Social Network Analysis and Mining (2023) 13:122
https://doi.org/10.1007/s13278-023-01122-8

ORIGINAL ARTICLE

Community deception in directed influence networks

Saif Aldeen Madi[1] · **Giuseppe Pirrò**[2]

Received: 10 March 2023 / Revised: 29 June 2023 / Accepted: 25 August 2023 / Published online: 25 September 2023
© The Author(s) 2023

Abstract
Community deception is about protecting users of a community from being discovered by community detection algorithms. This paper studies community deception in directed influence network (DIN). It aims to address the limitations of the state of the art through a twofold strategy: introducing directed influence and considering the role of nodes in the deception strategy. The study focuses on using modularity as the optimization function. It offers several contributions, including an upgraded version of modularity that accommodates the concept of influence, edge-based, and node-based deception algorithms.. The study concludes with a comparison of the proposed methods with the state of the art showing that not only influence is a valuable ingredient to devising deception strategies but also that novel deception approaches centered on node operations can be successfully devised.

Keywords Community Detection · Social Networks · Privacy

1 Introduction

Social network analysis has been an active area of research thanks to the accelerating growth of social media platforms with billions of users worldwide. A particular example is community detection (Fortunato 2010), which is a relatively well-established research problem. Community detection in networks is a crucial task with wide-ranging applications in diverse fields such as social media analysis, recommendation systems (Rezaeimehr et al. 2018), fraud detection (Sarma et al. 2020), biology, and transportation planning (Fortunato 2010). Identifying communities or clusters within a network provides insights into its underlying structure of complex systems and helps the understanding of interaction patterns among its nodes. This information can subsequently be used to develop targeted interventions to prevent the spread of diseases, optimize transportation networks, identify influential individuals, and enhance the efficiency of communication networks.

1.1 Motivations

However, detection algorithms may reveal sensitive information about an individual's social connections, affiliations, or behaviors that can be used to infer personal information about them. This raises subtle ethical dilemmas regarding user privacy (Fionda and Pirrò 2018; Waniek et al. 2018), freedom of speech, and security. For example, suppose a detection algorithm identifies a group of nodes that frequently interact with each other, such as membership in a specific political group or religion. In that case, it may reveal sensitive information about an individual's beliefs or preferences. On the other hand, identifying a cluster of nodes that frequently interact with each other but also have a few connections to nodes outside the cluster may inadvertently disclose information about the activities of those outside nodes. Such issues are amplified by the fact that social networking platforms, such as Facebook or Twitter, constitute an essential application area for community detection, and it is expected that such algorithms will play an increasingly influential role in the lives of millions of users. Such concerns ignited serious efforts toward designing algorithms that enable communities of users to protect their privacy through evading community detection algorithms. This new research

✉ Giuseppe Pirrò
giuseppe.pirro@unical.it

Saif Aldeen Madi
madi@di.uniroma1.it

[1] Department of Computer Science, Sapienza University of Rome, Piazzale Aldo Moro 5, 00185 Rome, Italy

[2] Department of Mathematics and Computer Science, University of Calabria, via Pietro Bucci 30B, 87036 Rende, CS, Italy

field has been variously referred to as community deception (Fionda and Pirrò 2018), or community hiding (Waniek et al. 2018), which aims at concealing a target community $\mathcal{C}$ from community detection by rewiring connections incident to its nodes. Stated more formally, if we are given a community structure $\overline{C} = \{C_1, C_2, ...C_k\}$ discovered by a community detection algorithm $\mathcal{A}_{det}$ (which need not be revealed) run over a network G, the goal is to find and perform a deception-wise set of edge updates such that when $\mathcal{A}_{det}$ is rerun on the updated network G', it returns a new community structure $\overline{C} = \{C_1, C_2, ...C_k\}$ with a better hidden $\mathcal{C}$. Some quality scores (for example, deception score (Fionda and Pirrò 2018)) have been defined to measure the performance of deception strategies. However, the state of the art has considered the problem from the perspective of structural information centered on edge updates, disregarding two crucial aspects in (social) networks: the variation of influence that nodes exert on each other and strategies that incorporate node operations for deception.

1.2 The importance of influence

Influence in social networks refers to individuals' ability to affect their peers' attitudes, beliefs, or behaviors (Gubanov et al. 2009). It is a crucial concept in understanding social dynamics. One of the most widely used models of influence is the cascade model (Wang et al. 2012), which assumes that individuals' behavior is influenced by their immediate neighbors, and that influence propagates through the network. This model has been used to study the spread of social network innovations, opinions, and behaviors. Another influence model is the threshold model (Chen et al. 2010), which assumes that individuals have a certain threshold for adopting a new behavior or belief and are more likely to adopt it if their peers have already done so. The model has been used to study the adoption of new technologies or products in social networks. In terms of definitions, influence can be measured in various ways, such as the number of followers, the frequency of interactions, or the degree of similarity between individuals (Cheng et al. 2010). Some researchers also distinguish between different types of influence, such as informational influence (where individuals are influenced by the information provided by their peers) and normative influence (where social norms or expectations influence individuals). Hence, since influence has an essential role in establishing connections and, as a by-product, communities, we contend that considering or deriving influence (Kumar et al. 2016) in social networks is vital for designing effective strategies to promote behavior change by members of a community to counteract community detection algorithms.

On the other hand, considering deception strategies that are (also) node-based can accommodate the natural flow of the community that wants to hide—$\mathcal{C}$ members can (strategically) join or leave the network. It could be argued that node-centered deception is encompassed by edge-centered deception; however, this is inaccurate for two primary reasons. Firstly, edge-centered deception does not account for the possibility of nodes joining or leaving the targeted community $\mathcal{C}$, which is a natural occurrence in a network; it is unclear how to insert edges for a new node in a profitable manner. Secondly, edge-centered methods do not examine deception from the perspective of moving a particular node from one community to another. These questions are only resolved when edge operations are treated in the context of nodes, that is, by incorporating the node role in the deception optimization function.

This paper aims to fill these gaps in community deception literature by presenting a new theoretical framework that accommodates the notion of influence and considers deception at the edge and node levels.

1.3 Contributions and outline

The aim of this work is to address all previous issues with a twofold strategy: (i) incorporating *directed influence* as a vital element in the deception process and (ii) examining the crucial *role of nodes* in devising effective deception strategies. We extensively investigate community deception in directed influence network (DIN) from an edge- and node-centric perspective. Given its wide acceptance in the community detection field as a cluster quality function (Fortunato 2010), we focus on optimizing deception using modularity as our preferred function. To achieve these objectives, we offer the following contributions:

1. *Formalization*: We formally introduce the concept of community deception within the context of DIN, shedding new light on this area. We present an upgraded version of modularity that seamlessly incorporates the notion of influence, enhancing the accuracy of our deception analysis.
2. *Edge-deception*: Our study delves into edge-based deception in DIN, including a comprehensive theoretical examination of the most advantageous edge updates from a deception standpoint. We introduce an innovative edge-centered deception algorithm called IDEC offering a sophisticated solution for effective deception by leveraging edge-based strategies' complexity and unique features.
3. *Node-deception*: We explore deception from the perspective of node updates, encompassing node additions, deletions, and movements between communities, further enhancing the versatility of our approach. We introduce a second deception algorithm, INDEC, combining edge-centered and node-centered deception techniques,

presenting a holistic solution to address various scenarios.

4. *Evaluation*: We evaluate our approaches on diverse datasets and perform a comprehensive comparison with state-of-the-art methods, demonstrating the superiority and effectiveness of our proposed strategies.

It is worth noting that this paper is an extension of our previous work published in CNA 2022 (Madi and Pirrò 2023). However, this current paper significantly differs in several key aspects. We have expanded the introduction to emphasize the criticality of considering influence and node operations in devising successful deception strategies. Additionally, we introduce the novel problem of node-centric community deception in Section. 5. Lastly, we have conducted a more comprehensive experimental evaluation, including a broader range of competitors, to provide a thorough and robust analysis of our proposed methods.

The remainder of the paper is organized as follows. Section 3 introduces the community deception problem. Section 4 introduces influence-based community deception problem in directed networks and a greedy algorithm called IDEC. Section 5 formalizes influence-based community deception from the node perspective along with a greedy algorithm called INDEC to solve the optimization problem. Section 7 reports on an experimental evaluation. We conclude in Section. 8.

2 Related work

Community detection plays a crucial role in network analysis as it unveils its members' unique characteristics and relationships, setting them apart from other communities within the network. Traditional approaches (see (Fortunato and Newman 2022) for a recent survey) have been widely used. More recently, there has been notable progress in community detection by adopting deep learning techniques (Su et al. 2022). These advanced methods offer distinct advantages when dealing with complex networks characterized by high-dimensional data. The usage of community detection algorithms can pose privacy risks due to the nature of the data being analyzed and the potential for information leakage. Among these, we mention: (i) *membership disclosure* due to the fact that these algorithms can reveal the affiliations and connections of individuals within a network, potentially exposing sensitive information about their social, professional, or personal relationships; (ii) *re-identification attacks*: If an adversary has access to auxiliary information or external datasets, they can combine it with the community detection results to re-identify individuals who were thought to be anonymized; (iii) community detection can create profiles of groups or communities, which might lead to stereotyping, bias, or unfair targeting of individuals based on their group membership. This can have negative consequences in various contexts, including employment, insurance, or law enforcement.

To mitigate these potential violations, community deception (Fionda and Pirrò 2018; Magelinski et al. 2021) or community hiding (Waniek et al. 2018; Mittal et al. 2021) is a relatively new research area. It is noteworthy that some authors refer to deception as an *attack* (e.g., Chen et al. (2019)), which reflects the perspective of the detector.

Table 1 compares IDEC and INDEC with the state of the art. Nagaraja (2010) studied how to hide a community by adding additions toward nodes with high centrality. DICE(Waniek et al. 2018) and MOD(Fionda and Pirrò 2018) are based on the heuristic of deleting intra-community edges and adding inter-community edges with the assumption that such edge updates minimize modularity. SAF(Fionda and Pirrò 2018) is a deception approach based on node safeness, which has been extended by Chen et al. (2021). Mittal et al. (2021) introduced the NEURALdeception approach based on permanence minimization. DSAF, DMOD, and DPERhave been proposed as counterparts of SAF, MOD, and NEURALin directed networks, respectively (Fionda et al. 2022a).

Q- ATTACK (Chen et al. 2019), REM (Liu et al. 2019), CGN (Liu et al. 2022), and PROHICO (Liu et al. 2021) are approaches that hide the entire community structure. However, we believe it is not easy to perform such a task by modifying the entire network; indeed, one cannot access the whole network (e.g., Facebook). Moreover, these approaches add and delete edges arbitrarily, which may not be possible; for example, one cannot add an edge to Facebook, establishing a friendship without the consent of the two nodes involved. Moreover, Q- ATTACK, due to the combinatorial nature of genetic algorithms, does not scale (it was only tested on nodes with approximately 100 nodes). The choice of NMI as an optimization criterion in CGN(Liu et al. 2022) approach depends on the output of a specific detection algorithm. EPA(Chen et al. 2020) is a genetic approach to hide communities and individuals. However, EPA did not scale due to the combinatorial nature and was tested on relatively small networks. Moreover, none of these approaches work on directed graphs, which is more challenging, similar to community detection in directed networks (Fortunato 2010). Besides, none of the existing approaches consider deception from the nodes' perspective. Last but not least, previous work on deception algorithms overlooked an essential component of social relationships, namely *influence* (Ghosh and Lerman 2008; Lu et al. 2014). Indeed, to the best of our knowledge, all previous deception algorithms have been devised using influence-less 0-1 edges. Such representation assumes that a pair of nodes is either connected or not, ignoring the strength of the influence a node exerts on its neighbors. We consider this a severe drawback of state-of-the-art community

Table 1 Comparison of IDEC and INDEC with related work

Strategy	NAGARAJA	DICE	SAF	MOD	NEURAL	DSAF	DMOD	DPER	Q- ATTACK	REM	EPA	CGN	PROHICO	IDEC	INDEC
Edge additions	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Edge deletions		✓	✓	✓	✓	✓	✓	✓	✓		✓	✓	✓	✓	✓
Directed networks						✓	✓	✓						✓	✓
Node-based deception														✓	✓
Node influence														✓	✓

deception for several reasons. First, real-world social relationships vary in their influence. For example, a person, say, Bob, probably exerts significantly more influence on his son than on his neighbor. Indeed, while it might be reasonable to assume that some social connections share relatively similar influences, it is certainly not a universal truth. So far, such variation in influence has not been covered by the deception literature. Secondly, influence has already been incorporated as an essential component in several detection algorithms (Ghosh and Lerman 2008; Lu et al. 2014; Wang and Street 2014; Ma et al. 2020). This necessarily leaves state-of-the-art deception algorithms lagging behind, failing to account for such influence-aware detection methods. Finally, and maybe more importantly, deception algorithms can make much more intelligent decisions when considering influence. Specifically, several previous deception algorithms have utilized an edge modification budget, implying the desire to perform deception with the least number of edge updates. By considering influence and direction, we can distinguish between edges' importance and carefully choose those which make the most deceptive effect to be modified.

3 Background

Community deception seeks to develop algorithms that enable a group of nodes to conceal their relationships from community detection algorithms. This study specifically investigates deception in the context of directed influence networks (DIN).

Definition 1 *(Directed influence network).* A directed influence network (DIN) is a directed weighted graph $G=(V, E, I)$, where V is the set of vertices, E is the set of edges, and I is a matrix that for each edge $(u, v) \in E$ maintains a score representing the influence that node u has on node v.

The network can be represented with an adjacency matrix $A = [I_{ij}]$, where I_{ij} denotes the influence of node i (the *influencer*) on node j (the *influenced*). It is important to note that, in a directed network like G, the influence I_{ij} may not be the same as I_{ji}. Further details regarding influence are provided in Sect. 3.1.

One significant task in social network analysis involves identifying groups of nodes that form communities. Communities have various applications, such as modeling social circles and interactions among groups of proteins. The discovery of communities enables a wide range of applications, including drug interaction discovery and recommender systems. This paper focuses on community detection where nodes belong to a single community. Specifically, we consider a non-overlapping community detection algorithm denoted as $\mathcal{A}_{det}$ that partitions V into a community structure $\bar{C} = C_1, C_2, ..., C_k$, where $C_i \subseteq V$ and $C_i \cap C_j = \emptyset$ for

Table 2 Notation Table

Symbol	Meaning	Formula
$E(C)$	The set of intra-community edges in community C	$\{(u, v)\|u, v \in C\}$
$\tilde{E}(C)$	The set of inter-community edges having one side in community C	$\{(u, v)\|u \in C \vee v \in C\}$
$\mathcal{I}_i^{\uparrow}$	Total influence outgoing from node i	$\sum_j I_{ij}$
$\mathcal{I}_j^{\downarrow}$	Total influence received by node j	$\sum_i I_{ij}$
$\mathcal{I}$	Total influence of the entire network	$\sum_{i,j \in V} I_{ij}$
η	Total intra-community influence of the network	$\sum_{C \in \bar{C}} \sum_{ij \in C} I_{ij}$
θ	total inter-community influence in network G	$\mathcal{I} - \eta$
$\bar{C}'$	Community structure excluding community C, where $(u, v) \in C$ is the edge to be modified	$\bar{C}' = \{C\|C \in \bar{C}$ and $u, v \notin C\}$
C^*	The community for which an edge (u, v) is added/deleted	$C^* = \{C\|C \in \bar{C}$ and $(u, v) \in E(C^*)\}$
$\mathcal{I}_C^{\uparrow}$	Let $C \in \bar{C}$	$\mathcal{I}_C^{\uparrow} = \sum_{i \in C} \mathcal{I}_i^{\uparrow}$
$\mathcal{I}_C^{\downarrow}$	Let $C \in \bar{C}$	$\mathcal{I}_C^{\downarrow} = \sum_{i \in C} \mathcal{I}_i^{\downarrow}$

all $i, j \in 1, ..., k$ and $i \neq j$. Moreover, we define two types of edges: an edge $(u, v) \in E$ within a community $C \subseteq V$ is referred to as an *intra-community* edge, while an edge (u, v) is an *inter-community* edge if $u \in C$, $v \in C'$, and $C' \neq C$. Table 2 summarizes the notation used throughout this paper.

3.1 Node and edge influence

Various methods have been proposed to measure node influence. Among these methods, degree centrality is widely used, quantifying a node's influence by counting the number of connections (edges) it has in the network. The underlying assumption is that nodes with more connections are more influential. Another commonly used method is eigenvector centrality, which considers both the number of connections a node has and the quality of those connections. However, these methods primarily focus on assessing node influence.

In contrast, this paper aims to investigate edge influence, specifically, the influence that a node s has on a node t. This influence can be represented as a weight assigned to the edge connecting them. Computing edge influence determines the strength of an edge's impact on the communication or interaction between two nodes.

Several methods exist to compute edge influence, including: (i) *Weighted edges*: This method assigns weights to edges based on their significance in facilitating communication or interaction between nodes. For example, in a social network, the strength of a tie between two individuals could be weighted by factors such as the frequency or intensity of their communication or the closeness of their relationship; (ii) *Link prediction*: The goal of this approach is to predict the likelihood of future edges forming between nodes in a network, based on the network's structure and other factors such as node attributes. The strength of an edge can be estimated by considering its predicted probability of forming or by analyzing the features that contribute to the prediction; (iii) *Influence propagation*: This method simulates the spread of influence or information through a network and identifies the edges that have a significant impact on this spread. For instance, the Linear Threshold Model assumes that nodes are influenced by their neighbors if the total influence from their neighbors exceeds a certain threshold. It identifies the edges that contribute the most to the overall influence propagation (Chen et al. 2010).

By exploring these methods, one can effectively capture and quantify the influence of edges within a network, enabling a comprehensive analysis of edge-centric deception strategies.

In this paper, we adopt the strategy to model influence based on a combination of edge weights and link prediction described in Kumar et al. (2016). This approach proposes two novel measures of node behavior: the goodness of a node intuitively captures how much this node is liked/trusted by other nodes, while the fairness of a node captures how fair the node is in rating other nodes' likeability or trust level. These two measures are then used to predict edge weights, that model influence in our case, via a regression model (see Kumar et al. (2016) for more details). The reason for this choice is the complexity of the approach, which is $\mathcal{O}(|E|)$, thus making the approach applicable even in large networks.

3.2 Community deception setting

This section provides an overview of the adversarial scenario in which our proposed algorithm operates. The framework of community deception on a network G involves three key actors: a detector $\mathcal{A}_{det}$, a deceptor $\mathcal{A}_{dec}$, and a target community $\mathcal{C}$. The detector aims to uncover a community structure that reflects genuine relationships within the network. However, this objective conflicts with the goals of the deceptor $\mathcal{A}_{dec}$, which aims to conceal the relationship between a specific group of nodes known as the "target community" or $\mathcal{C}$ for brevity. It is important to note that $\mathcal{C}$ represents a subset of network nodes, which can be relatively small in size, but not necessarily.

In this paper, we assume the role of the deceptor and operate under a worst-case scenario. This implies that in the current state of the network G, the target community $\mathcal{C}$ is entirely discoverable by the detector $\mathcal{A}_{det}$, i.e., $\mathcal{C} \in \bar{C}$. Consequently, we aim to introduce specific modifications to the nodes or edges of G so that $\mathcal{A}_{det}$ can no longer detect $\mathcal{C}$.

The effectiveness of deception, or the extent to which $\mathcal{C}$ remains hidden from $\mathcal{A}_{det}$, is generally quantified by a deception score (Fionda and Pirrò 2018; Waniek et al. 2018), which the deceptor $\mathcal{A}_{dec}$ aims to maximize. In this paper, we indirectly achieve this goal by minimizing modularity (Q_G) (Girvan and Newman 2002), which is commonly utilized as a quality function for community detection. Formally, the deceptor $\mathcal{A}_{dec}$ achieves deception by modifying the original network G to create a new network G'. If we denote the modularity of G and G' as Q_G and $Q_{G'}$, respectively, the deceptor's objective is to maximize the "modularity loss" $\Delta_Q = Q_G - Q_{G'}$.

In its original form, the modularity of an undirected and unweighted network with m edges can be expressed by the following equation, borrowed from Leicht and Newman (2008):

$$Q = \frac{1}{2m} \sum_{ij} \left[A_{ij} - \frac{k_i k_j}{2m} \right] \delta_{c_i, c_j} \tag{1}$$

where A_{ij} is value of edge ij in the adjacency matrix, k_i is the degree of node i, δ_{ij} is Kronecker delta, and c_i is the label of the community containing node i.

There are two main reasons for selecting modularity as the basis for our deception strategies. Firstly, modularity is widely utilized as a function for community detection (Fortunato 2010) and offers an intuitive criterion for defining a community: a group of nodes with more internal than external edges.

It is important to note that the deception score is a heuristic measurement subjectively defined by the deceptor based on their own perception of what constitutes "good" deception. In contrast, modularity is a more objective measure of effective network partitioning, enhancing our techniques' applicability.

4 Edge-based deception in DIN

This section presents IDEC, the pioneering influence-based deception approach designed specifically for directed networks with a focus on edge updates. Our approach, IDEC, is built upon modularity. The choice of modularity is driven by its intuitive nature, as it effectively captures our understanding of what constitutes a community: a cohesive group characterized by stronger connections among its members compared to connections with individuals outside the group. This intuitive appeal is crucial because we view modularity as a clustering metric rather than an objective function. Consequently, we contend that employing our deception mechanism can generally enhance the concealment level of a target community, even if the detector employs alternative clustering metrics.

4.1 Problem statement

In the context of edge-based deception, the objective of the deceptor is to introduce modifications to the edges that result in the maximum *modularity loss*. By doing so, the goal is to render it infeasible for the detector, $\mathcal{A}_{det}$, to include the target community C_t in the community structure $\bar{C}$. Mathematically, this can be formulated as an optimization problem with the following formulation:

$$\begin{aligned} \underset{G'}{\text{argmax}} \quad & \Delta_Q(G, G', \mathcal{C}) \\ \text{where} \quad & G' = (V, E') \\ & E' = (E \cup E^+) \backslash E^- \\ & E^+ \subseteq \{(u, v) | u \in \mathcal{C} \vee v \in \mathcal{C}, (u, v) \notin E\} \\ & E^- \subseteq \{(u, v) | u \in \mathcal{C} \vee v \in \mathcal{C}, (u, v) \in E\} \\ & |E^-| + |E^+| \leq \beta \end{aligned} \tag{2}$$

where Δ_Q is modularity loss described earlier, β is the budget of edge updates. The initial step is to integrate influence into directed modularity to address community deception while incorporating influence. This involves modifying the modularity metric to account for the influence between nodes in a directed network. By incorporating influence, we aim to capture the impact of node interactions on community structure more accurately.

Once the influence-based directed modularity is established, the next focus is exploring edge modifications' effects within and between communities. These edge modifications serve as the toolbox for our deception strategy. We analyze the impact of altering intra-community and inter-community

edges to determine how they can be strategically manipulated to achieve effective community concealment.

4.2 Modularity in DIN

Modularity for a directed network can be expressed as in Eq. (3), which is a slightly modified version of the one described by Leicht and Newman (2008):

$$Q = \frac{1}{m}\sum_{C\in\bar{C}}\sum_{ij\in C} I_{ij} - \frac{d_i^{out} d_j^{in}}{m} \tag{3}$$

where d_i^{out} and d_j^{in} are the out/in degrees of nodes i, j, respectively. However, since we are considering a DIN, we further modify the preceding function:

$$Q = \frac{1}{\mathcal{I}}\sum_{C\in\bar{C}}\sum_{ij\in C} I_{ij} - \frac{\mathcal{I}_i^{\uparrow}\mathcal{I}_j^{\downarrow}}{\mathcal{I}} \tag{4}$$

Now, we consider the effect of edge modifications on Δ_Q. We can simplify Eq. (4) as follows:

$$\begin{aligned} Q &= \frac{1}{\mathcal{I}}\sum_{C\in\bar{C}}\sum_{ij\in C} I_{ij} - \frac{\mathcal{I}_i^{\uparrow}\mathcal{I}_j^{\downarrow}}{\mathcal{I}} = \frac{\eta}{\mathcal{I}} - \frac{1}{\mathcal{I}^2}\sum_{C\in\bar{C}}\sum_{ij\in C}\mathcal{I}_i^{\uparrow}\mathcal{I}_j^{\downarrow} \\ &= \frac{\eta}{\mathcal{I}} - \frac{1}{\mathcal{I}^2}\sum_{C\in\bar{C}}\sum_{i\in C}\mathcal{I}_i^{\uparrow}\sum_{j\in C}\mathcal{I}_j^{\downarrow} = \frac{\eta}{\mathcal{I}} - \frac{1}{\mathcal{I}^2}\sum_{C\in\bar{C}}\mathcal{I}_C^{\uparrow}\mathcal{I}_C^{\downarrow} \end{aligned} \tag{5}$$

With the established formulation of modularity for directed influence network (DIN), we can now study edge updates' impact. Specifically, we focus on understanding how different edge modifications influence the modularity loss, denoted as Δ_Q. The modularity loss plays a crucial role as the main component of the objective function presented in Eq. (2).

By investigating the effect of edge updates on modularity loss, we aim to gain insights into the most effective strategies for manipulating edges to maximize the concealment of the target community. This analysis will inform the development of our deception algorithm and guide the selection of edge modifications that lead to significant reductions in modularity and enhance deception effectiveness.

4.3 Effect of edge updates on the modularity loss

This section establishes the theoretical basis of IDEC. We formally show the effect of edge deletion and addition on Δ_Q, considering both intra-community and inter-community edges.

4.3.1 Intra-community edges

Suppose an arbitrary intra-community edge (u, v) is deleted from a community $C^* \in \bar{C}$; Theorem 2 shows the necessary condition for this operation to cause modularity loss:

Theorem 2 *Deleting an intra-community edge results in modularity loss, $\Delta_Q > 0$, if and only if the following condition holds:*

$$\frac{\eta I_{uv}}{\mathcal{I}} + \theta + \frac{2\mathcal{I} - I_{uv}}{\mathcal{I}^2}\sum_{C\in\bar{C}}\mathcal{I}_C^{\uparrow}\mathcal{I}_C^{\downarrow} > \mathcal{I}_{C^*}^{\uparrow} + \mathcal{I}_{C^*}^{\downarrow} \tag{6}$$

Proof First, note that the modularity before modification can be expressed as:

$$Q_G = \frac{\eta}{\mathcal{I}} - \frac{1}{\mathcal{I}^2}\sum_{C\in\bar{C}}\mathcal{I}_C^{\uparrow}\mathcal{I}_C^{\downarrow} = \frac{\eta}{\mathcal{I}} - \frac{1}{\mathcal{I}^2}\left(\mathcal{I}_{C^*}^{\uparrow}\mathcal{I}_{C^*}^{\downarrow} + \sum_{C\in\bar{C}'}\mathcal{I}_C^{\uparrow}\mathcal{I}_C^{\downarrow}\right) \tag{7}$$

With slight modification of Eq. (7), we obtain modularity after intra-community edge deletion:

$$Q_{G'} = \frac{\eta - I_{uv}}{\mathcal{I} - I_{uv}} - \frac{1}{(\mathcal{I} - I_{uv})^2}\left((\mathcal{I}_{C^*}^{\uparrow} - I_{uv})(\mathcal{I}_{C^*}^{\downarrow} - I_{uv}) + \sum_{C\in\bar{C}'}\mathcal{I}_C^{\uparrow}\mathcal{I}_C^{\downarrow}\right) \tag{8}$$

With Eqs. (7) and (8), we obtain:

$$\begin{aligned} \Delta_Q =& Q_G - Q_{G'} \\ =& \left[\frac{\eta}{\mathcal{I}} - \frac{1}{\mathcal{I}^2}\left(\mathcal{I}_{C^*}^{\uparrow}\mathcal{I}_{C^*}^{\downarrow} + \sum_{C\in\bar{C}'}\mathcal{I}_C^{\uparrow}\mathcal{I}_C^{\downarrow}\right)\right] - \left[\frac{\eta - I_{uv}}{\mathcal{I} - I_{uv}} - \right. \\ & \left. \frac{1}{(\mathcal{I} - I_{uv})^2}\left((\mathcal{I}_{C^*}^{\uparrow} - I_{uv})(\mathcal{I}_{C^*}^{\downarrow} - I_{uv}) + \sum_{C\in\bar{C}'}\mathcal{I}_C^{\uparrow}\mathcal{I}_C^{\downarrow}\right)\right] \\ =& \frac{I_{uv}(\mathcal{I} - \eta)}{\mathcal{I}(\mathcal{I} - I_{uv})} + \frac{I_{uv}}{\mathcal{I}^2(\mathcal{I} - I_{uv})^2}\left[\left((2\mathcal{I} - I_{uv})\sum_{C\in\bar{C}}\mathcal{I}_C^{\uparrow}\mathcal{I}_C^{\downarrow}\right) - \right. \\ & \left. \left(\mathcal{I}^2\left(\mathcal{I}_{C^*}^{\uparrow} + \mathcal{I}_{C^*}^{\downarrow} - I_{uv}\right)\right)\right] \end{aligned} \tag{9}$$

With basic algebraic operations, we get to:

$$\Delta_Q = \frac{I_{uv}}{(\mathcal{I} - I_{uv})^2}\left[\mathcal{I} - \eta + \frac{\eta I_{uv}}{\mathcal{I}} - \mathcal{I}_{C^*}^{\uparrow} - \mathcal{I}_{C^*}^{\downarrow} + \frac{(2\mathcal{I} - I_{uv})}{\mathcal{I}^2}\sum_{C\in\bar{C}}\mathcal{I}_C^{\uparrow}\mathcal{I}_C^{\downarrow}\right] \tag{10}$$

Equation (10) shows that the sign of Δ_Q depends on the term between square brackets, which is positive if and only if:

$$\frac{\eta I_{uv}}{\mathcal{I}} + \theta + \frac{2\mathcal{I} - I_{uv}}{\mathcal{I}^2}\sum_{C\in\bar{C}}\mathcal{I}_C^{\uparrow}\mathcal{I}_C^{\downarrow} > \mathcal{I}_{C^*}^{\uparrow} + \mathcal{I}_{C^*}^{\downarrow} \tag{11}$$

□

The condition presented in inequality (6) has two important implications. Firstly, deleting an intra-community edge

will likely result in modularity loss. This is primarily because, in most real-world networks, the total influence of the target community (θ) is greater than the sum of the external influence flowing into and out of the target community ($\mathcal{I}^{\uparrow}_{C^*} + \mathcal{I}^{\downarrow}_{C^*}$). This inequality holds true, especially when the target community is smaller than the overall network. Hence, removing intra-community edges often leads to a decrease in modularity. Secondly, the left side of inequality (6) increases with higher values of $\mathcal{I}_{uv}$. This implies that selecting edges with a higher influence for deletion is preferred to maximize the potential modularity loss. Considering this, IDEC incorporates intra-community edge deletion as a potential operation at each algorithm iteration. Now, we proceed to the next theorem, which addresses the addition of intra-community edges and its implications.

Theorem 3 *Adding an intra-community edge results in modularity loss, $\Delta_Q > 0$, if and only if the following condition holds:*

$$\frac{\eta I_{uv}}{\mathcal{I}} + \mathcal{I}^{\uparrow}_{C^*} + \mathcal{I}^{\downarrow}_{C^*} > \theta + \frac{(2\mathcal{I} + I_{uv})}{\mathcal{I}^2} \sum_{C \in \bar{C}} \mathcal{I}^{\uparrow}_{C} \mathcal{I}^{\downarrow}_{C} \tag{12}$$

Proof Modularity after intra-community edge addition can be expressed as:

$$Q_{G^*} = \frac{\eta + I_{uv}}{\mathcal{I} + I_{uv}} - \frac{1}{(\mathcal{I} + I_{uv})^2} \Bigg((\mathcal{I}^{\uparrow}_{C^*} + I_{uv})(\mathcal{I}^{\downarrow}_{C^*} + I_{uv}) + \sum_{C \in \bar{C}'} \mathcal{I}^{\uparrow}_{C} \mathcal{I}^{\downarrow}_{C} \Bigg) \tag{13}$$

Using equation (13) we can compute Δ_Q:

$$\begin{aligned} \Delta_Q &= Q_G - Q_{G^*} \\ &= \left[\frac{\eta}{\mathcal{I}} - \frac{1}{\mathcal{I}^2} \left(\mathcal{I}^{\uparrow}_{C^*} \mathcal{I}^{\downarrow}_{C^*} + \sum_{C \in \bar{C}'} \mathcal{I}^{\uparrow}_{C} \mathcal{I}^{\downarrow}_{C} \right) \right] - \\ &\quad \left[\frac{\eta + I_{uv}}{\mathcal{I} + I_{uv}} - \frac{1}{(\mathcal{I} + I_{uv})^2} \left((\mathcal{I}^{\uparrow}_{C^*} + I_{uv})(\mathcal{I}^{\downarrow}_{C^*} + I_{uv}) + \sum_{C \in \bar{C}'} \mathcal{I}^{\uparrow}_{C} \mathcal{I}^{\downarrow}_{C} \right) \right] \end{aligned} \tag{14}$$

With basic algebraic operations, we get to:

$$\Delta_Q = \frac{I_{uv}}{(\mathcal{I} + I_{uv})^2} \left[\frac{\eta(\mathcal{I} + I_{uv})}{\mathcal{I}} - \mathcal{I} + \mathcal{I}^{\uparrow}_{C^*} + \mathcal{I}^{\downarrow}_{C^*} - \frac{(2\mathcal{I} + I_{uv})}{\mathcal{I}^2} \sum_{C \in \bar{C}} \mathcal{I}^{\uparrow}_{C} \mathcal{I}^{\downarrow}_{C} \right] \tag{15}$$

Equation (15) shows that the sign of Δ_Q depends on the term between square brackets, which is positive if and only if:

$$\frac{\eta(\mathcal{I} + I_{uv})}{\mathcal{I}} - \mathcal{I} + \mathcal{I}^{\uparrow}_{C^*} + \mathcal{I}^{\downarrow}_{C^*} - \frac{(2\mathcal{I} + I_{uv})}{\mathcal{I}^2} \sum_{C \in \bar{C}} \mathcal{I}^{\uparrow}_{C} \mathcal{I}^{\downarrow}_{C} > 0 \tag{16}$$

which can be reduced to:

$$\frac{\eta I_{uv}}{\mathcal{I}} + \mathcal{I}^{\uparrow}_{C^*} + \mathcal{I}^{\downarrow}_{C^*} > \theta + \frac{(2\mathcal{I} + I_{uv})}{\mathcal{I}^2} \sum_{C \in \bar{C}} \mathcal{I}^{\uparrow}_{C} \mathcal{I}^{\downarrow}_{C} \tag{17}$$

□

Theorem 3 demonstrates that adding an intra-community edge is unlikely to result in modularity loss; instead, it is more likely to increase modularity. This conclusion is derived through a similar argument as with intra-community edge deletion. Consequently, to maximize the potential modularity loss, IDEC does not consider the addition of intra-community edges as part of its operations.

4.3.2 Inter-community edges

Let $C^u, C^v \in \bar{C}$ be two arbitrary communities, then consider a inter-community edge (u, v), where $u \in C^u$ and $v \in C^v$. This section studies how the deletion/addition of (u, v) affects Δ_Q.

Theorem 4 *Deleting an inter-community edge will increase modularity if and only if:*

$$\eta \mathcal{I} I_{uv} + (2\mathcal{I} - I_{uv}) \sum_{C \in \bar{C}} \mathcal{I}^{\uparrow}_{C} \mathcal{I}^{\downarrow}_{C} > \mathcal{I}^2 \left(\eta - \mathcal{I}^{\downarrow}_{C^u} - \mathcal{I}^{\uparrow}_{C^v} \right) \tag{18}$$

Proof Note that the modularity before edge modification can be expressed as:

$$\begin{aligned} Q_G &= \frac{\eta}{\mathcal{I}} - \frac{1}{\mathcal{I}^2} \sum_{C \in \bar{C}} \mathcal{I}^{\uparrow}_{C} \mathcal{I}^{\downarrow}_{C} \\ &= \frac{\eta}{\mathcal{I}} - \frac{1}{\mathcal{I}^2} \left((\mathcal{I}^{\uparrow}_{C^u} \mathcal{I}^{\downarrow}_{C^u}) + (\mathcal{I}^{\uparrow}_{C^v} \mathcal{I}^{\downarrow}_{C^v}) + \sum_{C \in \bar{C}'} \mathcal{I}^{\uparrow}_{C} \mathcal{I}^{\downarrow}_{C} \right) \end{aligned} \tag{19}$$

Modularity after inter-community edge deletion can be expressed as:

$$Q_{G^*} = \frac{\eta}{\mathcal{I} - I_{uv}} - \frac{1}{(\mathcal{I} - I_{uv})^2} \left(((\mathcal{I}^{\uparrow}_{C^u} - I_{uv}) \mathcal{I}^{\downarrow}_{C^u}) + (\mathcal{I}^{\uparrow}_{C^v} (\mathcal{I}^{\downarrow}_{C^v} - I_{uv})) + \sum_{C \in \bar{C}'} \mathcal{I}^{\uparrow}_{C} \mathcal{I}^{\downarrow}_{C} \right) \tag{20}$$

Then we consider Δ_Q:

$$\begin{aligned}\Delta_Q &= Q_G - Q_{G^*} \\ &= \frac{\eta}{\mathcal{I}} - \frac{1}{\mathcal{I}^2}\left((\mathcal{I}^{\uparrow}_{C^u}\mathcal{I}^{\downarrow}_{C^u}) + (\mathcal{I}^{\uparrow}_{C^v}\mathcal{I}^{\downarrow}_{C^v}) + \sum_{C\in\bar{C}'}\mathcal{I}^{\uparrow}_{C}\mathcal{I}^{\downarrow}_{C}\right) \\ &\quad - \left[\frac{\eta}{\mathcal{I}-I_{uv}}\right. \\ &\quad - \frac{1}{(\mathcal{I}-I_{uv})^2}\left((\mathcal{I}^{\uparrow}_{C^u}-I_{uv})\mathcal{I}^{\downarrow}_{C^u}) + (\mathcal{I}^{\uparrow}_{C^v}(\mathcal{I}^{\downarrow}_{C^v}-I_{uv})\right) \\ &\quad \left. + \sum_{C\in\bar{C}'}\mathcal{I}^{\uparrow}_{C}\mathcal{I}^{\downarrow}_{C})\right)\Bigg]\end{aligned} \tag{21}$$

With further algebraic manipulation, equation (21) can be reduced to:

$$\begin{aligned}\Delta_Q = \frac{I_{uv}}{\mathcal{I}^2(\mathcal{I}-I_{uv})^2}\Bigg[&\eta\mathcal{I}I_{uv} - \mathcal{I}^2\left(\eta - \mathcal{I}^{\downarrow}_{C^u} - \mathcal{I}^{\uparrow}_{C^v}\right) \\ &+(2\mathcal{I}-I_{uv})\sum_{C\in\bar{C}}\mathcal{I}^{\uparrow}_{C}\mathcal{I}^{\downarrow}_{C}\Bigg]\end{aligned} \tag{22}$$

Note that the sign of Δ_Q depends on the term inside square brackets. Specifically, inter-community edge deletion will cause modularity loss (i.e., $\Delta_Q > 0$) if and only if:

$$\eta\mathcal{I}I_{uv} + (2\mathcal{I}-I_{uv})\sum_{C\in\bar{C}}\mathcal{I}^{\uparrow}_{C}\mathcal{I}^{\downarrow}_{C} > \mathcal{I}^2\left(\eta - \mathcal{I}^{\downarrow}_{C^u} - \mathcal{I}^{\uparrow}_{C^v}\right) \tag{23}$$

□

Theorem 4 demonstrates that deleting an inter-community edge will generally increase modularity in practical scenarios. This is because removing inter-community edges strengthens the existing community structure, possibly exposing the target community $\mathcal{C}$. As a result, IDEC does not include the operation of deleting inter-community edges in its strategy.

Theorem 5 *Adding an inter-community edge results in modularity loss, $\Delta_Q > 0$, if and only if:*

$$\eta\mathcal{I}I_{uv} + \mathcal{I}^2\left(\eta + \mathcal{I}^{\downarrow}_{C^u} + \mathcal{I}^{\uparrow}_{C^v}\right) > (2\mathcal{I}+I_{uv})\sum_{C\in\bar{C}}\mathcal{I}^{\uparrow}_{C}\mathcal{I}^{\downarrow}_{C} \tag{24}$$

Proof Modularity before node addition Δ_Q is again expressed by (11). Now, modularity after inter-community edge addition can be expressed as:

$$\begin{aligned}Q_{G^*} = \frac{\eta}{\mathcal{I}+I_{uv}} - \frac{1}{(\mathcal{I}+I_{uv})^2}&\left((\mathcal{I}^{\uparrow}_{C^u}+I_{uv})\mathcal{I}^{\downarrow}_{C^u}) + (\mathcal{I}^{\uparrow}_{C^v}(\mathcal{I}^{\downarrow}_{C^v}+I_{uv})) + \sum_{C\in\bar{C}'}\mathcal{I}^{\uparrow}_{C}\mathcal{I}^{\downarrow}_{C}\right)\end{aligned} \tag{25}$$

We now consider Δ_Q:

$$\begin{aligned}\Delta_Q &= Q_G - Q_{G^*} \\ &= \frac{\eta}{\mathcal{I}} - \frac{1}{\mathcal{I}^2}\left((\mathcal{I}^{\uparrow}_{C^u}\mathcal{I}^{\downarrow}_{C^u}) + (\mathcal{I}^{\uparrow}_{C^v}\mathcal{I}^{\downarrow}_{C^v}) + \sum_{C\in\bar{C}'}\mathcal{I}^{\uparrow}_{C}\mathcal{I}^{\downarrow}_{C}\right) \\ &\quad - \left[\frac{\eta}{\mathcal{I}+I_{uv}}\right. \\ &\quad - \frac{1}{(\mathcal{I}+I_{uv})^2}\left((\mathcal{I}^{\uparrow}_{C^u}+I_{uv})\mathcal{I}^{\downarrow}_{C^u}) + (\mathcal{I}^{\uparrow}_{C^v}(\mathcal{I}^{\downarrow}_{C^v}+I_{uv})\right) \\ &\quad \left. + \sum_{C\in\bar{C}'}\mathcal{I}^{\uparrow}_{C}\mathcal{I}^{\downarrow}_{C})\right)\Bigg]\end{aligned} \tag{26}$$

This can be reduced to:

$$\begin{aligned}\Delta_Q &= \frac{\eta I_{uv}}{\mathcal{I}(\mathcal{I}+I_{uv})} - \frac{1}{\mathcal{I}^2}\left((\mathcal{I}^{\uparrow}_{C^u}\mathcal{I}^{\downarrow}_{C^u}) + (\mathcal{I}^{\uparrow}_{C^v}\mathcal{I}^{\downarrow}_{C^v})\right. \\ &\quad \left. + \sum_{C\in\bar{C}'}\mathcal{I}^{\uparrow}_{C}\mathcal{I}^{\downarrow}_{C}\right) \\ &\quad + \frac{1}{(\mathcal{I}+I_{uv})^2}\left((\mathcal{I}^{\uparrow}_{C^u}+I_{uv})\mathcal{I}^{\downarrow}_{C^u}) + (\mathcal{I}^{\uparrow}_{C^v}(\mathcal{I}^{\downarrow}_{C^v}+I_{uv})\right) \\ &\quad \left. + \sum_{C\in\bar{C}'}\mathcal{I}^{\uparrow}_{C}\mathcal{I}^{\downarrow}_{C})\right)\end{aligned} \tag{27}$$

And finally, we reach:

$$\begin{aligned}\Delta_Q = \frac{I_{uv}}{\mathcal{I}^2(\mathcal{I}+I_{uv})^2}\Bigg[&\eta\mathcal{I}I_{uv} + \mathcal{I}^2\left(\eta + \mathcal{I}^{\downarrow}_{C^u} + \mathcal{I}^{\uparrow}_{C^v}\right) \\ &-(2\mathcal{I}+I_{uv})\sum_{C\in\bar{C}}\mathcal{I}^{\uparrow}_{C}\mathcal{I}^{\downarrow}_{C}\Bigg]\end{aligned} \tag{28}$$

Again, we note that the sign of Δ_Q depends on the term inside square brackets. Specifically, inter-community edge addition will result in modularity loss (i.e., $\Delta_Q > 0$) if and only if:

$$\eta\mathcal{I}I_{uv} + \mathcal{I}^2\left(\eta + \mathcal{I}^{\downarrow}_{C^u} + \mathcal{I}^{\uparrow}_{C^v}\right) > (2\mathcal{I}+I_{uv})\sum_{C\in\bar{C}}\mathcal{I}^{\uparrow}_{C}\mathcal{I}^{\downarrow}_{C} \tag{29}$$

□

Theorem 5 establishes that adding an inter-community edge in a network with sufficiently large $\mathcal{I}$ and η will likely result in modularity loss. Furthermore, inequality 24 suggests that selecting a higher influence edge increases the probability of modularity loss.

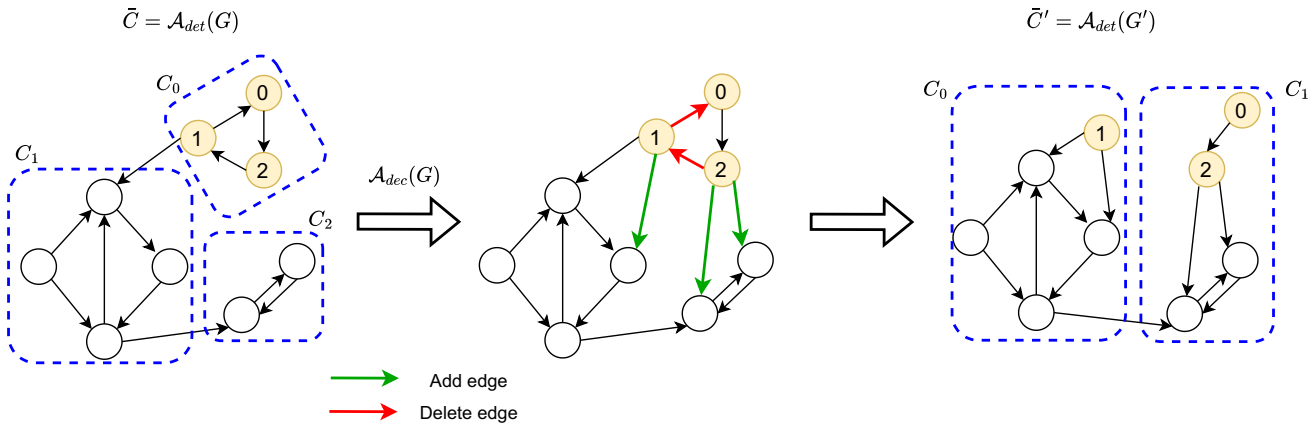

Fig. 1 Using edge deletion/addition to hiding a target community. Yellow nodes are members of the target community $\mathcal{C}$

It is important to note that the previous four theorems align with the intuition of modularity, emphasizing strong internal connections within communities and weak external connections. Deleting an intra-community edge naturally leads to modularity loss since it weakens the internal connections within the community. Similar reasoning can be applied to the addition of inter-community edges. Overall, these theorems provide insights into the effects of different edge modifications on modularity, supporting the decision-making process of IDEC in its deception strategy.

4.3.3 Running example

We provide a toy example (Fig. 1) to illustrate the three main stages of community deception through edge modifications. In the initial network state (leftmost side), the target community is completely discoverable by $\mathcal{A}_{det}$, indicated by $\mathcal{C} \in \bar{C}$. In the next stage (center), $\mathcal{A}_{dec}$ applies a series of edge deletions and additions within the target community, resulting in a modified network G'. Finally, when $\mathcal{A}_{det}$ is applied to G', it no longer assigns all target community members to a single community. The dispersion of target community members across different communities is desirable from a deception perspective and leads to a higher score. This example visually demonstrates the process of community deception through edge modifications and highlights the objective of dispersing target community members to increase concealment.

5 Node-based deception in DIN

We introduce our novel node-based community deception framework for DIN, utilizing the concept of modularity. Node-based operations are a higher-level deception mode that complements the previously discussed lower-level edge operations. The inclusion of node-based operations is crucial for effective community deception. The target community $\mathcal{C}$ is a dynamic entity that undergoes continuous changes, including adding or removing members. It is essential for $\mathcal{C}$ to accurately assess the impact of such changes on its concealment level, as reflected by the deception score. Existing deception algorithms primarily focus on edge modifications while keeping nodes intact. However, we argue that node-based operations are necessary to accommodate the inclusion or departure of new nodes in $\mathcal{C}$. By considering node-based operations, we can develop more comprehensive deception strategies that leverage the full range of possible modifications within the network.

5.1 Problem statement

As in edge-based deception, we now establish the theoretical basis for designing our node-based deception algorithm. Specifically, we describe the deceptor's goal with an optimization problem:

$$
\begin{aligned}
\underset{G'}{\text{argmax}} \quad & \Delta_Q(G, G', \mathcal{C}) \\
\text{where} \quad & G' = (V', E') \\
& V' = (V \cup V^+) \backslash V^- \\
& V^+ \subset \{v | v \notin V\} \\
& V^- \subset \mathcal{C} \\
& E' = (E \cup E^+) \backslash E^- \\
& E^- = \{(u, v) | u \in V^- \vee v \in V^-, (u, v) \in E\} \\
& E^+ \subseteq \{(u, v) | u \in V^+ \vee v \in V^+, (u, v) \notin E\} \\
& |E^-| + |E^+| \leq \beta
\end{aligned}
\tag{30}
$$

It is important to note the distinction between edge-based and node-based deception regarding the optimization problems and the underlying mechanisms involved. In Problem (2), which represents edge-based deception, the focus is on directly modifying the edge set of the network to find the optimal modified network G'. This involves adding or removing

edges to achieve the desired deception objective. On the other hand, in Problem (30), which represents node-based deception, the optimization problem incorporates node operations, reflected in updates to the node set V. However, it is crucial to highlight that the mechanism of edge updates in node-based deception differs from that in edge-based deception. In node-based deception, edge updates occur indirectly due to node updates. In other words, adding or removing nodes indirectly leads to corresponding changes in the edge set.

Although the edge updates in Problem (30) are a consequence of node operations, it is important to recognize that edge and node operations can be translated into each other regarding the budget β. Node operations can be seen as performing multiple edge changes simultaneously, and thus, the budget β represents the overall limit on the number of edge modifications regardless of whether they are achieved through direct edge updates or indirect node updates. Therefore, while the optimization problems and the mechanisms involved differ between edge-based and node-based deception, both approaches aim to achieve the desired deception objective within the given budget of edge modifications.

5.2 Effect of node updates on the modularity loss

In the remaining part of this section, we conduct an analysis of the impact of node deletions, additions, and movements on modularity loss, focusing on the general effect rather than relying on any particular community detection algorithm.

5.2.1 Node deletions

Let u be an arbitrary node to be deleted from some community $C^* \in \bar{C}$. The following Theorem shows modularity loss caused by this operation in the context of DIN.

Theorem 6 *Deleting a node $u \in C^*$, where $C^* \in \bar{C}$, results in the following modularity loss:*

$$\Delta_Q^- = \frac{\eta}{\mathcal{I}} - \frac{1}{\mathcal{I}^2}\sum_{C\in\bar{C}} \mathcal{I}_C^{\uparrow}\mathcal{I}_C^{\downarrow} - \left[\frac{\eta - \mathcal{I}_u^{C^*}}{\mathcal{I} - I_u} - \frac{1}{(\mathcal{I}-I_u)^2}\sum_{C\in\bar{C}'} \mathcal{I}_C^{\uparrow}\mathcal{I}_C^{\downarrow}\right] \tag{31}$$

where $\bar{C}'$ is the community structure after deleting u.

Proof The post-deletion modularity of G can be expressed as:

$$Q_{G'} = \frac{\eta - \mathcal{I}_u^{C^*}}{\mathcal{I} - I_u} - \frac{1}{(\mathcal{I}-I_u)^2}\sum_{C\in\bar{C}'} \mathcal{I}_C^{\uparrow}\mathcal{I}_C^{\downarrow} \tag{32}$$

where $\mathcal{I}_u^{C^*}$ is the influence of u intra-C^*. Now, we can express modularity loss Δ_Q^- as:

$$\begin{aligned}\Delta_Q^- &= Q_G - Q_{G'} \\ &= \frac{\eta}{\mathcal{I}} - \frac{1}{\mathcal{I}^2}\sum_{C\in\bar{C}} \mathcal{I}_C^{\uparrow}\mathcal{I}_C^{\downarrow} \\ &\quad - \left[\frac{\eta - \mathcal{I}_u^{C^*}}{\mathcal{I} - I_u} - \frac{1}{(\mathcal{I}-I_u)^2}\sum_{C\in\bar{C}'} \mathcal{I}_C^{\uparrow}\mathcal{I}_C^{\downarrow}\right]\end{aligned} \tag{33}$$

which means that $\Delta_Q^- > 0$ if and only if:

$$\underbrace{\frac{\eta}{\mathcal{I}} - \frac{1}{\mathcal{I}^2}\sum_{C\in\bar{C}} \mathcal{I}_C^{\uparrow}\mathcal{I}_C^{\downarrow}}_{a} > \underbrace{\frac{\eta - \mathcal{I}_u^{C^*}}{\mathcal{I} - I_u} - \frac{1}{(\mathcal{I}-I_u)^2}\sum_{C\in\bar{C}'} \mathcal{I}_C^{\uparrow}\mathcal{I}_C^{\downarrow}}_{b} \tag{34}$$

□

Inequality (34) sets the condition u must meet to promote deception. Satisfying this condition depends on several factors, including the network total influence $\mathcal{I}$ and the intra/inter influence ratio of node u. Therefore, the deceptor's goal would be to choose a deletion candidate that minimizes part b in (34). As the inequality shows, the best bet would be to choose a node with a high intra to inter-community influence ratio, as this will decrease the first term of b. This result conforms with the idea presented in Sect. 6.2, which says that deleting intra-community edges has a high potential to cause modularity loss. However, this puts forward a trade-off between maintaining connectivity among community members and achieving maximum deception. While Theorem 6 applies to any community in $\bar{C}$, the deceptor controls only $\mathcal{C}$ members. Therefore, node deletion should be generally avoided unless the gain in terms of deception outweighs the loss of connectivity within $\mathcal{C}$.

5.2.2 Node additions

Let u be a new node added to some community $C^* \in \bar{C}$, to obtain an updated community $C_u^* = C^* \cup \{u\}$. This addition results in a modified network G', and a modified community structure $\bar{C}' = \{\bar{C} \cup C_u^*\} \backslash \{C^*\}$. The following theorem shows this addition's necessary condition to cause DIN modularity loss.

Theorem 7 *Adding a node $u \notin V$ to a community $C^* \in \bar{C}$ results in the following modularity loss:*

$$\Delta_Q^+ = \frac{\eta}{\mathcal{I}} - \frac{1}{\mathcal{I}^2}\sum_{C\in\bar{C}} \mathcal{I}_C^{\uparrow}\mathcal{I}_C^{\downarrow}$$

$$-\left[\frac{\eta + \mathcal{I}_u^{C^*}}{\mathcal{I} + I_u} - \frac{1}{(\mathcal{I} + I_u)^2} \sum_{C \in \bar{C}'} \mathcal{I}_C^{\uparrow} \mathcal{I}_C^{\downarrow}\right] \tag{35}$$

Proof The post-addition modularity of G can be expressed as:

$$Q_{G'} = \frac{\eta + \mathcal{I}_u^{C^*}}{\mathcal{I} + I_u} - \frac{1}{(\mathcal{I} + I_u)^2} \sum_{C \in \bar{C}'} \mathcal{I}_C^{\uparrow} \mathcal{I}_C^{\downarrow} \tag{36}$$

where $\mathcal{I}_u^{C^*}$ is the total influence of u connected with C^*. Now, we can express modularity loss Δ_Q^+ as:

$$\begin{aligned} \Delta_Q^+ &= Q_G - Q_{G'} \\ &= \frac{\eta}{\mathcal{I}} - \frac{1}{\mathcal{I}^2} \sum_{C \in \bar{C}} \mathcal{I}_C^{\uparrow} \mathcal{I}_C^{\downarrow} \\ &\quad - \left[\frac{\eta + \mathcal{I}_u^{C^*}}{\mathcal{I} + I_u} - \frac{1}{(\mathcal{I} + I_u)^2} \sum_{C \in \bar{C}'} \mathcal{I}_C^{\uparrow} \mathcal{I}_C^{\downarrow}\right] \end{aligned} \tag{37}$$

this shows that $\Delta_Q^+ > 0$ if and only if:

$$\underbrace{\frac{\eta}{\mathcal{I}} - \frac{1}{\mathcal{I}^2} \sum_{C \in \bar{C}} \mathcal{I}_C^{\uparrow} \mathcal{I}_C^{\downarrow}}_{a} > \underbrace{\frac{\eta + \mathcal{I}_u^{C^*}}{\mathcal{I} + I_u} - \frac{1}{(\mathcal{I} + I_u)^2} \sum_{C \in \bar{C}'} \mathcal{I}_C^{\uparrow} \mathcal{I}_C^{\downarrow}}_{b} \tag{38}$$

□

As observed earlier, the deceptor aims to minimize part b in the above-mentioned inequality. Therefore, it is preferable for the new node to have minimal connections with members of the C^*. This preference aligns with the idea of minimizing intra-community edges within the target community, as discussed previously. While node deletions are typically focused on target community members, adding new nodes theoretically allows for flexibility in terms of deception. However, there are practical considerations and potential limitations to be aware of. Firstly, adding new nodes to any community without restrictions is often impractical in real-world scenarios due to the requirement of global network knowledge on the deceptor's part. Secondly, even in rare cases where the deceptor possesses such knowledge, adding new nodes to communities unrelated to the target community ($\mathcal{C}$) will not significantly impact its intra-/inter-edge density ratio, and thus, it will not effectively improve its deception score. As an alternative approach, we can heuristically define two rules inspired by the definition of modularity, which states that a community should be densely connected internally and sparsely connected externally. By considering Theorems 5 to 7, we can enhance the effectiveness of node addition in concealing the target community. These rules can guide the deceptor's decisions regarding which communities to add new nodes to, considering the desired deception objectives and leveraging the insights gained from the analysis.

- The new node, say u, shall be added to a community C^* with highest mutual influence with $\mathcal{C}$.
- u shall have maximum inter-community edges with $\mathcal{C}$.

The first rule focuses on selecting destination communities for node addition that already have relatively high inter-community edges connecting them to the target community ($\mathcal{C}$). According to the definition of modularity, the intra/inter-edge density ratio between these communities is not as high as with neighboring communities. By adding new edges between them, the node u will have better chances of influencing and potentially changing its community structure. This rule aims to exploit the connections between the target community and the chosen destination community to enhance deception. The second rule further emphasizes increasing the chances of modifying the community structure. It suggests selecting a destination community with a higher overall inter-community edge density, making it more susceptible to changes and alterations caused by node additions.

A toy example is presented in Fig. 2 to provide a clearer understanding of community deception through node addition. The initial network state on the left shows two communities, C_1 and $\mathcal{C}$, discovered by the detection algorithm ($\mathcal{A}_{det}$). This represents a worst-case scenario where the target community (yellow nodes) is completely detectable. Next, the deceptor algorithm ($\mathcal{A}_{dec}$) is applied, and two new nodes with additional edges (green) are strategically added, as shown in the middle part. The selection of new edges follows the heuristics described earlier to reinforce the connection between certain target community nodes and an external community (e.g., node 1 in the example). Finally, the rightmost part of Fig. 2 illustrates the modified community structure detected by $\mathcal{A}_{det}$ after the network modifications. It can be observed that node 1 is now assigned to community 1, effectively separated from the target community, resulting in a higher deception score. This example illustrates how node addition, guided by the defined rules and heuristics, can impact the community structure and improve the concealment of the target community, leading to a higher deception score.

5.3 Node movements

Node movement is the third type of node-based operation considered in community deception. Unlike node deletion and addition, which affect both the set of nodes V and the set of edges E, node movement only involves modifying the edges. It is important to differentiate node movement

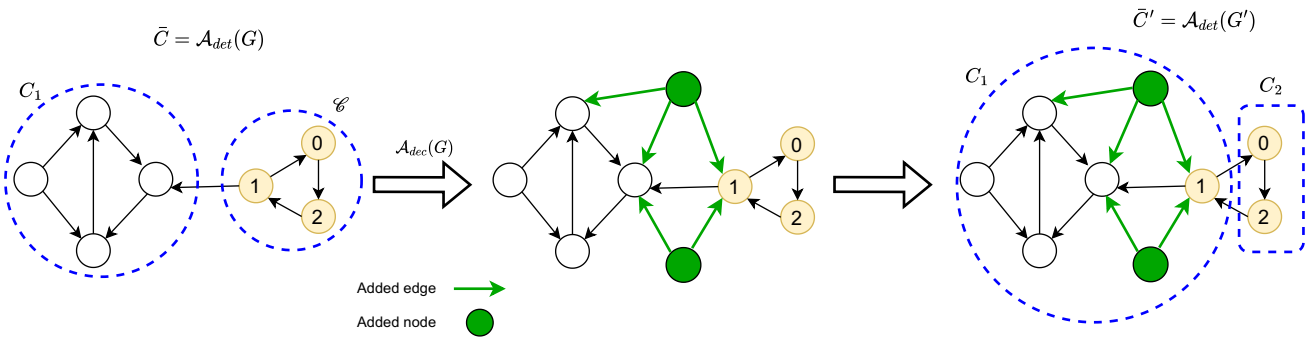

Fig. 2 Using node addition to hide a target community. On the left side, we can see the initial state of the network—G, where $\mathcal{C}$ is completely visible to $\mathcal{A}_{det}$. The deceptor adding two new nodes (in green) $\mathcal{A}_{det}$ misidentifies node 1 and separates it from the rest of $\mathcal{C}$

from edge-based operations discussed earlier. Although only the edges are modified, the goal is to move a specific node, denoted as u, from a source community C_i to a destination community C_j while altering only the edges adjacent to node u. Theorem 8 provides a quantitative analysis of the effect of node movement on network modularity. It establishes the condition under which node movement does not cause a modularity loss. The theorem provides valuable insights into the potential consequences of node movement and guides the deceptor in making informed decisions during the deception process.

Theorem 8 *The network's modularity after moving a node u from its community of origin $C_i \in \bar{C}$ to a destination community $C_j \in \bar{C}$ equals:*

$$Q^m_{G'} = \frac{\eta - \mathcal{I}_u^{C_i} + \mathcal{I}_u^{C_j}}{\mathcal{I} - I_u + I_u^*} - \frac{1}{(\mathcal{I} - I_u + I_u^*)^2} \sum_{C \in \bar{C}^m} \mathcal{I}_C^{\uparrow} \mathcal{I}_C^{\downarrow} \quad (39)$$

where $\mathcal{I}_u^{C_u}$ and $\mathcal{I}_u^{C_j}$ denote intra-community influence of u within origin and destination communities, respectively.

Equation (39) hints at how and which properties of the moved node affect the network's modularity after movement. For example, (39) shows that increasing the node mutual influence with its destination community $\mathcal{I}_u^{C_j}$, increases the new modularity $Q^m_{G'}$, meaning the new positioning of the node is enforced. However, at the same time, we must consider that adding more edges to the destination is costly in terms of the budget β.

5.3.1 Running example

We give another toy example of how node movement can foster community deception. Figure 3 shows how a given node (node 2) moves from its original community to a destination community. The leftmost part of the figure shows the network's initial state, where the detection algorithm $\mathcal{A}_{det}$ can identify the target community $\mathcal{C}$ as a single community. Then, going to the right, we see that the deceptor $\mathcal{A}_{dec}$ moves node 2 by altering its connections: deleting two internal edges and adding two new edges with community C_2. Finally, when $\mathcal{A}_{det}$ is applied to the modified network G', it assigns node 2 to the community C_2.

Apart from hiding a target community, node movement can also help hide an individual node. Indeed, referring to Fig. 3 once again, we see that we achieved both goals: the target $\mathcal{C}$ became dispersed among two communities, and a single node (e.g., node 2) became effectively disassociated with the target community. Moreover, since node movement involves changing only edges incident to a single node, such operations can be performed individually by the node itself. This can be of practical value if, for example, some node would like to disassociate itself from a given community.

6 Community deception algorithms

In this section, we present two algorithms for community deception: IDEC, an edge-based algorithm, and INDEC, a node-based algorithm. These algorithms address the edge-based and node-based community deception problems, respectively. The IDEC algorithm utilizes Theorems 2-5 to guide the selection of edge operations. It focuses on achieving community deception through individual edge modifications, considering one edge operation at each iteration. By leveraging these theorems, IDEC maximizes the potential for modularity loss in the target community. On the other hand, the INDEC algorithm is a novel node-based algorithm that operates at a higher level of deception. It performs node-by-node operations, causing multiple edge modifications in each iteration. The algorithm takes advantage of the insights provided by Theorems 6-8 to determine the most effective node-based operations for achieving community deception. It is worth noting that while IDEC focuses on micro-level deception through individual edge modifications, INDEC operates at a macro level, considering node movements and their resulting edge modifications. By combining edge and

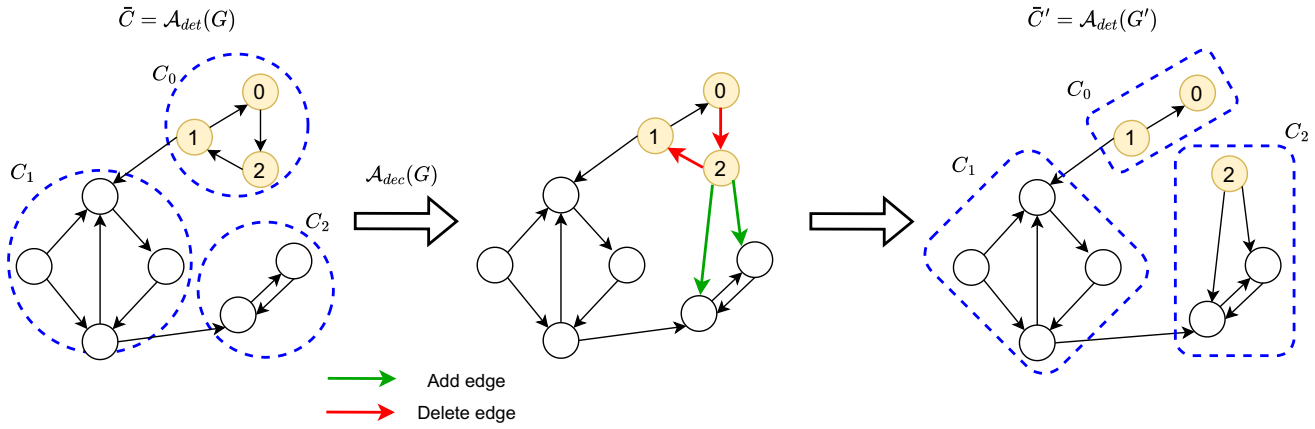

Fig. 3 Moving node 2 from community C_0 to C_2 by modifying its edges

node operations, both algorithms provide comprehensive strategies for community deception. Furthermore, a detailed complexity analysis is provided for each algorithm, ensuring an understanding of their computational efficiency and feasibility in practical applications.

6.1 Summary of the results

Before describing the algorithms, we summarize the edge/node operations with the highest potential to cause modularity loss according to the earlier theoretical analysis. Our proposed algorithms IDEC and INDEC use some or all these operations to enhance deception performance. Table 3 shows each operation, pointing to the theorem that justifies it and the algorithm where it has been employed.

6.2 Edge-based deception via IDEC

We now introduce our first contributed algorithm: IDEC. This edge-based greedy algorithm uses the results obtained in Theorems 2-5. First, we emphasize that IDEC allows edge modifications only by target community members. We can divide the algorithm into three main stages: finding candidate edges (lines 2-4), computing modularity loss (lines 5-7), and choosing the best edge operation (lines 8-18). As shown in Table 3, the most profitable edge operations are intra-community edge deletion and inter-community edge addition. Thus, IDEC considers only these operations to increase the likelihood of inflicting modularity loss and reduce the search space.

First, the procedure in line 3 selects one intra-community edge from the target community $\mathcal{C}$ as a deletion candidate. It takes the highest influence edge to maximize the possibility of causing modularity loss. Next, lines 3 and 4 select two inter-community edges for addition. The first edge is incoming, and the second is outgoing. As Algorithm 1 indicates,

Algorithm 1 The IDEC deception algorithm.

Input: Network G, target community $\mathcal{C} \subset V$, community structure $\bar{C}$.
Parameters: Edge budget β
Output:Network $G' = (V, E')$

1: **do**
2: intraEdg $\leftarrow$ getHighestIntra($\mathcal{C}$)
3: select$(n_u, n_v) \notin E, n_u \in C_i, n_v \in \mathcal{C} \cap C_j, C_i \in$ argmax$(\mathcal{I}^{\downarrow}_{C_i}), C_j \in$ argmax$(\mathcal{I}^{\uparrow}_{C_j})$
4: select$(n_p, n_t) \notin E, n_p \in C_l \cap \mathcal{C}, n_t \in C_m, C_l \in$ argmax$(\mathcal{I}^{\downarrow}_{C_l}), C_m \in$ argmax$(\mathcal{I}^{\uparrow}_{C_m})$
5: $\mathcal{ML}_{del} \leftarrow$ getDelLoss$(intraEdg, \bar{C}, G)$
6: $\mathcal{ML}^{\uparrow}_{add} \leftarrow$ getAddLoss$((n_p, n_t), \bar{C}, G)$
7: $\mathcal{ML}^{\downarrow}_{add} \leftarrow$ getAddLoss$((n_u, n_v), \bar{C}, G)$
8: **if** $\mathcal{ML}_{del} \geq \max(\mathcal{ML}^{\uparrow}_{add}, \mathcal{ML}^{\downarrow}_{add})$ and $\mathcal{ML}_{del} > 0$ **then**
9: $G \leftarrow (V, E \backslash \{\text{intraEdg}\})$
10: **else**
11: **if** $\mathcal{ML}^{\uparrow}_{add} \geq \mathcal{ML}^{\downarrow}_{add}$ and $\mathcal{ML}^{\uparrow}_{add} > 0$ **then**
12: $G \leftarrow (V, E \cup \{(n_p, n_t)\})$
13: **else**
14: **if** $\mathcal{ML}^{\downarrow}_{add} > 0$ **then**
15: $G \leftarrow (V, E \cup \{(n_u, n_v)\})$
16: **end if**
17: **end if**
18: **end if**
19: $\beta \leftarrow \beta - 1$
20: **while** $\beta > 0$ and ($\mathcal{ML}^{\uparrow}_{add} > 0$ or $\mathcal{ML}^{\downarrow}_{add} > 0$ or $\mathcal{ML}_{del} > 0$)

they are selected based on their respective communities' volume.

Secondly, lines 5-7 compute the modularity loss of the candidate edges based on Eqs. (10) and (28). Finally, the remaining part of the algorithm performs the update, yielding the highest modularity loss. Theorem 9 gives a detailed complexity analysis of IDEC:

Theorem 9 *Given a community structure $\bar{C}$ with k communities, and an edge budget β, algorithm 1 runs in $\mathcal{O}\big(|E| + \beta(|E_{\mathcal{C}}|+k)\big)$, where $E_{\mathcal{C}}$ is the number intra-community edges in $\mathcal{C}$.*

Table 3 A summary of most profitable edge and node operations

Level	Type	Operation	Relevant theorem	Algorithm
Edge	Intra-community	deletion	Theorem 2	IDEC & INDEC
	Inter-community	addition	Theorem 5	
Node	member of $\mathcal{C}$	deletion	Theorem 6	INDEC
	a new node $u \notin V$	addition	Theorem 7	
	member of $\mathcal{C}$	movement	Theorem 8	

Proof Notice that in the initialization phase of algorithm 1, we precompute $\mathcal{I}$ (total influence of the network), η (total influence only considering the intra-community edges), and θ (total influence only considering the inter-community edges. Moreover, to select inter-community edges (lines 3 and 4), we construct two lists, each containing the volume of each community considering *incoming* (resp. *outgoing*) edges. Thus, the initialization phase requires $\mathcal{O}(|E|)$ to compute $\mathcal{I}$, η, and θ and $\mathcal{O}(|V|)$ to compute the volumes.

Those values and data structures are computed only once and updated after each edge operation, which can be performed in constant time.

Choosing the best intra-community edge for deletion (line 2) needs to choose the edge with the biggest influence in $\mathcal{C}$, and this runs in $\mathcal{O}(|E_{\mathcal{C}}|)$, which is needed to loop over intra-community. Alternatively, one can consider a data structure like a heap and access the best edge in constant time. Notice here that computing modularity loss can be done in constant time after deleting the selected edge.

Next, we move to the procedure for selecting each new inter-community edge (lines 3 and 4); finding the maximum in the lists keeping *in* and *out* volumes requires $\mathcal{O}(k)$ where k is the number of communities; even in this case, one can use a priority queue or a heap. Overall, performing β edge updates requires $\mathcal{O}\big(|E| + \beta(|E_{\mathcal{C}}| + k)\big)$ □

6.3 Node-based deception via INDEC

We present a new greedy algorithm called INDEC, incorporating all three node operations presented so far. The primary purpose of this algorithm is to show how different node operations can work in tandem, augmenting edge operations as well. Algorithm 2 works initially by selecting three candidate nodes: a node to delete (line 2), to add (line 3), and to move (line 4). Along with each selected node, the corresponding function returns the edge set of the affected node, and the modularity loss (Δ^-,Δ^+, and Δ^m) that results if the operation is eventually performed. Besides each function, we indicate the Theorem used in computing the candidate node. Notice that we follow the node-adding heuristic described earlier in Sect. 5.2.2. Finally, the algorithm chooses the best operation based on the expected modularity loss, prioritizing node movement (line 5), which only requires modifying edges. If it's not beneficial (deception-wise), node addition is tested, and if also unsuccessful, node deletion is performed as a last resort.

Algorithm 2 The INDEC algorithm.

Input: Network G, target community $\mathcal{C} \subset V$, community structure $\bar{C}$.
Parameters: Edge budget β, and a set of addition candidates $\mathcal{N}$.
Output:Network $G' = (V', E')$

1: **do**
2: $(\mathbf{x}, E^-_{\mathbf{x}}, \Delta^-) \leftarrow$ selectBestDelNode($\mathcal{C}$)/*Thm. 6*/
3: $(\mathbf{y}, E^+_{\mathbf{y}}, \Delta^+) \leftarrow$ selectBestAddNode($\mathcal{N}$)/*Thm. 7*/
4: $(E^-_{\mathbf{z}}, E^+_{\mathbf{z}}, \Delta^m) \leftarrow$ selectBestMoveNode($\mathcal{C}, \bar{C}$)/*Thm. 8*/
5: **if** $\Delta^m > 0$ **then**
6: $G' \leftarrow (V, \{E \cup E^+_{\mathbf{z}}\} \backslash E^-_{\mathbf{z}})$
7: $\beta \leftarrow \beta - |E^+_{\mathbf{z}}| - |E^-_{\mathbf{z}}|$
8: **else**
9: **if** $\Delta^+ > \Delta^-$ and $\Delta^+ > 0$ **then**
10: $G' \leftarrow (V \cup \{\mathbf{y}\}, E \cup E^+_{\mathbf{y}})$
11: $\beta \leftarrow \beta - |E^+_{\mathbf{y}}|$
12: **else**
13: **if** $\Delta^- > 0$ **then**
14: $G' \leftarrow (V \backslash \{\mathbf{x}\}, E \backslash E_{\mathbf{x}})$
15: $\beta \leftarrow \beta - |E_{\mathbf{x}}|$
16: **end if**
17: **end if**
18: **end if**
19: **while** $\beta > 0$ and ($\Delta^+ > 0$ or $\Delta^- > 0$ or $\Delta^m > 0$)

The cost of this algorithm arises from lines (2-4) which find the candidate nodes and their associated data. In Theorem 10, we provide a detailed complexity analysis accounting for data initialization and each of the three functions.

Theorem 10 *Given a community structure $\bar{C}$ with k communities, and an edge budget β, algorithm 2 runs in $\mathcal{O}\big(|E| + \frac{\beta}{\gamma}(|V| + k)\big)$, where γ is the average edge operations per node operation.*

Proof Similar to algorithm 1, we note that several values and data structures are computed in the initialization phase. In particular, we precompute $\mathcal{I}$ (total influence in the network) and η (total intra-community influence in the network). Moreover, we construct a list of each community's mutual influence with the target community $\mathcal{C}$; which is needed to perform the node addition (line 3). Thus, the initialization phase of INDEC requires $\mathcal{O}(|E|)$ to compute $\mathcal{I}$ and η, and requires $\mathcal{O}(|V|)$ to compute the list of mutual influences. As

in our edge-based algorithm, we emphasize here that these values are precomputed once, and maintaining them after each node operation can be performed in constant time.

Now, we analyze the three operations (lines 2, 3, and 4). Notice that the procedure for node deletion in line 2 considers each member of $\mathcal{C}$ with complexity $\mathcal{O}(|V|)$; which occurs in a worst-case scenario $|\mathcal{C}| \approx |V|$. On the other hand, node addition (line 3) involves two steps described in section 5.2.2: choosing a destination community for the new node and adding its new edges. Both steps require $\mathcal{O}(k)$, thanks to the preconstructed list of inter-community influences constructed at the initialization phase, considering that adding new edges would have the same cost as that described in IDEC. Finally, the node movement procedure (line 4) considers each node in the target community, requiring $\mathcal{O}(|V|)$. Therefore, assuming each node operation adds/deletes γ edges, the overall complexity of INDEC is $\mathcal{O}\big(|E| + \frac{\beta}{\gamma}(|V| + k)\big)$, where β is edge budget. □

7 Experimental evaluation

In this section, we present a comprehensive experimental evaluation of both edge-based and node-based deception. The following experiments aim at illustrating the following points:

- *E1*: Compare our algorithms for deception in DIN with the state of the art (Sect. 7.4).
- *E2*: Measuring the impact of both edge and node-based deception on the community structure (Sect. 7.5).
- *E3*: Measure the efficiency of our algorithms in terms of execution time, and compare it with state-of-the-art competitors (Sect. 7.6).

7.1 Experimental setting

We performed the experiments in the following setting: given a network G, a community structure $\overline{C} = \{C_1, C_2, ...C_k\}$, found by some community detection algorithm $\mathcal{A}_{det}$, a target community $\mathcal{C}$ that we want to hide, and a budget of edge updates β, we apply a deception algorithm $\mathcal{A}_{dec}$ to obtain an updated network G'. Then, we ran the same $\mathcal{A}_{det}$ on G' to obtain another community structure $\overline{C'}$ where we can measure whether the level of hiding of $\mathcal{C}$ in $\overline{C'}$ is higher than in $\overline{C}$. Following are descriptions of detectors, deceptors, and datasets used in the experiments.

7.1.1 Detectors

We selected community detection algorithms that can be used on DIN, our targeted environment. Following is a brief description of each algorithm used. We utilized cdlib library[1] for implementing those detectors.

- Leiden Traag et al. (2019) (`leiden`) came as a response to a defect in the Louvain algorithm, which has shown a tendency to yield weakly connected communities.
- Directed modularity (Leicht and Newman 2008) (`dm`): a modularity maximizing algorithm that works with directed networks (Newman Jun 2004).
- Surprise community (`surprise`) An algorithm based on the notion of asymptotic surprise (Traag et al. 2015).
- Gemsec (Rozemberczki et al. 2019) (`gemsec`): an approach that leverages random walks to approximate the point-wise mutual information matrix obtained by pooling normalized adjacency matrix powers. This matrix is decomposed by an approximate factorization technique which is combined with a k-means-like clustering cost.
- Infomap (Rosvall and Bergstrom 2008) (`infomap`): Infomap is a network clustering algorithm based on the Map Equation (Rosvall and Bergstrom 2008). The idea is that networks have a flow that can be modeled via the Map Equation, which defines an information-theoretic encoding of the network, and is used to simplify the network structure regarding this flow.

7.1.2 Deceptors

We considered the following deceptors:

- Delete Internal Connect External (Waniek et al. 2018) (DICE): this is a simple deception heuristic that aims at minimizing modularity by deleting intra-community edges and adding inter-community edges.
- Modularity Minimization (Fionda and Pirrò 2018) (MOD): this approach improves on DICE by considering community degrees while selecting edges for addition/deletion.
- Safeness-based deception (Fionda and Pirrò 2018) (SAF): an approach for deception by maximizing community *safeness*. One strength of SAF is that it does not require complete knowledge of the community structure beforehand.
- Permanence-based deception (Mittal et al. 2021) (NEU): another approach based on maximizing *permanence loss*, which is computed over individual nodes.
- We considered the variants of SAF, MOD, and NEU for directed networks introduce in Fionda et al. (2022b) called DMOD, DSAF, and DPER, respectively.
- Random node/edge updates (RND): randomly selecting nodes/edges for deletion or addition.

[1] https://cdlib.readthedocs.io.

Table 4 Datasets and communities found by the detectors considered

Network	$\|V\|$	$\|E\|$	Number of communities				
			leiden	dm	surprise	infomap	gemsec
Freeman	~50	~500	5	5	7	6	5
Email	~1K	~25K	28	32	21	12	16
AnyBeat	~12K	~67K	129	81	143	156	112
WikiVote	~7K	~103K	30	34	43	51	49
Facebook-like	~900	~142K	6	5	6	7	5
Epinions	~75K	~508K	795	986	–	–	–
Slashdot	~77K	~905K	825	1115	–	–	–
SocialNet	~82K	~1 M	841	1120	–		–
Academia	~200K	~1.4M	89	93	–	–	–
GooglePlus	~211K	~1.5M	2105	–	–	–	–

7.2 Datasets

Since we focus on community deception in DIN, we used several directed datasets for experiments. These are of varying sizes and represent different domains to ensure the generality of results. As previously mentioned, we computed the influence between nodes as an edge weight using the approach described by Kumar et al. (2016). All datasets we used are available online[2].[3,4] Table 4 gives an overview of the networks considered. The table also reports, for each network, the number of communities found by the detectors considered. In some of the more extensive networks, the number of found communities has been omitted (and substituted by a **-**) because the detector couldn't complete community detection after a timeout of 3 h.

7.3 Evaluation methodology

To present a comprehensive experimental evaluation, we apply several metrics to compare the performance of our algorithms with state-of-the-art competitors. Each of the following measurements captures a different facet of the community deception performance:

- *Deception Score)* (H_s) (Fionda and Pirrò 2018): a numeric value in the range [0, 1] that quantifies *hiddenness* of the target community. Specifically, it measures several desiderata: reachability preservation, community spread, and community hiding. A higher DS corresponds with better deception. DS is expressed with the equation:

$$\left(1\text{-}\frac{|S(\mathcal{C})|\text{-}1}{|\mathcal{C}|\text{-}1}\right) \times \left(\frac{1}{2}(1\text{-}\max_{C_i \in \overline{\mathcal{C}}}\{\mathcal{R}(C_i, \mathcal{C})\}) + \frac{1}{2}(1\text{-}\frac{\sum_{C_i \cap \mathcal{C} \neq \emptyset} \mathcal{P}(C_i, \mathcal{C})}{|C_i \cap \mathcal{C} \neq \emptyset|})\right) \tag{40}$$

where $|S(\mathcal{C})|$ is the number of connected components in the subgraph induced by nodes in $\mathcal{C}$. $\mathcal{R}$ and $\mathcal{P}$ refer to the recall and precision of the detection algorithm, which is described in detail in Fionda and Pirrò (2018).

- *Normalized mutual information (NMI)*: another popular metric for measuring the deception quality is the normalized mutual information (NMI) (Danon et al. 2005). NMI quantifies the amount of standard information between two random variables, i.e., given two random variables, how much can we infer about one given the other? In the context of community deception, NMI measures how much a post-deception community structure reveals about the original one.
- *Delta Communities* (Δ): we also measure the difference between the number of discovered communities before and after deception.
- *Running time:* we compare the running time of our deception algorithms with other competitors, measuring only the time required for performing deception, excluding other common preprocessing tasks, such as initial community detection.

We performed the experiments on a server with a 6x 3.0 GHz (4 cores) CPU and 64 GB of RAM. The reported results are the average (95% confidence interval) of 10 runs. For all experiments, we set the budget β=0.6*$|E(\mathcal{C})|$; we noticed this was the best value balancing deception and detection. The code of the influence-based deceptors is available online[5]

[2] https://data4goodlab.github.io/dataset.html.

[3] https://snap.stanford.edu/.

[4] https://toreopsahl.com/datasets.

[5] https://tinyurl.com/DiNDec.

7.4 E1: deception score comparison

We begin our evaluation with the deception score. Figure 4 shows a deception score comparison of ten deceptors, including our novel IDEC and INDEC across five small networks. Similarly, Fig. 5 also depicts deception score performance but for much larger networks to give better insight into how our proposed methods would scale with higher network volumes compared to the state of the art. Note that, as for large networks, we could only obtain results against `leiden` and `dm`, since for other detectors, the execution time limit (3 h) has been exceeded. As described earlier, a higher deception score correlates with three desiderata: reachability, community spread, and community hiding.

First, to allow for a better appreciation of the deception scores shown in the Figures, we have to keep in mind that we start from a worst-case scenario, where the target community is completely revealed $\mathcal{C} \in \bar{C}$, which means that the deception score is initially 0. Thus, even less than 0.5 deception scores may represent a reasonable improvement in the goal of hiding the target community.

The results for smaller networks in Fig. 4 show that our proposed algorithms IDEC and INDEC outperform other competitors in most datasets and against different detection algorithms. However, their performance seems slightly lower against `leiden`, particularly on the Facebook dataset. This might be because, although we applied it to directed networks, `leiden` is initially designed for undirected (Traag et al. 2019). Figure 5 shows that in larger scale networks, IDEC and INDEC maintain their advantage over competitors, with more robust performance against `dm`.

We also observe a small but constant advantage of applying INDEC over IDEC. This can probably be attributed to the fact that INDEC gives more flexibility in terms of the possible operations—node movement, for instance, can select a specific node and move it toward a specific destination community.

Combining the results from both Figures reveals an interesting observation: IDEC and INDEC are more robust to the change in network size compared to other directed competitors: DSAF, DMOD, and DPER. Indeed, we can see that our proposed methods maintain relatively good performance over small and large networks. Other directed deceptors are more well-suited to higher-volume networks than smaller ones.

7.5 E2: Effects of deception on detection

The effects of deception algorithms on community detection are observed using two metrics we present shortly: normalized mutual information (NMI) and the change in the number of detected communities before and after deception occurs (referred to as Δ). Higher NMI values signify less change in the community structure (and hence to the number of communities.).

7.5.1 Normalized mutual information (NMI)

Now, we compare the effect of our novel algorithms with other state-of-the-art *directed* deceptors. First, we use NMI (described in Sect. 7.3) to measure the degree to which each of the competing deceptors affects the original community structure found by $\mathcal{A}_{det}$. Figure 6 shows NMI values (y-axis) plotted for different *detectors* (colored bars), using different networks (x-axis). Each subplot of Fig. 6 shows the results for one of the directed *deceptors*.

We can observe that all deceptors produce comparable NMI values, nearly in the range [0.7−0.9]. Meaning they can generally preserve much of the community structure after deception. Notably, however, the DPER algorithm seems to cause a more significant change in the communities, as reflected by the lower NMI. One of the lowest NMI values computed is DPER against `surprise`, using Email network. As Fig. 6 shows, the NMI value for this value is close to 0.5, indicating a substantial disruption in the community structure. This fact can be reinforced by the results of Δ in Fig. 8, which shows for the same combination (DPER, `surprise`, Email) significant negative change in the number of communities after deception is performed.

As for detection algorithms, NMI appears lowest for `leiden` across all deception algorithms shown in Fig. 6. Stated differently, `leiden` appears to be the most sensitive of the five detection algorithms to modifications by deceptors. On the other hand, community structures detected by `gemsec` seem the most robust to edge modifications.

Figure 7 also measures NMI values for much larger networks. The NMI measurements here are taken against two detection algorithms: `leiden` and `dm`. A noticeable trend we find here is that in most instances, `dm` has lower NMI than `leiden` and hence is more sensitive to deceptors. This trend was not found in the case of small networks (Fig. 6) showing that `dm` seems more affected by the change of the network scale.

7.5.2 Change in the number of communities

We proceed now to evaluate the change in the number of communities discovered by various detectors before and after deception (Δ). Generally, high values of Δ might correspond to more community spread, which in turn, positively affects the deception score. But we need to stress here the fact that the deception score measures other desiderata as well, meaning its correlation with Δ is not straightforward. Again, we show results for smaller and larger scale networks because as we have already seen in the last section, network size can have a significant effect on the performance of different competitors.

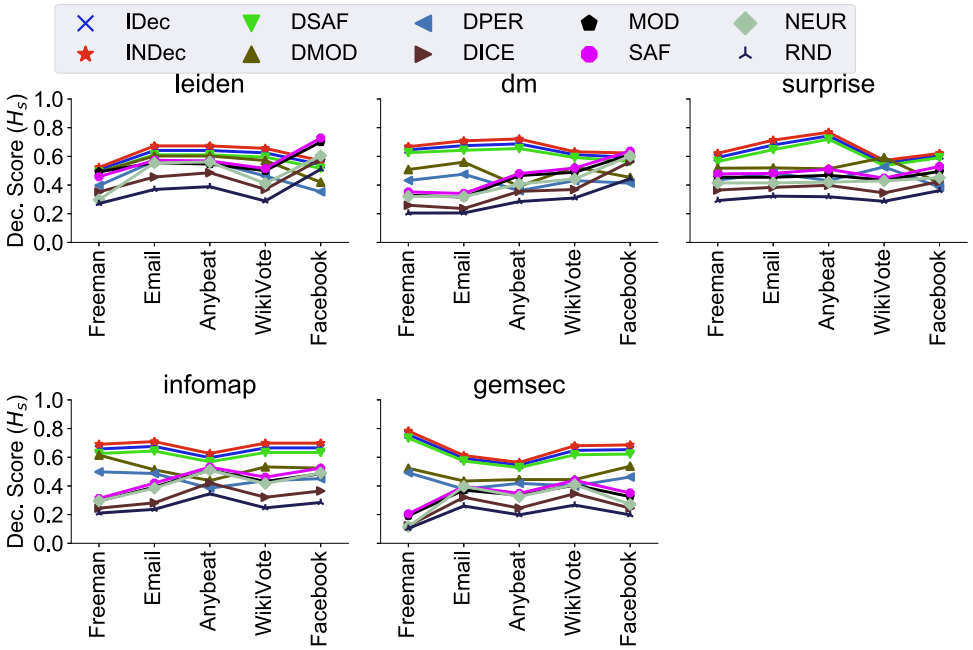

Fig. 4 Deception score comparison on small networks

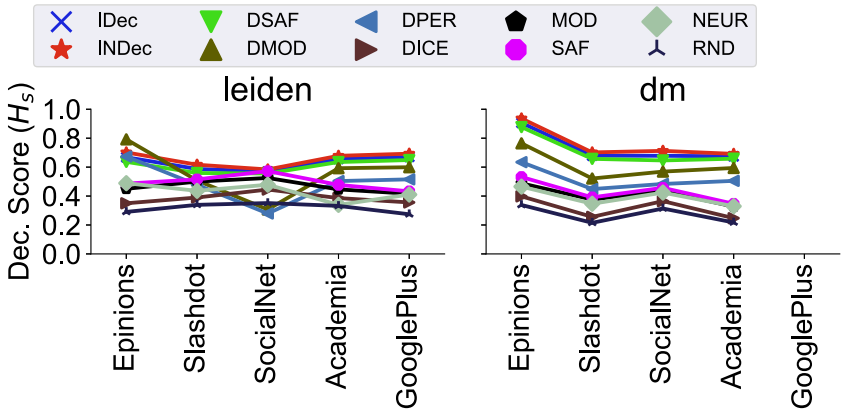

Fig. 5 Deception score comparison on large networks

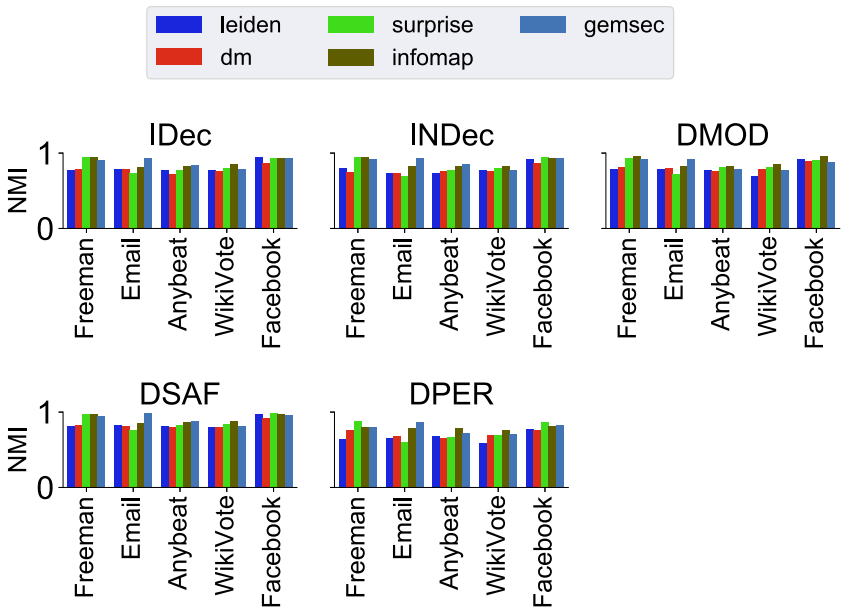

Fig. 6 NMI values comparing communities before and after deception on small networks

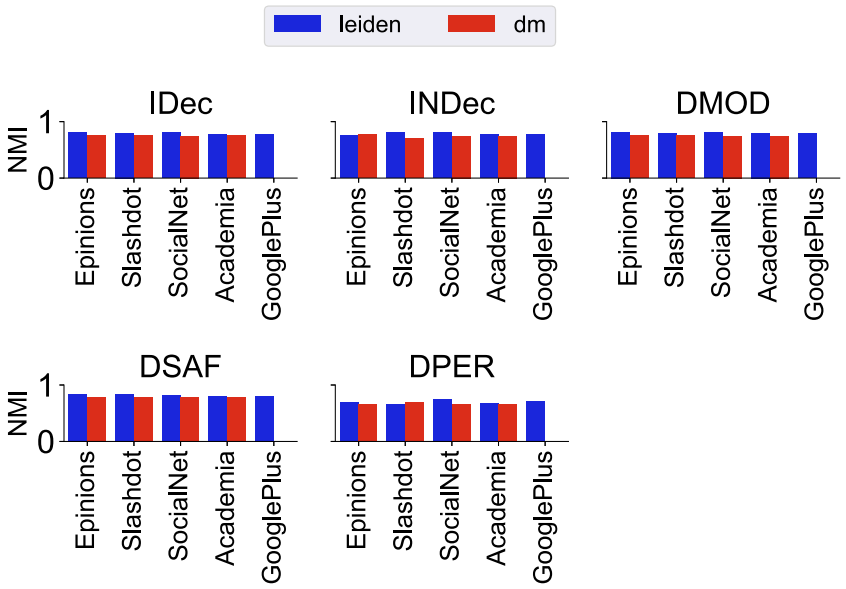

Fig. 7 NMI values comparing communities before and after deception on large networks

Figure 8 plots changes in the numbers of communities Δ on the vertical axis, against several small-size networks on the horizontal axis. Since both IDEC and INDEC accommodate directed influence for the first time, we considered several state-of-the-art deceptors (Fionda et al. 2022a) that work only with directed networks (although they ignore influence). In the vast majority of cases, both of our novel deceptors IDEC and INDEC resulted in an increase in the number of communities, and this agrees with their relatively higher deception scores shown previously in Fig. 4.

We can point to the variation between different detectors in terms of their robustness to deception algorithms. For instance, `surprise` shows high sensitivity to deception algorithms, and particularly in this case, we can see that both IDEC and INDEC perform comparably well by adding more communities after deception, compared to other deceptors. Furthermore, the noticeably higher magnitude of Δ using `Anybeat` network can be explained with the help of Table 4, which shows a significantly greater network volume and significantly more initial communities.

Next, we consider the group of larger networks in Fig. 9. The first observation that arises is that most deceptors have a negative effect on communities detected by `leiden`, and connecting this with the same results for small networks, we find that Δ seems to decrease as the network volume increases. Again, lower Δ for `leiden` explains its lower deception score on large networks shown in Fig. 5. On the contrary, compared to smaller networks, deceptors are now able to make higher positive changes against `dm`.

Combining Figs. 8 and 9 can give us an impression of the relation between Δ and network size. More specifically, the relative magnitude of Δ gets smaller for very large networks like SocialNet, Academia, and GooglePlus. Indeed, for the latter three networks, changing β edges seems unlikely to cause a significant disruption in the community structure in a network with a million-plus edge. A second observation is a deceptor like DPER cannot make noticeable changes in Δ, in most cases.

7.6 E3: Running time

Our final evaluation metric is running time. Figure 10 shows (on a log-scale) a comparison of running time for all ten deceptors using different small networks against each detection algorithm. The Figure shows that IDEC and INDEC perform comparably well to other *directed* deceptors and slightly slower than DSAF. This fact can be attributed to the minor cost of computing the safeness loss compared to modularity loss since safeness is computed only over the target community (Fionda and Pirrò 2018), not the entire community structure.

Additionally, it is possible to infer some correlation from Fig. 10 between the initial number of communities found by the detection algorithm and computation time. Specifically, for most deceptors, and against all tested detectors, the execution time for `Freeman` and `Facebook` is much lower than in other networks. Indeed, by referring to Table 4, we find that both networks are split into 5-7 communities by all considered detectors.

We can also observe an irregularly slow performance of MOD in Email network. One hypothesis to explain this behavior is that MOD, which partly relies on deleting intra-edges from the target community, consumes more time in finding a non-bridge edge for deletion, especially since it does not consider directions. Similar behavior of MOD in this partic-

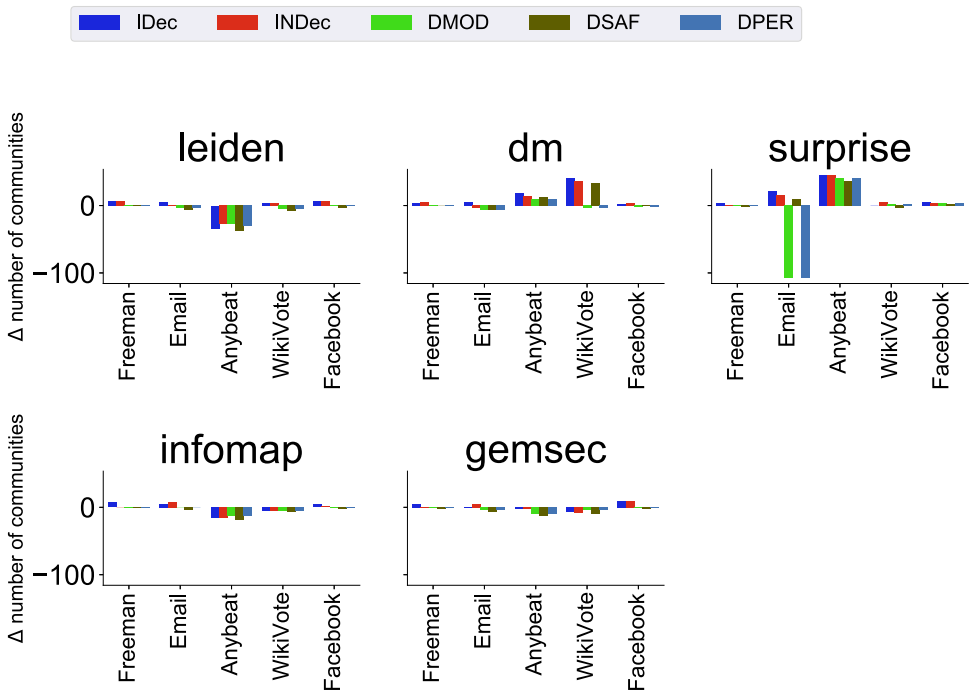

Fig. 8 Δ on the number of communities for small networks

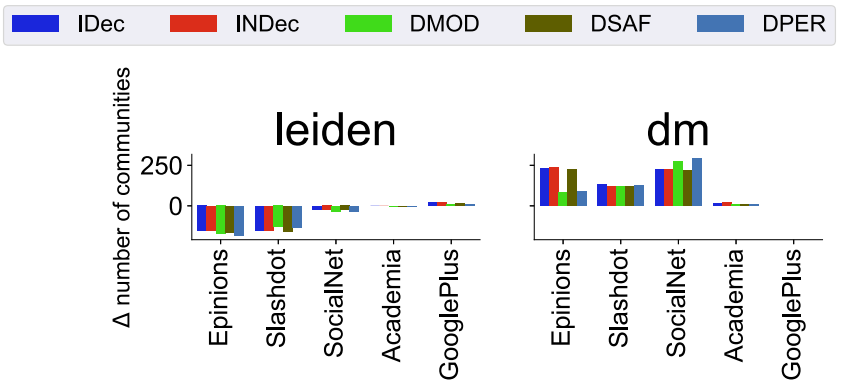

Fig. 9 Δ on the number of communities for large networks

ular network has been also mentioned by Fionda and Pirrò (2018); Fionda et al. (2022b).

We also consider time efficiency on much larger networks, as shown in Fig. 11. Notice that we consider only two detectors since other detectors could not produce a community structure within the set time limit (see section 7.2). As in the case of small networks, IDEC, and INDEC perform faster than other directed competitors, except DSAF which, as we indicated earlier, uses community-centric computation. Interestingly, over all networks, INDEC is faster than IDEC; this is more evident in Fig. 11 with extra large networks. This is probably an advantage of the node-based approach taken by INDEC. Specifically, each node operation INDEC modifies a group of edges at once, subsequently consuming the edge budget faster than IDEC. This result also conforms with the complexity analysis outlined in Sect. 6.

We can also observe a trend of generally increasing execution time as the network volume increases going from Epinions to GooglePlus, which is expected given the increase in edge and node count as shown in Table 4.

8 Concluding remarks and future work

Community detection algorithms are used to identify groups of nodes or individuals in a network that share common features or characteristics. These algorithms are widely used in various fields, including social network analysis, biology, and computer science. However, using these tools can harm users' privacy in different ways, from revealing group membership to exposing personal behaviors. This paper introduced the novel problem of influence-based community deception, which is about hiding a target community $\mathcal{C}$ from community detection algorithms in a setting where edges carry information about the mutual influence of nodes. We developed a twofold strategy of introducing directed influence and considering the role of nodes in the deception process. Our

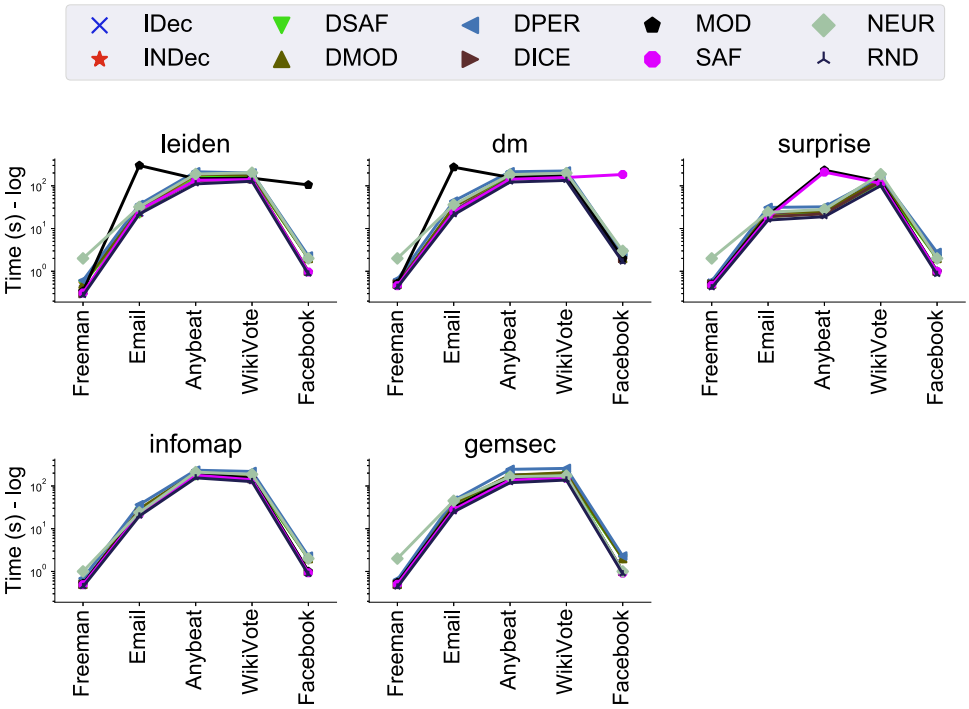

Fig. 10 Running time (log-scale) for small networks

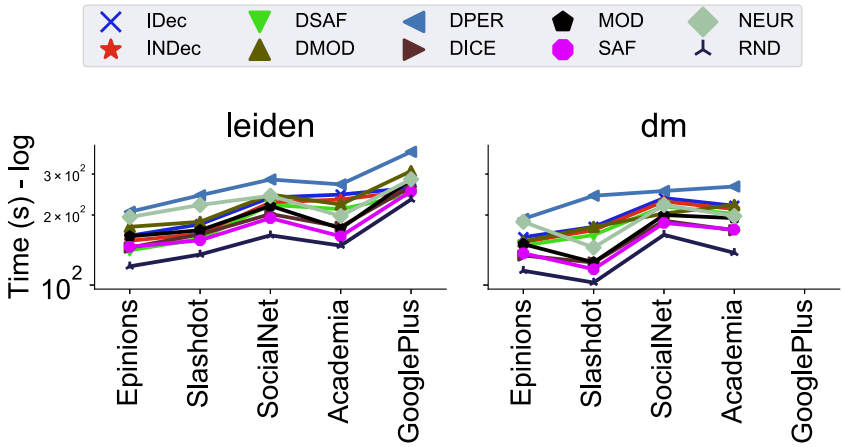

Fig. 11 Running time (log-scale) for large networks

study is more comprehensive than the state of the art since it includes aspects related to modifying the topology of a network by adding or removing edges and using adversarial node operations that, if carefully used, can disrupt the network's community structure, making it harder for detection algorithms to the target community. We believe the present paper provides valuable insights into community deception, and the proposed methods can be used in various applications, such as social network analysis and cybersecurity. We showed that while community detection algorithms are valuable tools for identifying communities in networks, they are not immune to countermeasures.

This paper has highlighted a vulnerability of community detection algorithms to community deception. While we considered deception as a tool to preserve user privacy, we cannot ignore the possibility of misusing them to hide unlawful groups. Therefore, as a future direction, we plan to investigate possible defense techniques against community deception, such as designing detection algorithms based on more robust quality functions. Indeed, as we showed in the experiments, some detectors were more sensitive to deception than others, this opens interesting research questions on the specific characteristics that make them so amenable to attacks. Moreover, another line of future research is to tackle deception when considering approaches that compute communities based on embeddings. Here, two interesting challenges are how to measure the contribution of edge modifications toward deception and how to define an effective deception optimization function.

Author Contributions SAM and GP wrote the main manuscript text, prepared the experiments and conducted the experiments. All authors reviewed the manuscript.

Funding Open access funding provided by Università della Calabria within the CRUI-CARE Agreement.

Declarations

Conflict of interest The authors declare no competing interests

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.

References

Chen J, Chen L, Chen Y, Zhao M, Yu S, Xuan Q, Yang X (2019) Ga-based q-attack on community detection. IEEE Trans Comput Soc Syst 6(3):491–503

Chen J, Chen Y, Chen L, Zhao M, Xuan Q (2020) Multiscale evolutionary perturbation attack on community detection. IEEE Trans Comput Soc Syst 8(1):62–75

Chen W, Yuan Y, Zhang L (2010) Scalable influence maximization in social networks under the linear threshold model. In: 2010 IEEE international conference on data mining, pp 88–97. IEEE

Chen X, Jiang Z, Li H, Ma J, Philip SY (2021) Community hiding by link perturbation in social networks. IEEE Trans Comput Soc Syst 8(3):704–715

Cheng J, Fu AW-C, Liu J (2010) K-isomorphism: privacy preserving network publication against structural attacks. In: SIGMOD, pp 459–470

Danon L, Diaz-Guilera A, Duch J, Arenas A (2005) Comparing community structure identification. J Statist Mech Theory Exp 9:2005

Fionda V, Madi SA, Pirrò G (2022) Community deception: from undirected to directed networks. Soc Netw Anal Min 12(1):74

Fionda V, Madi SA, Pirrò G (2022) Community deception: from undirected to directed networks. Soc Netw Anal Min 12(1):1–24

Fionda V, Pirrò G (2018) Community deception or: how to stop fearing community detection algorithms. IEEE Trans Knowl Data Eng 30(4):660–673

Fortunato S (2010) Community detection in graphs. Phys Rep 486(3):75–174

Fortunato S, Newman ME (2022) 20 years of network community detection. Nature Phys 18(8):848–850

Ghosh R, Lerman K (2008) Community detection using a measure of global influence. In: Proceedings of the second international conference on advances in social network mining and analysis. SNAKDD'08, pp 20-35, Berlin, Heidelberg. Springer-Verlag

Girvan M, Newman ME (2002) Community structure in social and biological networks. PNAS 99(12):7821–7826

Gubanov DA, Novikov DA, Chkhartishvili AG (2009) Models of influence in social networks. Upravlenie bol'shimi sistemami 27:205–281

Kumar S, Spezzano F, Subrahmanian V, Faloutsos C (2016) Edge weight prediction in weighted signed networks. In: 2016 IEEE 16th international conference on data mining (ICDM)

Leicht EA, Newman ME (2008) Community structure in directed networks. Phys Rev Lett 100(11):118703

Liu D, Chang Z, Yang G, Chen E (2022) Hiding ourselves from community detection through genetic algorithms. Inf Sci 614:123–137

Liu X, Fu L, Wang X, Hopcroft JE (2021) Prohico: A probabilistic framework to hide communities in large networks. In: IEEE INFOCOM 2021-IEEE conference on computer communications, pp 1–10. IEEE

Liu Y, Liu J, Zhang Z, Zhu L, Li A (2019) Rem: from structural entropy to community structure deception. Adv Neural Inf Process Syst 32:12938–12948

Lu Z, Zhu Y, Li W, Wu W, Cheng X (2014) Influence-based community partition for social networks. Comput Soc Netw 1(1):1

Ma T, Liu Q, Cao J, Tian Y, Al-Dhelaan A, Al-Rodhaan M (2020) Lgiem: global and local node influence based community detection. Future Gener Comput Syst 105:533–546

Madi SA, Pirrò G (2023) Influence-based community deception. In: Mantegna RN, Rocha LM, Cherifi C, Micciche S (eds) Cherifi H. Complex Networks and Their Applications XI. pp. Springer International Publishing, Cham, pp 175–187

Magelinski T, Bartulovic M, Carley KM (2021) Measuring node contribution to community structure with modularity vitality. IEEE Trans Netw Sci Eng 8(1):707–723

Mittal S, Sengupta D, Chakraborty T (2021) Hide and seek: outwitting community detection algorithms. IEEE Trans Comput Soc Syst 8(4):799–808

Nagaraja S (2010) The impact of unlinkability on adversarial community detection: effects and Countermeasures. In: PETS, pp 253–272

Newman MEJ (2004) Fast algorithm for detecting community structure in networks. Phys Rev E 69:066133

Rezaeimehr F, Moradi P, Ahmadian S, Qader NN, Jalili M (2018) Tcars: time- and community-aware recommendation system. Future Gener Comput Syst 78:419–429

Rosvall M, Bergstrom CT (2008) Maps of random walks on complex networks reveal community structure. PNAS 105(4):1118–1123

Rozemberczki B, Davies R, Sarkar R, Sutton C (2019) Gemsec: Graph embedding with self clustering. In: Proceedings of the 2019 IEEE/ACM international conference on advances in social networks analysis and mining, pp 65–72

Sarma D, Alam W, Saha I, Alam MN, Alam MJ, Hossain S (2020) Bank fraud detection using community detection algorithm. In: 2020 Second international conference on inventive research in computing applications (ICIRCA), pp 642–646

Su X, Xue S, Liu F, Wu J, Yang J, Zhou C, Hu W, Paris C, Nepal S, Jin D, Sheng QZ, Yu PS (2022) A comprehensive survey on community detection with deep learning. IEEE Trans Neural Netw Learn Syst, 1–21. https://doi.org/10.1109/TNNLS.2021.3137396

Traag VA, Aldecoa R, Delvenne J-C (2015) Detecting communities using asymptotical surprise. Phys Rev E 92(2):022816

Traag VA, Waltman L, Van Eck NJ (2019) From louvain to leiden: guaranteeing well-connected communities. Sci Rep 9(1):1–12

Wang C, Chen W, Wang Y (2012) Scalable influence maximization for independent cascade model in large-scale social networks. Data Min Knowl Discov 25:545–576

Wang W, Street WN (2014) A novel algorithm for community detection and influence ranking in social networks. In: 2014 IEEE/ACM international conference on advances in social networks analysis and mining (ASONAM 2014), pp 555–560

Waniek M, Michalak TP, Wooldridge MJ, Rahwan T (2018) Hiding individuals and communities in a social network. Nat Human Behav 2(2):139–147

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.