sum_{(j,i)\in E_{inf}} f_{ji} - \sum_{(i,k)\in E_{inf}} f_{ki} = -1 \qquad \forall i \in V_{inf} \setminus \{s_0\} \tag{4}$$

$$\sum_{(j,i)\in E_{inf}} x_{ji} = 1 \qquad \forall i \in V_{inf} \setminus \{s_0\} \tag{5}$$

$$x_{ij} \leq f_{ij} \qquad \forall (i,j) \in E_{inf} \tag{6}$$

$$(|V_{inf}| - k')x_{ij} \geq f_{ij} \qquad \forall (i,j) \in E_{inf} \tag{7}$$

$$x_{ij} \in \{0,1\} \qquad \forall (i,j) \in E_{inf} \tag{8}$$

$$f_{ij} \geq 0 \qquad \forall (i,j) \in E_{inf} \tag{9}$$

For each $(i,j) \in E_{inf}$, the binary variable x_{ij} assumes value 1 if and only if the edge is chosen to belong to the solution, while f_{ij} is a flow variable used to ensure the connectivity of the solution. The objective function (1) maximizes the weight of the found arborescence. Constraint (2) imposes s_0 to have k children nodes. Constraint (3) states that s_0 must produce one flow unit for each node in $V_{inf} \setminus \{s_0\}$ and, along with flow conservation constraints (4), ensures that each node in the infected network is reached. Constraints (5) ensure that each node in $V_{inf} \setminus \{s_0\}$ has exactly one parent. Finally, Constraints (6)-(7) impose a positive flow on each and only edge chosen to belong to the solution. In (7), $|V_{inf}| - k'$ is a tight upper bound on the maximum flow on an edge for a feasible solution. Indeed, if in a solution there are $k'-1$ children of s_0 that are leaves (corresponding to trivial arborescences with a single node), then $|V_{inf}| - 1 - (k' - 1)$ flow units must reach the remaining one.

Clearly, each node $i \in V_{inf} \setminus \{s_0\}$ is identified as one of the k' sources if and only if $x_{s_0 i} = 1$ in the optimal solution.

In order to model the case in which the number of arborescences composing the branching is unknown, it is sufficient to remove Constraint (2) and substitute the $|V_{inf}| - k'$ bound in (7) with $|V_{inf}| - 1$. Indeed, this is a tight bound in the case in which s_0 has a single child (corresponding to a branching composed of a single arborescence).

3. Monitor Placement

In this section, approaches for the Monitor Placement problem are discussed. The proposed heuristic is the object of Section 3.1, while a MILP-based approach is presented in Section 3.2.

3.1 The Approach

Given the set of (identified or suspected) misinformation sources $S \subset V$, and set of sensitive target nodes $T \subset V$ ($S \cap T = \emptyset$), the aim is to identify a minimum cardinality set of nodes $M \subseteq V$ on which monitors should be installed in order to avoid information originating from any element of S to reach any element of T. Furthermore, we require no more than $|S| \leq k \leq |V| - |T|$ nodes to be infected overall. Finally, it is a realistic assumption that in most practical scenarios monitors cannot be placed directly on sources, hence we require $M \cap S = \emptyset$.

3.1.1 Monitors and Cuts

Let us start by considering a simple setting where we have a network represented by the graph $G = (V, E, w)$, with $w(u, v) = 1$ for each edge $(u, v) \in E$, a single source s of misinformation and a single target t to protect. Let C be a (s, t)-cut of the graph G. By definition of cut, if we remove from G all edges in C then there will be no paths from s to t. Thus, by placing monitors in the endpoints of the edges in C we can guarantee that all the information diffused by s will be blocked before it reaches t.

Observe that the number of monitors required by this approach depends on the size of the cut. Therefore, in order to minimize the number of required monitors we need an $(s - t)$-cut of minimum size. However, our requirements are not only to protect t from the misinformation but also to have a small number of nodes exposed to misinformation. Observe that a minimum cut does not give any guarantee on the number of nodes that can be reached by the misinformation before monitors detect it. Suppose, for example, that the minimum cut contains only edges adjacent to t. In this case, by placing monitors on the endpoints of these edges we have that only the target node t and the nodes hosting the monitors are protected by the misinformation. Thus, we have to impose another constraint to our cut: the set of nodes reachable from s after the removal of the edges in the cut must be small. To meet these additional requirement we will consider *unbalanced cuts*.

Formally, given a graph G, a source s, a target t, and an integer k, a *k–unbalanced (s, t)-cut* is a partition of the nodes of the graph in two sets, L and R, such that $s \in L$, $t \in R$, and $|L| \leq k$. The size of the cut (L, R) is given by the number of edges that have an endpoint in L and the other endpoint in R, i.e. $W(L, R) = |\{(u, v) \in E \colon u \in L, v \in R\}|$. A minimum k–unbalanced (s, t) cut is a cut (L^*, R^*) such that $W(L^*, R^*) = \min_{L,R\colon s \in L, t \in R, |L| \leq k} W(L, R)$. Roughly speaking, a minimum k–unbalanced (s, t)-cut is a (s, t)-cut of minimum size among all the $(s - t)$-cuts where the source side is bounded to contain at most k nodes.

Interestingly, a polynomial time algorithm is known for computing a minimum k–unbalanced cut for every graph G (Gallo et al., 1989; Hayrapetyan et al., 2005). The basic idea of this algorithm consists in finding a minimum cut in a graph G^α obtained from G by adding edges of weight α from all the nodes of the graph to t. Clearly, if $\alpha = 0$ then

$G^\alpha = G$. If $\alpha > 0$, instead, the size of a cut (L, R) of G^α is given by the size of the same cut in the original graph G plus an additive factor of $\alpha|L|$. As α increases the size of L becomes more and more relevant with respect to the size of the cut. Hence, if α is sufficiently large, then a cut of G^α becomes a k–unbalanced cut of G.

Even if this algorithm seems to be very "expensive" in computational terms (we could compute a lot of cuts to find the correct value of α), Gallo et al. (1989) proved that using the *parametric-flow technique* we can efficiently build a new cut on top of the previous one. Moreover, Gallo et al. (1989) described a procedure to compute the next α that rapidly converges to a value that produces a minimum k–unbalanced cut.

3.1.2 The Heuristic

Even if the core of our solution is given by the computation of an unbalanced cut, as described above, there are still several aspects and optimizations that have to be addressed in designing our heuristic.

First of all, the approach described above was designed for a single source - single target scenario on an unweighted graph (actually, we assumed that all edge weights are equal). Here, we will explain how we can adapt our approach to many sources - many targets scenarios on weighted graphs. We address the problem of many sources and targets through a *source and target contraction.* Let $G = (V, E, w)$ be a weighted graph representing our network, let S be the set of known or suspected misinformation sources (as, e.g., the one returned by our source identification algorithm) and T be the set of targets. Then we consider a new graph $G^* = (V^*, E^*, w^*)$ in which we contract all sources in a single node s^*, and all targets in a single node t^*, i.e., $V^* = (V \setminus (S \cup T)) \cup \{s^*, t^*\}$ and $E^* = \bigcup_{i=1}^{5} E_i^*$, where

- $E_1^* = \{(u, v)\colon (u, v) \in E \text{ and } u, v \in V^* \setminus \{s^*, t^*\}\}$,
- $E_2^* = \{(s^*, v)\colon (u, v) \in E, u \in S \text{ and } v \in V^* \setminus \{s^*, t^*\}\}$,
- $E_3^* = \{(u, s^*)\colon (u, v) \in E, u \in V^* \setminus \{s^*, t^*\} \text{ and } v \in S\}$,
- $E_4^* = \{(t^*, v)\colon (u, v) \in E, u \in T \text{ and } v \in V^* \setminus \{s^*, t^*\}\}$, and
- $E_5^* = \{(u, t^*)\colon (u, v) \in E, u \in V^* \setminus \{s^*, t^*\} \text{ and } v \in T\}$.

As for the weights, we clearly set $w^*(u, v) = w(u, v)$ for every $(u, v) \in E_1^*$. For edges $(s^*, v) \in E_2^*$, let $C(v)$ be the set of sources that are connected with v in the original graph, i.e., $C(v) = \{s \in S\colon (s, v) \in E\}$. Then, $w^*(s^*, v) = 1 - \prod_{s \in C(v)}(1 - w(s, v))$, that is the probability that at least one of the source nodes transmit the misinformation to v. Similarly, for edges $(u, s^*) \in E_3^*$, let $D(u)$ be the set of sources at which u is connected in the original graph, i.e., $D(u) = \{s \in S\colon (u, s) \in E\}$. Then, $w^*(u, s^*) = 1 - \prod_{s \in D(u)}(1 - w(s, v))$. A similar approach can be taken for edges in E_4^* and E_5^*.

The graph G^* has now a single source s^* and a single target t^*. Since this graph is weighted we need to specify which cuts we should compute. A natural choice would be to take minimum cuts (i.e. cuts that minimize the sum of the weights of their edges). However, since an edge weight represents the probability that information flows on that edge, placing monitors on the endpoints of a minimum cut would mean to place monitors on endpoints of edges where it is unlikely that the misinformation spreads. Monitoring these edges can then be a useless waste of resources.

We propose, instead, to place monitors on edges with large transmission probability. This indeed would also help in reducing the number of nodes infected by misinformation: in fact, not only the monitor placement guarantees that target nodes will not be reached by misinformation and there are no more than k nodes reached by misinformation, but it may be also the case the number of infected nodes is much less than k since edges between nodes in the size of the cut containing the sources have small transmission probabilities. In order to achieve this goal, we run the minimum k–unbalanced (s^*, t^*)-cut procedure on the graph $\hat{G} = (V^*, E^*, \hat{w})$, where edge weights are integers and they are inversely proportional to their weights in G^*. We observe that the use of integer weights has the positive side effect to make easier to compute the next α to use in the computation of the unbalanced cut.

Another optimization is related to the placement of monitors in the endpoints of the unbalanced cut's edges. In our informal discussion for the single source case we stated that monitors can be placed on all the endpoints of the cut's edges. However, it is clearly unnecessary to place monitors on all these nodes. Instead, we will use a more clever placement algorithm in order to reduce the number of monitors. Specifically, given a cut (L, R) of $\hat{G}$, where L is the side of the cut that contains s^*, we consider the unweighted graph $C = (W, F)$ induced by the edges of (L, R), i.e, $W = \{u \in L\colon (u, v) \in E^*, v \in R\} \cup \{v \in R\colon (u, v) \in E^*, u \in L\}$ and $F = \{(u, v) \in E^*\colon u \in L, v \in R\}$. Then, we compute a *minimum vertex cover* M of C and place monitors in all the nodes in M. Notice that, since C is a bipartite graph, it is possible to compute its minimum vertex cover in polynomial time (via a reduction to a problem of min cut/max flow).

Summarizing, our procedure works as described in Algorithm 1. Notice that our heuristic may place a monitor in s^*. In this case, we simply replace s^* with all its neighbors.

```
**Input:** Graph $G$, Sources $S$, Targets $T$, and integer $k$.
**Output:** Monitor vertices $M$.
1 $G^*, s^*, t^* =$ SourceContraction$(G, S, T)$
2 $\hat{G} =$ WeightConversion$(G^*)$
3 $(L, R) =$ UnbalancedCut$(\hat{G}, s^*, t^*, k)$
4 $C =$ BipartiteGraphFromCut$(L, R)$
5 $M =$ VertexCover$(C)$
6 **return** $M$
```

Algorithm 1: Algorithm for monitoring placement

3.2 MILP Approach

In order to validate the performances of the heuristic described by Algorithm 1, we provide a MILP method that provides a solution to the problem without the need of artificially transform the input (i.e., the MILP does not need to contract the sources and to convert weights). Clearly, the MILP resolution turns out to be much slower than our proposed heuristic, and it does not scale with the size of the network.

As above, the problem is solved in two separate steps. In the first one, we look for a k-unbalanced S-T cut. That is, in order to separate sources from targets, we look for a partition of V in two disjoint subsets L and R ($L \cup R = V$, $L \cap R = \emptyset$), such that $S \subseteq L$, $T \subseteq R$ and $|L| \leq k$. As above, among all feasible solutions, we look for the one that maximizes the weight of the edges that cross the cut from L to R, in order to reduce the probability of infection for the nodes contained in L. Once the cut is found, monitors could be possibly placed in all endpoints of edges crossing the cut. However, as previously explained, it is possible to reduce the number of required monitors by only placing one in a single endpoint of any such edges. Formally, given the sets L and R, we construct the unweighted graph $G_{cut} = (V_{cut}, E_{cut})$, with $E_{cut} = \{(i,j) \in E | i \in L, j \in R\}$ and $V_{cut} = \{i \in L : (i,j) \in E_{cut}\} \cup \{j \in R : (i,j) \in E_{cut}\}$, and look for a minimum-cardinality vertex cover of G_{cut} that does not contain elements of S.

Regarding the first step, in more detail, we define $U = V \setminus \{S \cup T\}$ as the set of *undecided* nodes, that is, nodes that in the k-unbalanced cut may belong to either L and R. It is easy to understand that S, T and U define a partition of V. Given the bound on the size of the subset containing S in any feasible solution (that is, $|L| \leq k$), it is also easy to note that $|U \cap L|$ cannot be greater than $k - |S|$, and that therefore $|U \cap R|$ must be at least equal to $|U| - (k - |S|)$. Furthermore, we consider a modified weighting function w', such that each edge weight $w'(i,j)$ is inversely proportional to the original weight $w(i,j)$. We propose the following mathematical formulation **[MP1]**:

$$\textbf{[MP1]} \min_{(i,j)\in E} w'(i,j) y_{ij} \tag{10}$$

$$s.t.$$

$$y_{ij} \geq z_j - z_i \qquad \forall (i,j) \in E \tag{11}$$

$$z_i = 0 \qquad \forall i \in S \tag{12}$$

$$z_i = 1 \qquad \forall i \in T \tag{13}$$

$$\sum_{i \in U} z_i \geq |U| + |S| - k \tag{14}$$

$$z_i \in \{0,1\} \qquad \forall i \in V \tag{15}$$

$$y_{ij} \geq 0 \qquad \forall (i,j) \in E \tag{16}$$

Each binary variable z_i is equal to 0 if $i \in V$ is chosen to belong to L, and 1 if it will belong to R. Variables y_{ij} will be equal to 1 if (i,j) crosses the cut ($i \in L, j \in R$) and 0 otherwise.

The objective function (10) minimizes the weight of the edges crossing the cut. By effect of the w' function, this favors the selection of edges with high transmission probability. Constraints (11) impose y_{ij} to be at least equal to 1 if $i \in L$ and $j \in R$. In every other case, y_{ij} is free to assume (also by effect of Constraints (16)) any value ≥ 0. Given that all weights are strictly positive, and the minimization imposed by (10), in the optimal solution the y_{ij} variables will be set to their minimal feasible values, that is, $y_{ij} = 1$ if $i \in L$ and $j \in R$ and $y_{ij} = 0$ otherwise, as desired.

Constraints (12) and (13) make sure that all sources and all targets belong to L and R, respectively. Finally, Constraint (14) imposes the above illustrated lower bound on the number of nodes in U that must belong to R.

Finally, after identifying the cut and constructing the graph G_{cut} as described above, we model the problem of identifying the minimum number of monitors and their locations as follows:

$$\textbf{[MP2]} \min_{i \in V_{cut}} h_i \tag{17}$$

$$s.t.$$

$$h_i + h_j \geq 1 \qquad \forall (i,j) \in E_{cut} \tag{18}$$

$$h_j = 1 \qquad \forall (i,j) \in E_{cut} | i \in V_{cut} \cap S \tag{19}$$

$$h_i \in \{0,1\} \qquad \forall i \in V_{cut} \tag{20}$$

Each binary variable h_i is equal to 1 if and only if a monitor should be placed in node i. Constraints (18) impose the installation of a monitor in at least an endpoint of each edge, while Constraints (19) state that a monitor should be installed on each neighbor of nodes in $V_{cut} \cap S$.

4. Experiments

We validated the proposed approaches by performing experiments on a set of freely available benchmark instances, deriving from real-world data. All tests were conducted on a machine running CentOS Linux 7, equipped with an Intel Xeon E5-2650 v3 processor running at 2.3 GHz and 128 GB of RAM. The IBM ILOG CPLEX solver (version 12.6.1) was used to solve the proposed [**SI**], [**MP1**] and [**MP2**] formulations. The algorithms and the Independent Cascade infection model are implemented in Python.

Next subsections contain descriptions of our computational experience for the Source Identification and Monitor Placement problems, respectively.

4.1 Source Identification

Tests related to source identification were conducted on 12 instances. In particular, we considered 10 directed graphs and 2 undirected ones. With respect to directed instances, we considered the Epinions network to make our results comparable with the one given by Nguyen et al. (2012a) and Zhang et al. (2015a). For this graph we adopted an approach for generating transmission probabilities similar to the one used in these papers, i.e. as described by Richardson et al. (2003): to each node u, it has been assigned a quantity $\gamma_u \in [0,1]$ chosen according to a Gaussian distribution with mean 0.5 and standard deviation 0.25; then to an edge (u,v) it is assigned weight $w(u,v)$ uniformly chosen from $[\max\{\gamma_u + \gamma_v - 1, 0\}, \min\{\gamma_u - \gamma_v + 1, 1\}]$. This instance will be referred to as *Epinions-1* from now on.

Furthermore, we considered the following 8 instances coming from the Social category of the Konect database[1]: “Advogato” (from now, referred to as *Advogato*); “Digg Friends” (*Digg*), “Epinions trust” (*Epinions-2*), “Google+” (*Gplus*), “Slashdot Zoo” (*Slashdot*), “Twitter lists” (*Twitter*), “Youtube friendship” (*Youtube-1*), “Youtube links” (*Youtube-2*). Finally, we considered the “Political blogs” (*Polblogs*) network of hyperlinks, available

1. http://konect.uni-koblenz.de

	Advogato		Digg		Epinions-1		Epinions-2		Facebook-1		Facebook-2	
$\|S\|$	nodes	edges	nodes	edges	nodes	edges	nodes	edges	nodes	edges	nodes	edges
2	182.33	1222.33	152.00	1409.00	177.67	2240.00	175.33	2325.00	223.67	454.00	126.00	916.67
3	190.33	1621.00	143.33	3864.00	145.00	1719.67	171.00	1265.33	156.33	346.00	172.67	1398.67
5	203.33	2082.33	118.33	1231.00	136.67	1454.33	185.67	1101.67	170.00	336.67	116.00	1032.00
	Gplus		Polblogs		Slashdot		Twitter		Youtube-1		Youtube-2	
$\|S\|$	nodes	edges	nodes	edges	nodes	edges	nodes	edges	nodes	edges	nodes	edges
2	206.67	251.67	145.67	2193.67	169.00	818.00	154.67	306.67	153.00	727.33	116.67	511.00
3	167.33	173.00	203.67	4004.33	160.67	1331.33	137.00	233.67	197.67	818.67	176.00	595.67
5	177.33	189.00	149.33	3190.67	145.00	725.00	117.33	199.00	150.67	522.67	153.67	571.00

Table 1: Infected networks sizes

through the database of network data maintained by Mark Newman[2], used in a recent study on social influence during elections (Wilder & Vorobeychik, 2018). The two considered undirected graphs are "Facebook (NIPS)" (*Facebook-1*) and "Facebook friendships" (*Facebook-2*), again available through the Konect database. For all mentioned instances, except Epinions-1, a random weight between 0 and 1 was generated for each edge and used as transmission probability.

For each network, to simulate an infection process, we chose at random a subset S of nodes and assumed them to be the misinformation sources. Afterwards, we ran the Independent Cascade diffusion process starting from them. Given a set of infected nodes V_{inf} resulting from the process, $G_{inf} = (V_{inf}, E_{inf})$ is the subgraph induced in the input graph G by V_{inf}, to be used as input instance for the **[SI]** model. We chose 2,3 and 5 as values for $|S|$, and simulated the process for 3 different random choices of the sources for each value of $|S|$ (for a total of 9 simulations for each network). It follows that we obtained 108 different infected networks (9 for each of the 12 input instances). Average sizes of such networks for every choice of instance and $|S|$ are reported in Table 1.

4.1.1 The Results

We first analyze in the detail the performances of our approach when the number of sources to be found is fixed to $k' = |S|$. As mentioned, all instances are solved quickly using the MILP model, indeed computational times reached around 30 seconds for a single test in the worst case. For 9 out 12 instances (that is, excluding Facebook-1, Facebook-2 and Youtube-2), no individual test required more than 3 seconds.

Performances on directed networks are shown in Figure 1. In the figure, the nodes marked as source by the model (that is, children of the super-source s_0) are identified as nodes at distance 0. Given the 9 computational tests run for each instance, the overall number of sources to be identified is 30 (i.e., 2 sources for three simulations, 3 sources for other three simulations, and 5 sources for the remaining simulations). Then, for each instance we report how many of them, in percentage, were correctly identified (on Figure 1 this is the percentage of *true sources* when the value of the distance from identified sources is set to 0).

Overall, we note that at least 80% of the true sources are identified correctly for 6 out of 10 instances (Digg, Epinions-1, Epinions-2, Gplus, Slashdot, Twitter). For Advogato and Youtube-2, 63.33% of the sources are identified correctly (that is, 19 out of 30). For Youtube-1, the true sources at distance 0 are 16. The only result below 50% is related to

2. http://www-personal.umich.edu/˜mejn/netdata/

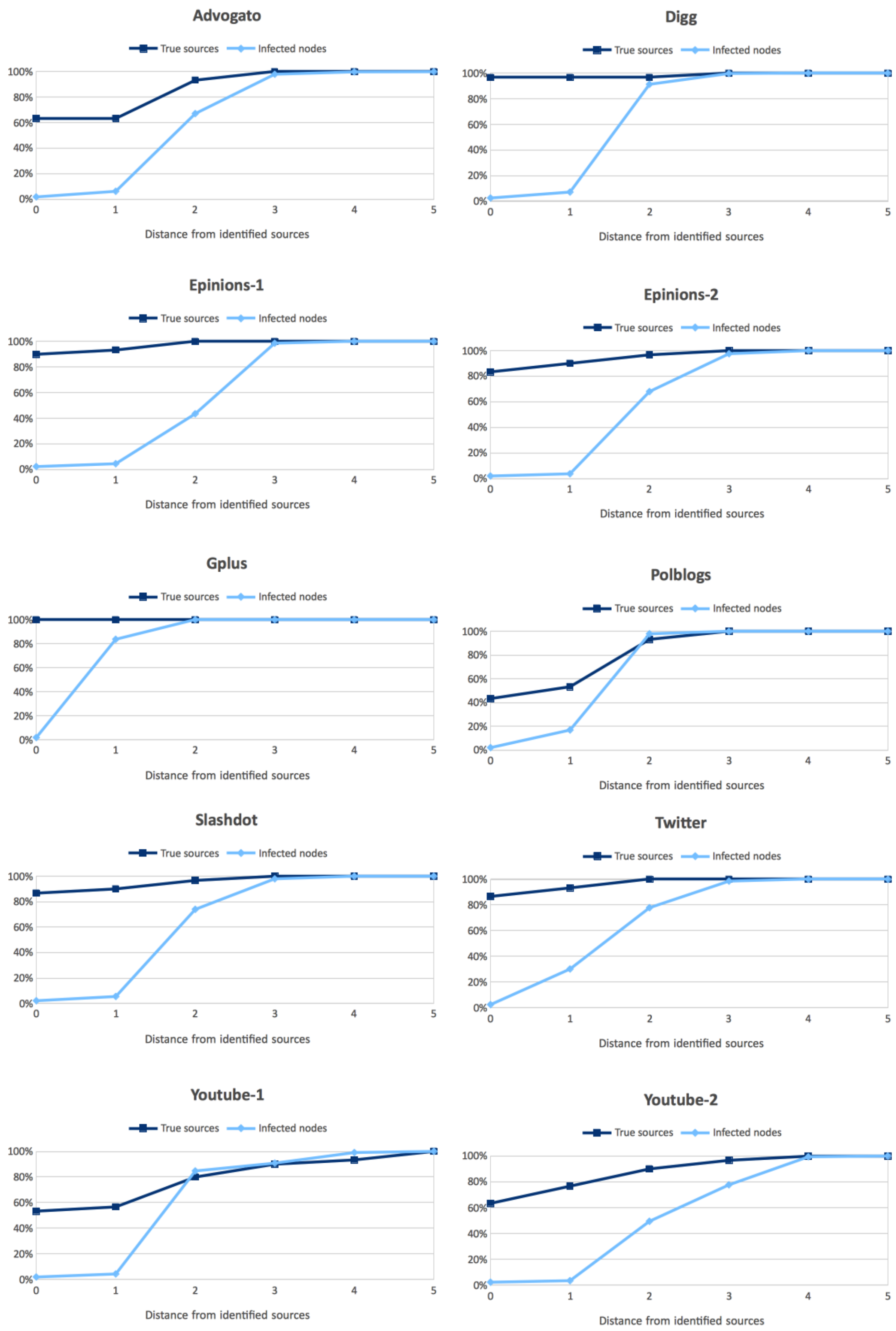


Figure 1: Source identification model performances on directed social graphs

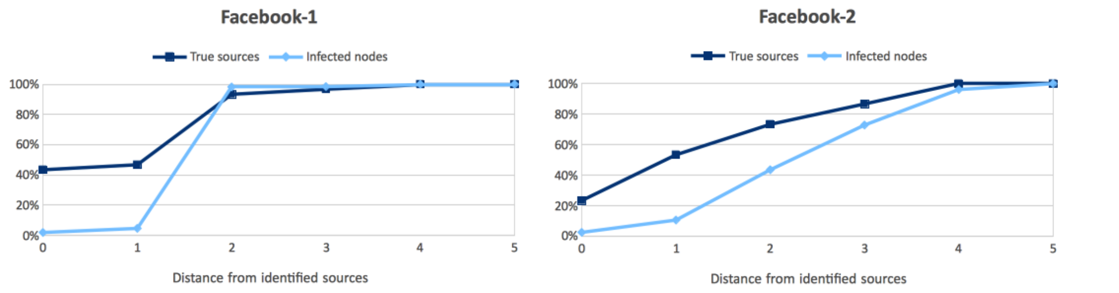


Figure 2: Source identification model performances on undirected (Facebook) social graphs

Polblogs, for which 13 true sources are at distance 0. We remark that nodes at distance 0 correspond to below 2.5% of the overall infected nodes for all instances. An analysis of the inferior performances on this instance will be presented later.

To better assess the performances of the method, we are also interested to know how far from the identified sources are the missed ones, that is, true sources that were not marked as such in a solution found by the model. This metric can be very important in practical applications. Indeed, if by keeping under observation some additional nodes the number of controlled true sources increases significantly, the trade-off can be very favorable. We define nodes at distance 1 as nodes that are not at distance 0 but are directly connected to at least one of them. Generalizing, nodes at distance i are nodes connected to those at distance $i-1$ that were not reached yet. In Figure 1 we show the percentage of *infected nodes* that are at distance i from the identified source, where the percentage is evaluated on the total number of nodes in the 9 corresponding infected networks. For all 12 instances, 100% of the infected nodes are reached within distance 5 from the identified sources.

To illustrate in detail how our results are reported in Figure 1, we describe the subfigure related to the Epinions-1 network. At distance 0, 27 of the 30 identified sources correspond to true ones. Furthermore, given that the overall number of nodes in the Epinions-1 infected networks is 1378, the 30 identified sources correspond to 2.18% of them. By also considering the nodes at distance 1, we reach 31 new nodes, and an additional true source is among them. Hence, 93.33% of the sources are found by considering 4.43% of the nodes. By adding nodes at distance 2, we find 100% of the true sources, however the number of observed nodes grows to 43.47% of the nodes. In general, we note that for Epinions-1, Epinions-2, Slashdot, Youtube-1 and Youtube-2 adding nodes at distance 1 allows to identify additional sources while keeping the number of observed nodes relatively low (around 5.5% for Slashdot and below 5% in the other cases). In the case of Polblogs, adding nodes at distance 1 the number of reached true sources grows to 53.33% (16 out of 30), considering 16.84% of the infected nodes.

We move on the results for undirected instances, shown in Figure 2. Results are generally worse than what shown for the directed case. At distance 0, we identify 43.33% and 23.33% of the true sources for Facebook-1 a Facebook-2, respectively. Looking at the Facebook-1 subfigure, we note that almost all nodes (98.18%) are reached within distance 2, similarly to the Polblogs case. However for Facebook-1, even considering nodes at distance 1 the number of true sources reached remains below 50%. Facebook-2, as said, has the lowest

Advogato		Digg		Epinions-1		Epinions-2		Facebook-1		Facebook-2	
#Src	%True src	#Src	%True src	#Src	%True src	#Src	%True src	#Src	%True src	#Src	%True src
23	63.33	33	96.67	29	90.00	31	83.33	24	43.33	20	20.00
Gplus		Polblogs		Slashdot		Twitter		Youtube-1		Youtube-2	
#Src	%True src	#Src	%True src	#Src	%True src	#Src	%True src	#Src	%True src	#Src	%True src
30	100.00	15	33.33	32	86.67	31	86.67	25	56.67	29	63.33

Table 2: Source identification model performances with unknown sources number

success rate with respect to correctly identified sources among the 12 considered instances. Adding nodes ad distance 1 (and hence considering 10.53% of the overall infected nodes) allows to reach 53.33% of the true sources. However, it is necessary to add nodes at level 3 (corresponding to 72.83% of the nodes) to reach the 80% threshold.

The results for Polblogs and the undirected Facebook graphs seem to suggest that the proposed method is less fit for these two cases. We will now discuss them separately. Looking at Table 1, we note that the Polblogs infected networks are significantly denser than the others. This may depend from the nature of the instance itself, which represents a network of hyperlinks rather than connections among people. As an additional metric, we also measured the average number of edges between nodes belonging to consecutive distance levels. This value is below 100 for Twitter and Youtube-2, between 101.31 and 174.88 for all other directed or undirected networks except Polblogs, and 452.04 for Polblogs. The significantly higher number of connections leads to a larger solutions space, that is, to a larger number of feasible spanning branchings, which in turn can correspond to a higher number of nodes that could correspond to the original sources with high probability.

Inferior performances on undirected graphs were an expected result, since the proposed maximum spanning branching approach is naturally oriented towards directed ones. More in general, given the assumption to only know the subset of infected nodes, any approach aimed at identifying the misinformation sources on undirected networks is bound to face severe difficulties. Indeed, given any two infected nodes linked by a bidirectional connection, each of them has the same probability of having been the cause of infection for the other.

We believe that performances on undirected networks can be improved by means of any type of additional information capable of (completely or partially) break the symmetry deriving from its bidirectional connections. A best-case scenario would be the one in which we are able to know, for each edge connecting two infected nodes, which of its endpoints was infected first. Such preceding orders would allow us to interpret all the edges of the infected network as directed edges. Another type of useful information could be related to asymmetric transmission probabilities among the two endpoints of a connection. For instance, given two users connected by a bidirectional friendship bond on a social network, it is reasonable to assume one of the two to have a larger influence over the other. Also in this case, taking into account this additional level of information would require to interpret the originally undirected network as a directed one.

Finally, we briefly discuss the results related to the case in which the number of sources is not known in advance. We executed the tests on the same set of 108 infected networks, using the **[SI]** formulation modified as described in Section 2. Computational times were again very low, with each test running within 30 seconds. In Table 2 we report, for each instance, how many sources were identified in total (*#Src*) and how many of them correspond in percentage to true ones (*%True src*). We note that, with the exception of Gplus, a higher or lower number of sources was always returned. In terms of true sources identification, however, the performances are quite comparable. Indeed, the same number of true sources as the previous tests were found for 9 out of 12 instances, that is Advogato, Digg, Epinions-1, Epinions-2, Facebook-1, Gplus, Slashdot, Twitter and Youtube-2. For Facebook-2 and Polblogs, the percentage of found true sources drops from 23.33% and 43.33% to 20% and 33.33%, respectively, again confirming the higher difficulty to detect sources in these cases. Finally, in the case of Youtube-1, 17 sources are found instead of 16, hence the percentage grows from 53.33% to 56.67%. To resume, moderate drops in the source identification rate occurred only for 2 highly challenging instances. The proposed method proved therefore to be applicable and competitive even in the realistic scenario in which it is not possible to know in advance the number of the sources that should be identified.

4.1.2 Experiments with Different Sources Selection

We performed some additional experiments in order to verify how different choices for the subset of sources S (as opposed to the uniformly random case) can affect the performances of our approach. In particular, we chose nodes with relatively low or relatively high centrality in the network. To this end, we considered two different centrality metrics, namely PageRank (Page et al., 1998) and out-degree. While the out-degree of a node models the number of users directly reached by the information that it produces, that is, the number of its followers, PageRank gives a measure of the influence of the node in the network beyond these direct connections.

These additional tests were run on three directed networks for which the model performed with different degrees of success when sources are chosen at random, that is, Advogato, Epinions-1 and Polblogs. Indeed, we recall that for these instances we identified correctly the 63.33%, 90% and 43.33% of the sources, respectively.

We now first describe the tests related to sources with low centrality. We premise that there is a clear motivation for focusing on this case. Indeed, as documented in the scientific literature, misinformation originates often from users with questionable credibility, unknown identities and a limited number of followers (Gupta et al., 2013; Vosoughi et al., 2018), and false news become viral quickly only once they reach more popular users. In some cases, these source users may be likened or may coincide with automated spamming accounts ("bots"), which have similar features (Eshraqi et al., 2015; Masood et al., 2019; Wang, 2010).

For each instance and each centrality metric, we considered as potential sources only the users ranked in the lowest third according to the metric itself. Again, we chose 2, 3 and 5 as values for $|S|$, and for each value of $|S|$ we simulated the infection process for 3 different sets of sources, chosen at random among the potential ones. It follows that each instance and each metric define a test scenario composed of 9 independent tests, with a total of 30 sources to be identified for each scenario, as in the previously discussed ones. The results are illustrated in Figure 3.

We can observe that a significant number of sources is always identified, and that indeed we improve the results on the same instances and random sources selection in 5 out of 6 cases. In more detail, depending on the chosen centrality metric, either the 90% or the

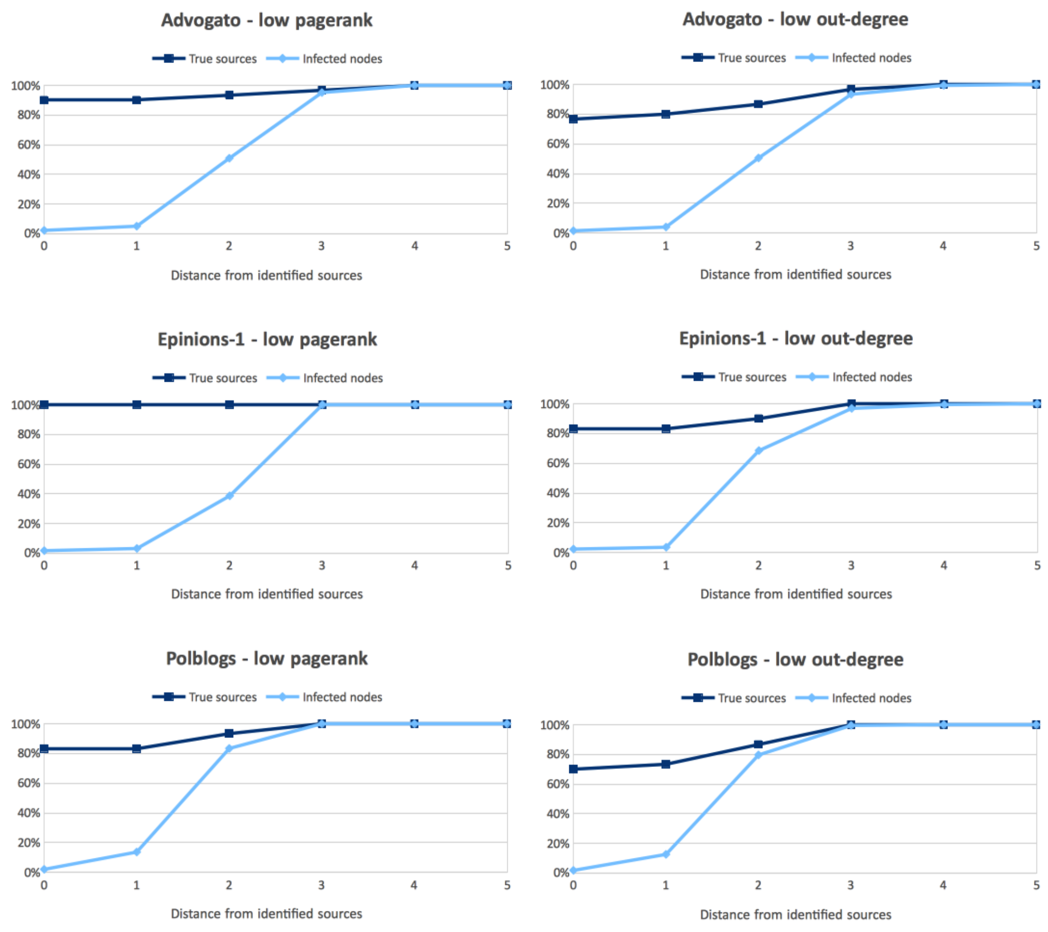


Figure 3: Source identification model performances - sources with low centrality

76.67% of the true sources are at distance 0 from the identified ones for Advogato, either the 100% or the 83.33% for Epinions-1, and either the 83.33% or the 70% for Polblogs. We find it is highly remarkable that in a more realistic scenario, as the one resulting from the adopted sources selection strategy, the performances for Polblogs improve from below 50% to at least 70%.

The general improvement in performances can be clearly explained. Indeed, commenting the tests with random sources selection, we noted that Polblogs (which was the worst-performing directed network) had the densest infected subgraphs, which led to larger solution spaces. Conversely, for any instance, choosing sparsely connected sources is likely to produce infected networks containing fewer feasible spanning branchings. Furthermore, we note that the best performances are in general obtained when sources are chosen according to PageRank, witnessing the higher effectiveness of this metric to express the overall influence of the nodes in the network.

As mentioned, in a single case out of 6 (Epinions-1 with sources chosen according to out-degree) performances are worse with respect to the random case, that is, 83.33% (25 out of 30 sources) instead of 90% (27 out of 30). Given the small performance gap, and the consistently good performances of the method on this instance, this can be easily justified by considering the randomness existing in both the sources selection phase and the Independent Cascade infection process.

We find instructive also to test our algorithms in the case that sources are selected among nodes with high centrality (specifically, random sources are selected among the potential sources, that are the nodes ranked in the highest third according to the two centrality metrics). The main aim for performing these experiments is to confirm our intuition that this selection strategy would negatively affect the performances of the proposed method, for reasons that are opposite to those of the previously discussed improvements obtained when sources have low centrality.

The results of these tests are illustrated in Figure 4. The results fully confirm our expectations. Indeed, we note that the percentage of successfully identified sources drops to 26.67% for Advogato and, depending on the considered metric, to either 50% or 43.33% for Epinions-1 and to either 10% or 20% for Polblogs.

We point out that this strategy has limited real-world adherence. Indeed, popular nodes of the network are not likely to compromise their reputation and status by willingly share misinformation, and may only occasionally and inadvertently do so. When a node with high centrality with respect to a network or a portion of it shares misinformation repeatedly, it is likely to be a known source, such as an account officially linked to a known disinformation website (Pierri et al., 2020; Shao et al., 2018), and therefore there is no need to identify it.

Finally, we remark that choosing sources according to centrality metrics did not negatively affect the computational time needed to solve the model. Indeed, none of these additional 108 tests required more than 7.5 seconds, and in 102 out of 108 cases the computational time was below 3 seconds.

4.1.3 Comparison with the Algorithm Proposed by Nguyen et al. (2012b)

In order to appreciate our results, we find useful to compare our results with the ones obtained by Nguyen et al. (2012b). To this aim, we implemented in Python the algorithm

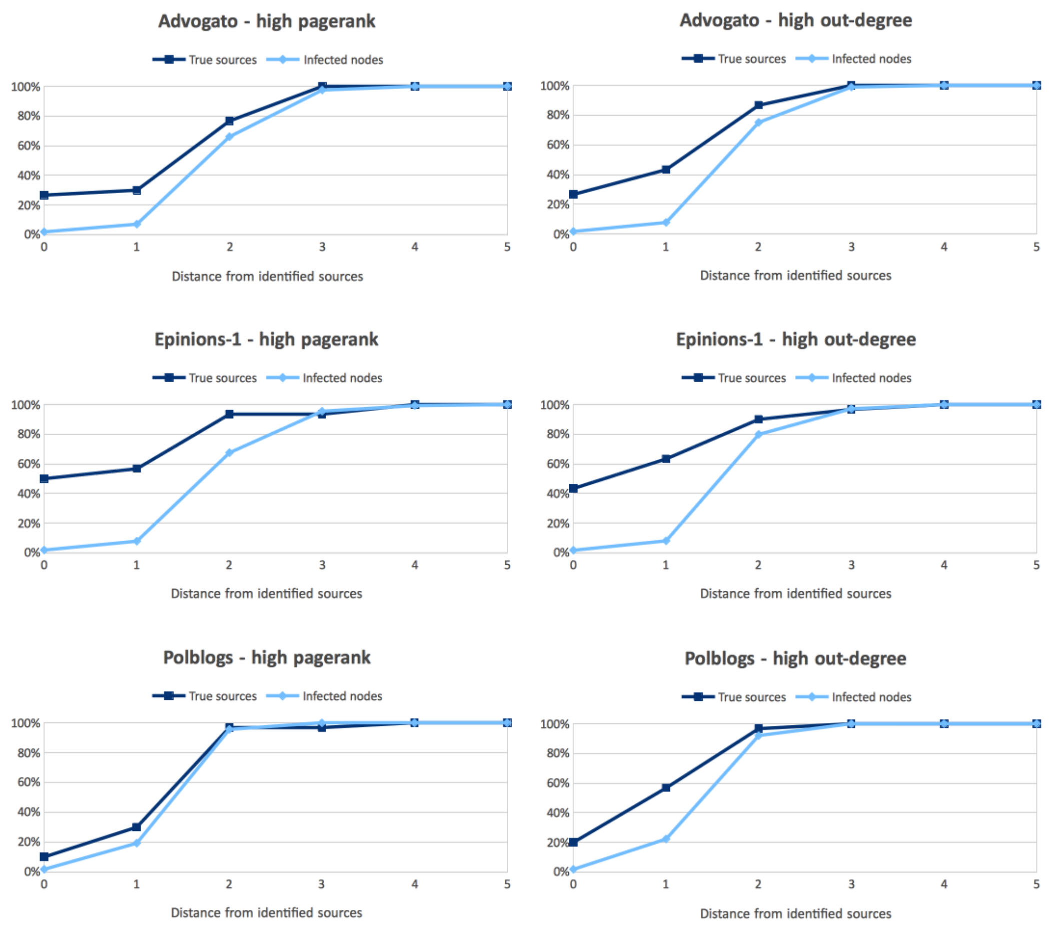


Figure 4: Source identification model performances - sources with high centrality

proposed in that work, that from now on we name NNT. In order to make our results comparable with the one given in their paper, we run our experiments only with a fixed number k of sources, with k ranging from 1 to 4 and on the Epinions-1 network.

Due to the poor performances of algorithm NNT on this network, we considered also two smaller networks, namely *Wiki-Vote* and *Gnutella08* (Leskovec & Krevl, 2014): indeed, Wiki-Vote is a dense network with of 7115 nodes and 103689 edges; Gnutella08 has comparable size, but it is much sparser, since it has 6301 nodes, but only 20777 edges.

Moreover, in order to compare the performances of the two approaches with respect to the number of infected nodes we grouped our experiments in five groups, depending on the size of the set of infected nodes V_{inf}: $[100, 250]$, $[500, 650]$, $[1000, 1200]$, $[1500, 1700]$, and $[2100, 2700]$. To force each test to be in one of these ranges, we choose a random integer i within that range, and we stop the cascade process as soon as i nodes have been infected by misinformation.

For each of these experimental settings, i.e., for every graph, each value of k, and each range, the experiment has been repeated at least 15 times.

First, we tested our heuristic for single source identification. As you can see in Figure 5, it was able to find the right sources in approx. 70% of the experiments, with a slight decrease of the success rate only when the number of infected nodes is very large. Instead, the algorithm proposed by Nguyen et al. (2012a), run on the same inputs, finds the correct source in less than 10% of experiments, and it never finds the correct source when the number of infected nodes is within the range $[2100, 2700]$.

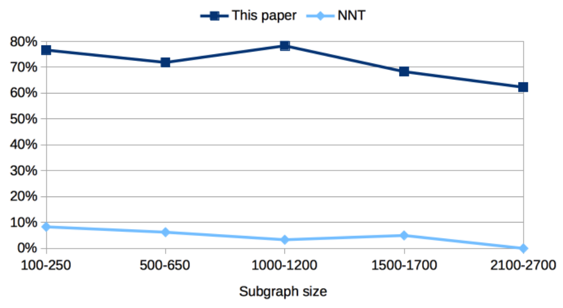
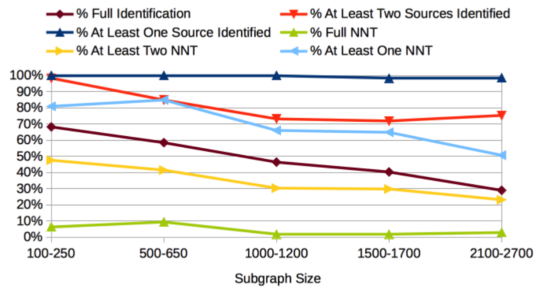
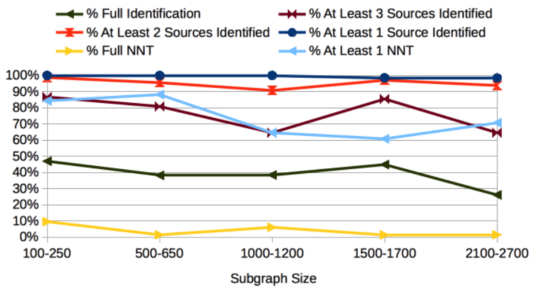


Figure 5: Comparison of success rates of the single source identification.

We also compared the performances of our algorithm and NNT with multiple sources. Clearly, in this case an algorithm can correctly identify all the sources or only part of them. Figure 6 shows the rate of (partial) successes of our approach when $k = 3$ and when $k = 4$ (results for $k = 2$ are similar and we do not present them here). As you can see, in almost all the experiments our approach correctly identified at least half of sources and in more than 70% of experiments it correctly identified all sources except at most one. Moreover, it was able to correctly identify all the sources in at least 40% of experiments, even if the success rate tends to decrease as the number of infected nodes increases. This success rate is more than five times larger than the one achieved on the same inputs by NNT, and this rate is up to twenty times larger when the number of infected nodes is large.



(a) $k = 3$



(b) $k = 4$

Figure 6: Comparison of success rates of the multiple source identification.

4.2 Monitor Placement

For the Monitor Placement problem, we performed tests on 8 instances, corresponding to 5 directed networks and 3 undirected ones. The considered directed networks are the Advogato, Digg, Epinions-1, Polblogs and Slashdot instances described in the previous subsection. As undirected networks, in addition to Facebook-1 and Facebook-2, we considered the "Hamsterster Full" instance available through the Konect database (from now on, referred to as *Hamsterster*). We further categorize these instances by their size. That is, from now on we refer to Advogato, Facebook-1, Hamsterster and Polblogs instances as *small*, and to the remaining ones as *large*. Small instances have between 1224 and 5155 nodes, while for the large ones the number of nodes ranges between 63731 and 279630.

In the tests, we consider the case in which a target node t has to be protected by a set S composed of 10, 20, 30, 40 or 50 known or suspected misinformation sources. In the k-unbalanced cut solution that we look for, we always limit the size of the subset containing the sources to 5% (rounded to integer) of the number of nodes. Indeed, a value below this threshold would lead to an infeasible scenario for the smallest instance (Polblogs), while we experimentally verified that a larger threshold rarely brings improvements. In more detail, we run the models **[MP1]** and **[MP2]** to find solutions for all instances and values of $|S|$, considering a single random choice for the elements of S and t, and (rounded) thresholds set to 5%, 6%, 7% and 8% of the number of nodes. The higher threshold values allowed to identify a smaller number of monitors only for Polblogs and $|S| \geq 30$, with improvements up to 10.55%.

In the computational tests reported in this section, for each instance and each value of $|S|$, we made 3 different random choices for the elements of S and for t. Therefore, overall we considered 120 different test cases (15 for each different instance).

4.2.1 The Results

Here we compare the two approaches described above for the Monitor Placement problem: the heuristic described in Algorithm 1 and the one given by solving the MILP formulations **[MP1]** and **[MP2]** in sequence. Our intent is twofold. On the small instances, we show both approaches to perform similarly. However, even if on small instances the MILP-based is usually effective, and sometimes even faster than the heuristic, on the large ones, it is

Instance	**|S|**	**# Monitors**			**Time**	
		mod	**heu**	**%gap**	**mod**	**heu**
Advogato	10	110.67	113.67	2.71	16.83	27.19
	20	172.33	181.67	5.42	51.06	44.12
	30	217.00	230.33	6.14	67.68	77.71
	40	329.67	336.33	2.02	60.68	100.14
	50	313.33	344.67	10.00	66.62	102.61
Facebook-1	10	5.67	5.67	0.00	2.91	0.53
	20	5.67	5.67	0.00	0.87	0.37
	30	7.33	7.67	4.55	1.28	0.38
	40	7.67	8.33	8.70	1.12	0.54
	50	210.67	194.33	-7.75	1.04	31.60
Hamsterster	10	107.67	111.67	3.72	4.97	39.49
	20	207.33	218.67	5.47	27.38	105.92
	30	319.67	340.33	6.47	28.06	133.73
	40	333.67	354.33	6.19	13.67	101.17
	50	470.33	501.00	6.52	17.89	177.87
Polblogs	10	101.33	110.33	8.88	11.91	66.50
	20	179.00	183.00	2.23	14.37	60.76
	30	188.33	187.00	-0.71	2.11	81.56
	40	341.67	323.33	-5.37	0.72	190.96
	50	367.00	337.67	-7.99	0.45	186.83

Table 3: Model-based vs. Heuristic approach for Monitor Placement on small graphs

shown that the resolution of the proposed MILPs becomes less and less viable, due to the steep increase of computational times deriving from the larger set of decision variables and the resulting larger solutions space. However, in this case, this approach still provides validation for the heuristic, which is shown to still produce solutions of good quality, within a fraction of the time needed to solve the models. This would allow to confidently use the heuristic to produce results on even larger networks, where validation through mathematical models would not be possible.

The results for the small instances are summarized in Table 3. Each row contains average results related to the scenarios corresponding to the same choice of instance and $|S|$ value. The first two columns contain the scenario characteristics. Values under the *# Monitors* heading contain the average number of monitors obtained by using each approach, while values under the *Time* one report computational times, in seconds. The *mod* and *heu* sub-headings refer to the MILP-based approach and the heuristic one, respectively; from now on, we will use these heading names to refer to the two approaches. Finally, the *%gap* subheading refers to percentage gaps between mod and heu solutions, evaluated as $100 \times \frac{heu-mod}{mod}$.

We note that the gap is never significant, indeed it varies in the $[-7.99, 10]$ interval. A negative gap means that heu found on average better solutions in the related scenario. While this may seem unintuitive, it must be remembered that the minimum k-unbalanced cut is used as a proxy of the actual monitor minimization problem, as well as to favor the presence of edges with low transmission probability in the subset containing the misinformation sources. Therefore, the k-unbalanced cut with minimum weight (which is always identified by [**MP1**]) could potentially not lead to the minimum number of monitors. Overall, we

Instance	|S|	# Monitors			Time	
		mod	heu	%gap	mod	heu
Digg	10	7.67	7.67	0.00	1922.67	180.64
	20	72.67	75.67	4.13	42277.23	301.81
	30	137.00	180.00	31.39	38684.60	409.50
	40	115.67	130.67	12.97	36573.10	267.80
	50	256.67	258.00	0.52	54586.37	341.81
Epinions-1	10	73.00	75.67	3.65	2303.80	68.57
	20	128.67	150.33	16.84	3791.17	73.36
	30	108.33	146.00	34.77	3079.57	95.12
	40	159.33	159.33	0.00	2631.34	121.67
	50	325.33	331.00	1.74	4036.33	226.17
Facebook-2	10	185.67	199.67	7.54	11456.06	465.08
	20	496.33	529.33	6.65	23493.13	1015.44
	30	791.00	833.33	5.35	43993.73	1162.04
	40	1066.67	1181.00	10.72	39974.83	2226.79
	50	1573.33	1708.33	8.58	39872.60	3752.70
Slashdot	10	18.00	22.67	25.93	2518.09	53.11
	20	165.33	175.33	6.05	4679.56	94.55
	30	134.67	145.33	7.92	4208.21	94.22
	40	92.33	99.00	7.22	3449.51	54.63
	50	207.00	216.33	4.51	3182.19	142.63

Table 4: Model-based vs. Heuristic approach for Monitor Placement on large graphs

note that mod finds better solutions in 16 out of 20 scenarios, with heu performing better in the remaining 4.

We note that, in general, the number of monitors to be placed grows as $|S|$ grows. The 10% gap occurs for Advogato with $|S|$=50, while the -7.99% gap occurs for Polblogs, again in the case $|S|$=50.

With respect to computational times, both algorithms result to be efficient. However for heu, we notice a general increase for larger $|S|$ values. No such trend can be noted for mod. Overall, heu requires up to 190.96 seconds, while mod runs within 66.62 seconds in the worst case.

Table 4 contain results for large instances. All table headings have the same meanings discussed for Table 3. We note that, for these scenarios, gaps are always positive. However, heu remains remarkably competitive. Indeed, out of 20 scenarios, the gap is null twice, within 5% in 7 cases, within 10% in 14 cases, and over 20% (up to 34.77%) only three times. An interesting result is that the gap is always below 10% in the cases with $|S|$=50, which generally require a higher number of monitors. For lower values of $|S|$, the higher gap values are in part justified by the fact that solution values, in absolute terms, are smaller; see for instance the case of the Slashdot instance with $|S|$=10, where the 25.93% gap corresponds to a difference of 4.67 monitors.

In terms of computational times, heu is (as anticipated) more efficient by a significant margin. Indeed, it is in all cases faster by at least one order of magnitude. Overall, the highest computational time for heu corresponds to the Facebook-2 instance with $|S|$=50, and it is equal to 3752.70 seconds. The related computational time for mod is 39872.60, while its highest one, corresponding to Digg with $|S|$=50, is equal to 54586.37 seconds. Furthermore, heu runs within 100 seconds in 7 out of 20 scenarios, and within 1000 seconds in 16 cases. On the other hand, the fastest computational time for mod is 1922.67 seconds, corresponding to Digg with $|S|$=10.

4.2.2 Comparison with the Algorithm Proposed by Zhang et al. (2015a)

As above, we find instructive to compare our heuristic with previous works. Specifically, we will show the results for the comparison with the algorithm `MMSC` proposed by Zhang et al. (2015a). To this reason we implemented this algorithm in Python. Due to the large running time of the `MMSC` algorithm, we compared the performances of the two approaches only on three small networks: the Wiki-Vote and the Gnutella08 network discussed above, and on a subset of the Epinions-1 network, named *SUB-Epinions*, consisting of 7479 nodes and 25855 edges. The network SUB-Epinions has been created by randomly choosing an integer n between 7000 and 7500, selecting n nodes at random from the largest strongly connected component of Epinions, and considering the graph induced by these nodes. Edge weights have been randomly assigned.

We also followed the choice of misinformation sources made by Zhang et al. (2015a). We considered a set S of sources, with $|S| = 10, 20, \dots, 50$, and only one target node t. Sources are selected randomly among the set of nodes with low out-degree that are neighbors of the $|S|$ nodes with the largest degrees. Target is selected uniformly at random among nodes with low in-degree. Here, we say that the degree of node is low (high) if it is below (above, resp.) the average degree of the network.

For each graph G and each set of sources S, we first contracted sources into a single source (see Section 3 for details) and then we run algorithm `MMSC` with parameters $\tau = 0.1$ and $\delta \in \{1, 2\}$ (recall that if we increase δ, then we are allowing more nodes to be infected by misinformation).

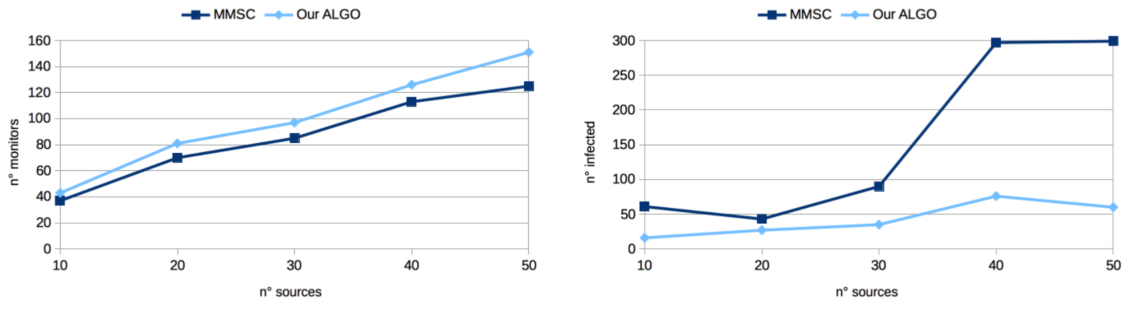
In order to make the results of the algorithms comparable, we would like to have more or less the same expected number of nodes that are reached by misinformation. For this reason, we run 100 separate executions of the Independent Cascade diffusion process on the network G with sources from S and monitors placed according to algorithm `MMSC`, and let k be the average number of nodes infected by misinformation in these executions. Then we run our heuristic on input (G, S, t, k)

For each graph, each value of $|S|$ and each value of δ we executed the experiment 10 times and evaluated both the average number of monitors and the average number of vertices reached by misinformation.

The results of our experiments show very different behaviors for the cases of $\delta = 1$ and $\delta = 2$. When $\delta = 1$ our heuristic places a number of monitors that is slightly greater than algorithm `MMSC`.

We remark that this slightly increase in the number of monitors, never greater than 20%, is counterbalanced by the much more stronger results of our heuristic in terms of limitations to the spread of misinformation.

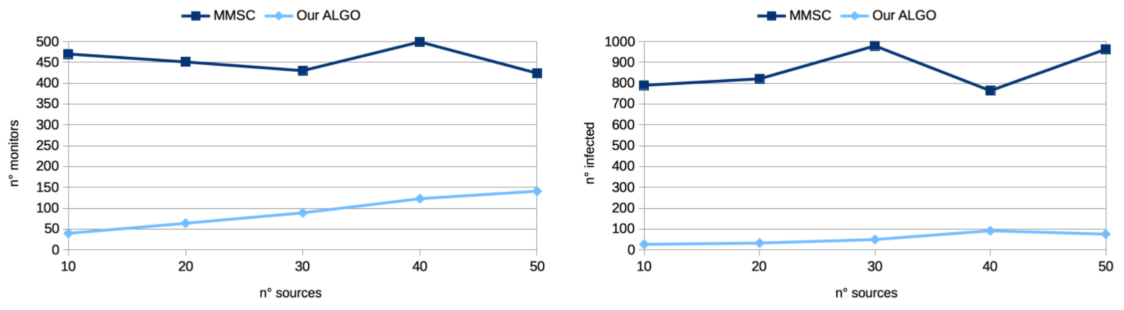
Moreover, with our heuristic the average number of nodes that are reached by misinformation even in presence of monitors is much less than `MMSC` and the difference between the two algorithms explodes as the number of sources increases. In Figure 7a and 7b, we show results only for the Wiki-Vote network, since results for the other networks are similar.



(a) Monitors (b) Infected

Figure 7: Performances of the two algorithms for Monitor Placement on Wiki-Vote, when $\delta = 1$.

When $\delta > 1$ our heuristic outperforms the `MMSC` algorithm with respect to both the number of monitors placed and the number of nodes exposed to the misinformation. As you can see in Figure 8a, the number of monitors placed by our heuristic remains almost unchanged regardless of the value of δ, whereas the number of monitors placed by `MMSC` explodes. Moreover, as shown in Figure 8b, even if `MMSC` places much more monitors, our heuristic has much better performances with respect to the number of infected nodes.



(a) Monitors (b) Infected

Figure 8: Performances of the two algorithms for Monitor Placement on Wiki-Vote, when $\delta = 2$.

5. Conclusions and Future Work

In this paper we considered the problem of contrasting the spread of misinformation in an online social network. We proposed two heuristics for first identifying the sources of misinformation and then placing a set of monitors on nodes of the network to limit the spread of misinformation.

Our heuristics are based on well-studied graph-theoretic problems, namely computing the maximum spanning branching of a directed graph, or an unbalanced cut. Both our heuristics can have arbitrarily large approximation guarantees, due the previously known hardness results. However, they performed very well in the extensive tests we run and largely outperformed previously known algorithms.

We also observed that there are implementations of our approaches that return the solution very quickly even for large networks. However, we believe that further research on how improve both the performances and the running time of source identification and monitor placement algorithms would be of huge practical and theoretical relevance.

In our setting infected nodes are surely recognized. It would be very interesting to consider the case that infected nodes are recognized only with some level of confidence. Our feeling is that our techniques still work by simply down-weighting the edges that leave a node with a factor that corresponds to the probability that node is infected. However, we do not have run experiments on this extension.

Finally, we assume that misinformation spreads according to a cascade model. However, it would be interesting to evaluate the extent at which our approach works with other well-known models, such as epidemics and threshold models, and their noisy variants (Auletta et al., 2013a, 2013b).

Yet another interesting direction would be to design misinformation containment strategies that are robust even in settings in which the location of seeds of misinformation may change over time and adapt itself to minimize the effectiveness of placed monitors. A first step in this direction has been recently taken by Auletta et al. (2020).

Acknowledgments

A preliminary version of this paper appeared in the proceedings of the AAMAS 2017 conference (Amoruso et al., 2017). Vincenzo Auletta and Diodato Ferraioli are partially supported by "GNCS-INdAM" and by the Italian MIUR PRIN 2017 Project ALGADIMAR "Algorithms, Games, and Digital Markets".

References

Amoruso, M., Anello, D., Auletta, V., & Ferraioli, D. (2017). Contrasting the spread of misinformation in online social networks. In *AAMAS '17*, pp. 1323–1331.

Aspnes, J., Chang, K., & Yampolskiy, A. (2006). Inoculation strategies for victims of viruses and the sum-of-squares partition problem. *Journal of Computer and System Sciences*, *72*(6), 1077 – 1093.

Auletta, V., De Nittis, G., Ferraioli, D., Gatti, N., & Longo, D. (2020). Strategic monitor placement against malicious flows. In *ECAI '20*.

Auletta, V., Ferraioli, D., Pasquale, F., Penna, P., & Persiano, G. (2013a). Logit dynamics with concurrent updates for local interaction games. In *ESA '13*, pp. 73–84.

Auletta, V., Ferraioli, D., Pasquale, F., & Persiano, G. (2013b). Mixing time and stationary expected social welfare of logit dynamics. *Theory of Computing Systems*, *53*(1), 3–40.

Budak, C., Agrawal, D., & El Abbadi, A. (2011). Limiting the spread of misinformation in social networks. In *WWW '11*, pp. 665–674.

Camerini, P. M., Fratta, L., & Maffioli, F. (1980). The k best spanning arborescences of a network. *Networks*, *10*(2), 91–109.

Chu, Y.-J., & Liu, T.-H. (1965). On the shortest arborescence of a directed graph. *Science Sinica*, *14*.

Comin, C. H., & da Fontoura Costa, L. (2011). Identifying the starting point of a spreading process in complex networks. *Physical Review E*, *84*(5), 056105.

Edmonds, J. (1967). Optimum Branchings. *Journal of Research of the National Bureau of Standards*, *71B*, 233–240.

Einstein, K. L., & Glick, D. M. (2015). Do I think BLS data are BS? The consequences of conspiracy theories. *Political Behavior*, *37*(3), 679–701.

Eshraqi, N., Jalali, M., & Moattar, M. H. (2015). Detecting spam tweets in twitter using a data stream clustering algorithm. In *2015 International Congress on Technology, Communication and Knowledge (ICTCK)*, pp. 347–351.

Fan, L., Lu, Z., Wu, W., Thuraisingham, B., Ma, H., & Bi, Y. (2013). Least cost rumor blocking in social networks. In *ICDCS '13*, pp. 540–549.

Ferrara, E., Varol, O., Davis, C., Menczer, F., & Flammini, A. (2016). The rise of social bots. *Commun. ACM*, *59*(7), 96–104.

Frier, S. (2018). The end of Facebook's easy growth. `https://www.bloomberg.com/news/articles/2018-07-26/the-end-of-facebook-s-easy-growth`.

Gallo, G., Grigoriadis, M. D., & Tarjan, R. E. (1989). A fast parametric maximum flow algorithm and applications. *SIAM Journal on Computing*, *18*(1), 30–55.

Goldenberg, J., Libai, B., & Muller, E. (2001a). Talk of the network: A complex systems look at the underlying process of word-of-mouth. *Marketing Letters*, *12*(3), 211–223.

Goldenberg, J., Libai, B., & Muller, E. (2001b). Using complex systems analysis to advance marketing theory development: Modeling heterogeneity effects on new product growth through stochastic cellular automata. *Academy of Marketing Science Review*, *2001*, 1.

Gupta, A., Lamba, H., & Kumaraguru, P. (2013). $1.00 per RT #BostonMarathon #PrayForBoston: Analyzing fake content on twitter. In *2013 APWG eCrime Researchers Summit*, pp. 1–12, Article number 6805772.

Habiba, Yu, Y., Berger-Wolf, T. Y., & Saia, J. (2010). Finding spread blockers in dynamic networks. In *ASONAM '10*, pp. 55–76.

Hamidian, S., & Diab, M. (2016). Rumor identification and belief investigation on twitter. In *WASSA '16*, pp. 3–8.

Hayrapetyan, A., Kempe, D., Pál, M., & Svitkina, Z. (2005). *Unbalanced Graph Cuts*, pp. 191–202.

Hosni, A. I. E., Li, K., Ding, C., & Ahmed, S. (2018). Least cost rumor influence minimization in multiplex social networks. In *NIPS '18*, pp. 93–105.

Hotez, P. J. (2016). Texas and its measles epidemics. *PLoS Medicine*, *13*(10), 1–5.

Jiang, J., Wen, S., Yu, S., Xiang, Y., Zhou, W., & Hossain, E. (2014). Identifying propagation sources in networks: State-of-the-art and comparative studies. *IEEE Communications Surveys and Tutorials, accepted*, *17*(9).

Jin, F., Dougherty, E., Saraf, P., Cao, Y., & Ramakrishnan, N. (2013). Epidemiological modeling of news and rumors on twitter. In *SNA-KDD '13*, p. 8.

Kempe, D., Kleinberg, J., & Tardos, É. (2003). Maximizing the spread of influence through a social network. In *KDD '03*, pp. 137–146.

Kimura, M., Saito, K., & Motoda, H. (2008). Minimizing the spread of contamination by blocking links in a network. In *AAAI'08*, pp. 1175–1180.

Ko, H., Hong, J. Y., Kim, S., Mesicek, L., & Na, I. S. (2019). Human-machine interaction: A case study on fake news detection using a backtracking based on a cognitive system. *Cognitive Systems Research*, *55*, 77–81.

Kuhlman, C. J., Anil Kumar, V. S., Marathe, M. V., Ravi, S. S., & Rosenkrantz, D. J. (2010). Finding critical nodes for inhibiting diffusion of complex contagions in social networks. In *ECML PKDD '10*, pp. 111–127.

Lappas, T., Terzi, E., Gunopulos, D., & Mannila, H. (2010). Finding effectors in social networks. In *KDD '10*, pp. 1059–1068.

Leskovec, J., & Krevl, A. (2014). SNAP Datasets: Stanford large network dataset collection. `http://snap.stanford.edu/data`.

Lewandowsky, S., Ecker, U. K., Seifert, C. M., Schwarz, N., & Cook, J. (2012). Misinformation and its correction: continued influence and successful debiasing. *Psychological Science in the Public Interest*, *13*(3), 106–131.

Li, S., Zhu, Y., Li, D., Kim, D., & Huang, H. (2013). Rumor restriction in online social networks. In *IPCCC '13*, pp. 1–10.

Luckerson, V. (2014). Fear, misinformation, and social media complicate ebola fight. `https://time.com/3479254/ebola-social-media/`.

Masood, F., Ammad, G., Almogren, A., Abbas, A., Khattak, H. A., Ud Din, I., Guizani, M., & Zuair, M. (2019). Spammer detection and fake user identification on social networks. *IEEE Access*, *7*, 68140–68152.

Mian, A., & Khan, S. (2020). Coronavirus: the spread of misinformation. *BMC Medicine*, *18*, 89.

Nguyen, D. T., Nguyen, N. P., & Thai, M. T. (2012a). Sources of misinformation in online social networks: Who to suspect?. In *MILCOM '12*, pp. 1–6.

Nguyen, N. P., Yan, G., Thai, M. T., & Eidenbenz, S. (2012b). Containment of misinformation spread in online social networks. In *WebSci '12*, pp. 213–222.

Page, L., Brin, S., Motwani, R., & Winograd., T. (1998). The PageRank citation ranking: Bringing order to the web. In *Proceedings of the 7th International World Wide Web Conference*, pp. 161–172.

Pierri, F., Artoni, A., & Ceri, S. (2020). Investigating italian disinformation spreading on Twitter in the context of 2019 European elections. *PLoS ONE*, *15*(1), e0227821.

Pogue, D. (2017). How to stamp out fake news. *Scientific American*, *316*(2), 24–24.

Richardson, M., Agrawal, R., & Domingos, P. (2003). Trust management for the semantic web. In *ISWC '03*, pp. 351–368.

Shah, D., & Zaman, T. (2011). Rumors in a network: who's the culprit?. *IEEE Transactions on Information Theory*, *57*(8), 5163–5181.

Shao, C., Hui, P.-M., Wang, L., Jiang, X., Flammini, A., Menczer, F., & Ciampaglia, G. (2018). Anatomy of an online misinformation network. *PLoS ONE*, *13*(4), e0196087.

Shin, J., Jian, L., Driscoll, K., & Bar, F. (2018). The diffusion of misinformation on social media: Temporal pattern, message, and source. *Computers in Human Behavior*, *83*, 278 – 287.

Shu, K., Sliva, A., Wang, S., Tang, J., & Liu, H. (2017). Fake news detection on social media: A data mining perspective. *ACM SIGKDD Explorations Newsletter*, *19*(1), 22–36.

Tong, G., Wu, W., Guo, L., Li, D., Liu, C., Liu, B., & Du, D. (2020). An efficient randomized algorithm for rumor blocking in online social networks. *IEEE Transactions on Network Science and Engineering*, *7*(2), 845–854.

Vijaykumar, S., Nowak, G., Himelboim, I., & Jin, Y. (2018). Virtual Zika transmission after the first U.S. case: who said what and how it spread on twitter. *American Journal of Infection Control*, *46*(5), 549 – 557.

Vosoughi, S., Roy, D., & Aral, S. (2018). The spread of true and false news online. *Science*, *359*(6380), 1146–1151.

Wang, A. H. (2010). Don't follow me: Spam detection in twitter. In *2010 International Conference on Security and Cryptography (SECRYPT)*, pp. 142–151.

Weeks, B. E., & Garrett, R. K. (2014). Electoral consequences of political rumors: Motivated reasoning, candidate rumors, and vote choice during the 2008 u.s. presidential election. *International Journal of Public Opinion Research*, *26*(4), 401–422.

Wilder, B., & Vorobeychik, Y. (2018). Controlling elections through social influence. In *AAMAS '18*, pp. 265–273.

Wu, L., Li, J., Hu, X., & Liu, H. (2017). Gleaning wisdom from the past: Early detection of emerging rumors in social media. In *SDM '17*, pp. 99–107.

Yang, S., Shu, K., Wang, S., Gu, R., Wu, F., & Liu, H. (2019). Unsupervised fake news detection on social media: A generative approach. In *AAAI '19*.

Zhang, H., Alim, M. A., Li, X., Thai, M. T., & Nguyen, H. T. (2016). Misinformation in online social networks: Detect them all with a limited budget. *ACM Transactions on Information Systems*, *34*(3), 18:1–18:24.

Zhang, H., Alim, M. A., Thai, M. T., & Nguyen, H. T. (2015a). Monitor placement to timely detect misinformation in online social networks. In *ICC '15*, pp. 1152–1157.

Zhang, H., Zhang, H., Li, X., & Thai, M. T. (2015b). Limiting the spread of misinformation while effectively raising awareness in social networks. In *CSoNet '15*, pp. 35–47.

Zubiaga, A., Aker, A., Bontcheva, K., Liakata, M., & Procter, R. (2018). Detection and resolution of rumours in social media: A survey. *ACM Computing Surveys*, *51*(2), 32:1–32:36.