A multiagent network for peer norm enforcement

**Adrian Perreau de Pinninck · Carles Sierra ·
Marco Schorlemmer**

Abstract   In a multiagent system where norms are used to regulate the actions agents ought to execute, some agents may decide not to abide by the norms if this can benefit them. Norm enforcement mechanisms are designed to counteract these benefits and thus the motives for not abiding by the norms. In this work we propose a distributed mechanism through which agents in the multiagent system that do not abide by the norms can be ostracised by their peers. An ostracised agent cannot interact anymore and looses all benefits from future interactions. We describe a model for multiagent systems structured as networks of agents, and a behavioural model for the agents in such systems. Furthermore, we provide analytical results which show that there exists an upper bound to the number of potential norm violations when all the agents exhibit certain behaviours. We also provide experimental results showing that both stricter enforcement behaviours and larger percentage of agents exhibiting these behaviours reduce the number of norm violations, and that the network topology influences the number of norm violations. These experiments have been executed under varying scenarios with different values for the number of agents, percentage of enforcers, percentage of violators, network topology, and agent behaviours. Finally, we give examples of applications where the enforcement techniques we provide could be used.

Keywords   Multiagent systems · Norms · Enforcement · Social network · Ostracism

A. Perreau de Pinninck (✉) · C. Sierra · M. Schorlemmer
IIIA, Artificial Intelligence Research Institute, CSIC, Spanish National Research Council,
Bellaterra, Barcelona, Spain
e-mail: adrianp@iiia.csic.es

C. Sierra
e-mail: sierra@iiia.csic.es

M. Schorlemmer
e-mail: marco@iiia.csic.es

1 Introduction

Multiagent systems (MAS) are formed by a group of agents that interact with one another with the purpose of achieving their personal goals. In order for multiagent systems to be viable, the agents involved should have a better chance of achieving their goals by interacting with others in the MAS than by trying to achieve them on their own. When interacting with other autonomous entities, which are potentially selfish, planning can be a complex task. For situations where planning is cumbersome, norms may be established that restrict the set of valid actions, thus simplifying the planning process. However, an autonomous agent can choose whether to follow the norms or not.

It is mainly in the interest of the designer of norms in multiagent systems that agents abide by them. In such sense, we define a table showing the coarse gain achieved by the designer (or by the community as a whole) for interactions among pairs of agents of the system, which are define as joint actions. Agents have two main choices: to abide by the norm or to violate it. The most desired interactions are those in which both agents abide, this brings about a positive outcome. On the other hand, interactions where any agent violates the norm are not desirable, being those interactions where both agents violate the norm the most undesirable of all (see Fig. 1). Nonetheless, some individual agents may get more satisfaction out of executing illegal actions themselves, otherwise there would be no need for enforcement. The situation where norms are beneficial when all agents abide, but where agents also have an incentive to break the norm is the one where the normative behaviour is hardest to achieve.

The purpose of this work is to find behavioural properties of the agents that are better suited to achieve norm compliance at the global level. We examine those behavioural properties that can serve as distributed norm enforcement techniques in the MAS we study. We do not aim to study the internals of agents in order to determine what might motivate them. We treat agents as black boxes whose internals we do not have access and we study the outcome in norm abidance when different agents with different behaviours interact.

Ostracism is the exclusion by general consent from common privilege or social acceptance.[1] Ostracism is a peer norm enforcement technique, i.e., a technique applied among equals to enforce norms. In the approach described in this work there is no co-ordinated action by the community to ostracise, it is a gradual process by which a violator agent is removed through the actions of its peers. Furthermore, there is no explicit expiration time for ostracism, it is individual agents who choose whether to readmit the violator or not. The inspiration for our approach comes from the network security area, where firewalls are the most commonly used tool to avoid undesirable interactions. Such firewalls are managed by technicians which set up the rules under which they operate. These rules define which communications they allow and which they block.

In our approach a MAS is structured as a network of agents where the links between agents define a neighbour relationship. Agents in this network can execute joint actions with each other. We take a strong stance on speech act theory, by which all agent actions are illocutions which can be encapsulated as messages. In order to execute a joint action an agent a will search for a path through the network leading to a partner b where all the path's intermediate agents have granted a access to the next step in the path. Once the path is found, agents a and b can execute a joint action. In our approach the set of all actions agents can execute

---

[1] Ostracism was first practised by the ancient Greeks as a method of temporary banishment by popular vote without trial or special accusation. The way ostracism was decided in Athens was by casting a vote in pieces of broken pottery called ostraka. If enough votes were cast, the person with the highest number of votes was forced into exile for ten years, after which he was allowed to return without loss of status. If he tried to return before that he would face a death sentence.

Fig. 1 Global outcomes
of interactions

must be defined, and the normative behaviour defines which actions are permitted depending on the environment. Having to search for a path through the network in order to execute a joint action makes agents depend on other agents. Such dependence allows agents to block access to the network to those agents which violate the norms by executing actions which are forbidden. A violator agent is effectively ostracised from the network when enough agents have blocked it.

The peer norm enforcement techniques introduced in this paper could be used in applications where the norms are well known to all the participants and whose compliance can be objectively tested. For instance, in a file sharing network where the file size cannot exceed 100 MB, or in a news sharing application where certain linguistic expressions are forbidden. In order to join the file or news sharing systems, an agent would have to create at least one link to one of the agents already in the system thereby creating a social network of information sharing agents. Given that the file sharing or information sharing networks have a set of norms about the information that can be sent, sending information which does not fulfil these norms would be forbidden. By using the enforcement techniques proposed in the current article, those agents which violate the norm repeatedly would eventually be ostracised and could not continue harming the rest of the agents.

We have designed a totally distributed system that allows norm enforcement. Therefore, there are only two ways in which an agent can find out whether another agent is a norm violator: by being the partner in a joint action where the other agent executed a forbidden action, or by having the contents of a joint action, where a forbidden action was executed, being disclosed to it. In the MAS model we propose, we use encryption techniques that guarantee that the disclosure of the joint action contents can only be verified as truthful by the agents in the joint action path. This restriction removes any incentive to disclose non-existing joint actions, or to disclose to agents not in the joint action path.

The process through which an agent is ostracised is shown in Fig. 2. At first a norm violator (dark grey node) is believed to be an abiding member of the MAS, therefore the other agents will execute joint actions with it (the light grey nodes can execute joint actions with the violator). At some point the norm violator will execute a forbidden action and the partner of this joint action will realise this. Furthermore, the intermediate agents in the path between them will also know if the partner discloses the joint action contents. If agents knowing about this forbidden behaviour block the norm violator (black nodes are blocking the violator), its access to other agents in the network will be restricted (i.e., white nodes cannot execute joint actions with the violator). When all its neighbours find out about its forbidden behaviour and block it, the norm violator is effectively ostracised.
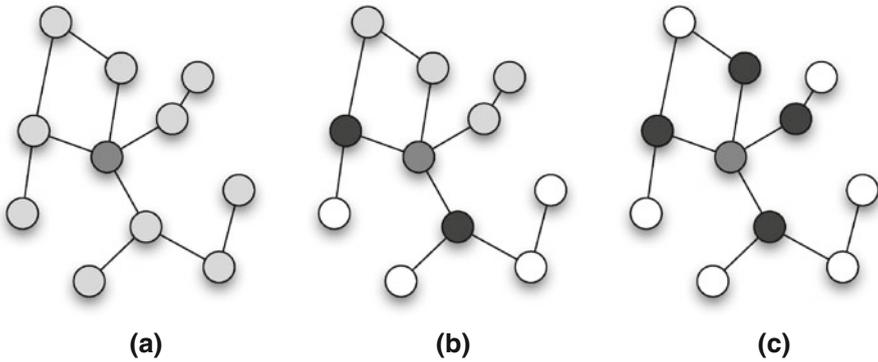
Fig. 2 Ostracising a violator. **a** Unrestricted violator; **b** Semi-restricted violator; **c** Ostracized violator

When designing the multiagent system, our main concern was how effectively norm violators could be stopped, and what behavioural properties agents had to exhibit in order to speed up the ostracism process. Some work on this was already published by us in [12,13]. On those papers we defined an initial model, which we have improved in this article and from which we have extracted some analytical results. Furthermore, on our previous work we did some exploratory analysis that guided us towards formulating certain properties of the system that would account for a reduction on the number of norm violations. In this work we have generalised the model used in the above mentioned previous work by allowing more interactions than just those that can be modelled through game theory, thus, we have been able to concentrate on studying how the behavioural properties of the agents affect the abidance to the norms at the global level without dealing with the agent motivations. We have proven analytically that when all agents in the network exhibit certain behavioural properties, there exists an upper bound to the total number of illegal actions that can be executed. Nonetheless, since agents are autonomous, we cannot ensure that they will all exhibit such behavioural properties. In those cases where there is a subset of agents that apply the norms, we have shown that the number of norm violations executed against them also has an upper bound under certain conditions. Notwithstanding, there are still cases in which either these condition do not hold or we want to continue to quantify norm violations to all agents. For these cases, we have run several experiments to support our claim that when agents exhibit such behavioural properties, the number of illegal actions that are executed is reduced. Furthermore, one of the analytical proofs supports our intuition that the network topology has an impact on the ostracism efficiency. The proof shows that the network size, i.e., the total number of links in the graph, is the upper bound of illegal actions under certain conditions. We have also run experiments whose results support this hypothesis.

The remaining of the paper is structured as follows. Section 2 reviews related work in the area of norm enforcement. Section 3 describes the multiagent system model and Sect. 4 defines the agent behaviour model, some properties it can exhibit, and shows analytically how they influence norm enforcement. Section 5 presents a detailed description of the scenario employed in the experiments. Section 6 gives an account of the simulations, and analyses the resulting data. Section 7 provides some examples of how the model and techniques could be applied in real world applications. Finally, Sect. 8 presents a discussion on the paper's results and future work that follows from this research.

2 Related work

Norms have been studied as a means of co-ordinating actions and granting access to limited resources for a long time in human societies through the study of law, philosophy and the social sciences [17,35]. Research in norms to co-ordinate artificial systems has been greatly influenced from those areas of knowledge [8]. Furthermore, enforcement techniques in human societies have also served as a basis for some of the research in the artificial. Not all the techniques used on humans can be used on artificial agents, since the latter do not feel pain, embarrassment, and usually do not care for money. Nonetheless, artificial agents in multiagent systems are similar to humans in that they are somewhat gregarious since they need other agents to achieve their goals.

Norm enforcement in multiagent systems is done under one of two premises: (a) the designer can control the actions agents realise in the system and can stop forbidden actions before they take place, or (b) no one except the agent can control its actions, and enforcement must be executed after actions have taken place through the use of sanctions or rewards. Systems such as S-$MOISE^+$ [21] or Ameli [16] are designed with the first premise in mind. All interactions between agents are mediated by some trusted component implemented by the system designer which verifies that actions are permitted. Other systems use model checking techniques to verify that the agent code will fulfil all of the system's norms [1,26]. Our approach deals with the second premise.

As far as we know, when using sanctions and rewards as incentives for permitted actions, approaches can be categorised through two criteria [41]. The first criterion defines who is in charge of applying the sanctions, i.e., self-enforcement versus third party enforcement. Self-enforcement means that it is the victim of a norm violation that will execute the sanction. Third party enforcement allows agents not directly involved in the norm violation to apply sanctions. The second criterion to classify enforcement methods depends on whether the agents can avoid interactions or not with other agents. When interactions cannot be avoided, a future interaction is needed to execute the sanction against the norm violator. When interactions can be avoided the sanction to norm violators consists on not interacting with them in the future so they cannot enjoy the benefits of interaction. In such cases, the problem is shifted to knowing with whom to interact. This problem is also known as boundary maintenance.

The first research on enforcement via agent simulations [3] sought ways to ensure co-operation in situations where agents had high incentives to avoid co-operation. A utilitarian approach was taken in that research by modelling interactions amongst agents through an iterated prisoner's dilemma (IPD). In that approach interactions could not be avoided and enforcement was reciprocated through self-enforcement. Therefore, an agent that did not co-operate did not receive reciprocated co-operation in the future. This was termed *the shadow of the future*. The problem with limiting the interaction in an IPD, where interactions cannot be avoided, is that an agent is forced to stop co-operating in order to sanction a noncooperator. This can end up in a spiral of noncooperation. If the norm is to co-operate, the norm must be broken in order to sanction. Later research solved this problem by modifying the original game of prisoner's dilemma and adding an enforcement stage where agents could decide to sanction those that did not co-operate [4]. The utilitarian point of view to enforcement showed that when norm-violators are punished, violating the norm was not the utility maximising strategy [9]. Nonetheless, if applying sanctions has a cost of its own, then agents may not want to sanction others, which brings about a free-riding problem [7]. In such cases, adding a norm saying that agents ought to enforce norms does not solve the problem. It is just shifted up a level since agents may not enforce the enforcement norm, bringing about an infinite regression.

Mechanism design also studies how to get agents to act in the way that the designer of the system wants them to. Through mechanism design the rules of the games are designed so that a specific outcome can be achieved. Incentives are provided, from a game theoretic point of view, for utility maximising agents to present specific properties when interacting with others. The properties that are commonly sought for are: truthfulness, budget balance, and social welfare. In order to really understand mechanism design one has to get technical. The interested reader is referred to the excellent introductions in [22,25,29], and the most recent survey on mechanism design for computer scientists in [27]. Our work somewhat looks for social welfare by trying to reduce the number of norm violations. Nonetheless, we do not assume rationality in the game theoretic sense and neither do we aim to get the agents to act a certain way through individual incentives. We study how the use of the norm enforcement techniques we provide influence the number of norm violations and leave for further work to create incentives so that the agents do use those techniques which are shown to be useful.

Many works (including ours) take a third party enforcement approach where interactions can be avoided. The main idea is that agents will only interact with those agents that are part of their group, and agents are considered part of the same group if they follow the same norms. In order to know what rules an agent follows, one must observe its actions or be told what they are. The main issue in these works is to calculate an agent's reputation as an in-group measure [9,19,20,43]. When an agent has to decide whether to interact with another, it must be sure that its model of the other agent is good enough, otherwise it could err in its decision. The methods that gather information about other agents must take the possibility of false feedback into account. An agent a can detect false feedback from agent b by comparing b's feedback about agents with which a has interacted with its own experience [40]. In our case we minimise the effect of lying by encrypting the communications between interacting agents and only allowing the joint action contents to be disclosed to the agents that allowed the interaction to take place. Any of the agents receiving the feedback could verify its validity by asking for a copy of the joint action, which is signed by both agents (see Sect. 4.4). We take a strong stance on speech act theory, in which all the actions performed by agents are speech acts, which are implemented as messages that can be encrypted. Moreover, since feedback is only sent to those in the path taken to interact, they can verify that an actual interaction took place. Furthermore, our approach does not leave the decision of whether or not to interact completely in the hands of the potential partners. In our approach two agents that want to interact must find a path in the network that allows this interaction. Since the agents in the network have a say in the decision to interact, the potential partners may have an imperfect model of each other, but they trust that the agents in the path between them may have a better model.

Norm enforcement is tightly linked to norm spread (sometimes termed norm emergence). Research in norm spreading is focused on finding the conditions under which norms spread in a multiagent system [38]. The two main approaches to spreading a norm are either evolutive or learning. In the evolutive approach agents with low success leave the system (i.e., die) and agents with high success are copied (i.e., reproduce) and some mutation is possible. In the learning approach agents that are less successful copy the behaviour of the agents that are more successful [32]. Both approaches look for the conditions which guarantee that the behaviours that survive are those that follow the norms. Some of this research deals with structured multiagent systems. In [23] the influence of simple structures such as regular graphs, trees, and hierarchies on the spread of norms is studied. Later work studied the influence of complex structures possessing free-scale or small-world properties that can be found

in many natural systems [11]. Finally, in [30] the relationship between norm emergence and other graph parameters such as its clustering factor and diameter is studied.

The model presented in this work assumes that the norm is predefined and cannot be changed by the agents in the MAS. All agents are assumed to know the norm. There has been much research on normative languages. Some languages are based on deontic logic [37] such as [6,33]. Other languages are based on process calculi through different mathematical formalisations such as Ambient Calculus [15], π-calculus [28], event calculus [24,42] or the lightweight co-ordination calculus (LCC) [31]. A more readable set of normative languages are defined through rules. Examples of these languages can be found in [10,18,34]. In our work we do not aim to use any of these languages for the norm definition. We assume that the norm designer provides a function that defines the permitted actions given the current environment.

When the interactions among agents are complex, and so are the norms defined to regulate them, then agents need sophisticated techniques for norm violation detection. Research on norm violation detection assumes that the content of interaction among agents can be scrutinised by organisational agents in order to detect when the interaction reaches illegal states [2,36]. In our approach, where interactions are simple joint actions, norm violation detection is trivial. By making the contents of the joint action available to the intermediate agents they can know when a norm violation has taken place.

3 The model

The model described in this section defines a special kind of multiagent system (MAS) which is structured as a network with fixed links. We will refer to these special MAS as *multiagent networks* (MAN). Agents in a MAN may execute joint actions with others, in these joint actions only a finite set of actions can be executed by each agent (see Definition 4). We take a strong stance on speech act theory, by which all agent actions are illocutions.

Throughout the formalisation below the following types of symbols will be used: Latin capital letters refer to sets (e.g., A). Lower case Latin letters refer to elements of sets (e.g., $a \in A$). Lower case Greek letters refer to functions (e.g., η). Finally teletype words are used to refer to concrete values (e.g., void), and bold words for predicates (e.g., **violator**). Furthermore, in all mathematical formula the variables are universally quantified unless specified otherwise.

The multiagent networks we define form a graph where the vertices are agents and the edges are direct communication channels between them. Two agents are neighbours if there is an edge between them. Furthermore, the model defines the set of actions that agents can execute, containing a special action (i.e., void) that means that the agent refuses to interact.

Definition 1 A multiagent network is a tuple $\mathcal{N} = \langle A, \eta, C \rangle$ where:

- A is a finite set of *agents*.
- $\eta : A \rightarrow 2^A$ is a *neighbourhood function* returning an agent's neighbours such that it is:
    - irreflexive, i.e., $\forall a \in A \ (a \notin \eta(a))$
    - undirected, i.e., $\forall a, a' \in A \ (a' \in \eta(a) \leftrightarrow a \in \eta(a'))$
- C is a finite set of *actions* that agents can potentially execute, with a distinguished element void.

In this work, to be neighbours means to be linked through a direct communication channel. Nonetheless, if two agents in the network are not neighbours, they may still interact through a path in the network.

Definition 2 Given a multiagent network $\mathcal{N} = \langle A, \eta, C \rangle$, a *path* is a finite sequence of agents $p = \langle a_1, a_2, \ldots, a_n \rangle$ where $a_i \in A$, such that:

1. its length is greater than one, i.e., $n > 1$;
2. any pair of consecutive agents are neighbours, i.e., $a_{i+1} \in \eta(a_i)$ for $i = 1, \ldots, n - 1$;
3. it contains no cycles, i.e., $i \neq j$ iff $a_i \neq a_j$ for $i, j = 1, \ldots, n$.

Agent a_1 is referred to as the *initiator* and a_n the *partner*. The other agents in the path are referred to as *mediators*. Let P be the set of all paths in network $\mathcal{N}$. Let $\mu : P \to 2^A$ be the mediator function. Given a path $p \in P$, $\mu(p)$ is the set of mediators in a path, i.e., $\mu(\langle a_1, a_2, \ldots, a_n \rangle) = \{a_i \mid 1 < i < n\}$.

A multiagent network defines how joint actions are executed and how agents observe them. During the execution of a MAN, two processes may occur: The joint action execution and the execution disclosure. The joint action execution process may be driven by the need of agents to act together. The disclosure process may be driven by the agents willing to make the contents of a joint action known in order to make norm violators identity known to others. This can either be motivated by revenge to the norm violator, or by altruism towards others that may encounter the same norm violating agent in the future. Nonetheless, we do not aim to study the motivations of agents in this work.

The joint action execution process is made up of the following stages:

1. A path is *constructed* that links an initiator and a partner.
2. The initiator and partner agents execute a *joint action* through the constructed path.

Agents may be able to execute many joint actions in parallel, but we assume that all events they perceive can be ordered. The events an agent can perceive are either the proposal of neighbours as potential partners, the execution of actions by a partner in a joint action, or the disclosure of the contents of a joint action.

Definition 3 Given a multiagent network $\mathcal{N} = \langle A, \eta, C \rangle$ a *partner proposal* is a tuple $\langle a, a', A' \rangle$ such that:

- $a \in A$ is the agent seeking for a partner agent that queries for neighbours.
- $a' \in A$ is the agent that proposes a set of its neighbours as potential partners.
- $A' \subseteq \eta(a')$ is the set of potential partners proposed by a' which must be a subset of its neighbours

Let F be the set of all partner proposals in $\mathcal{N}$.

Definition 4 Given a multiagent network $\mathcal{N} = \langle A, \eta, C \rangle$ a *joint action* is a tuple $\langle p, c, d, J \rangle$ such that:

- $p = \langle a_1, a_2, \ldots, a_n \rangle$ is a path in P.
- $c \in C$ is the initiator's action (i.e., a_1's action).
- $d \in C$ is the partner's action (i.e., a_n's action).
- J is the set of previously executed joint actions either by the initiator or partner, and that have been observed by both (see discussion below Definition 7 about the actions observed by the agents from the environment). Therefore, $\langle p, c, d, J \rangle \notin J$

Let G be the set of all joint actions in $\mathcal{N}$. Consequently, $J \subseteq G$. A joint action $\langle p, c, d, J \rangle$ is of *mutual consent* when $c \neq \texttt{void}$ and $d \neq \texttt{void}$.

Definition 5 Given a multiagent network $\mathcal{N} = \langle A, \eta, C \rangle$, a *disclosure* is a tuple $\langle a, \langle \langle a_1, a_2, \ldots, a_n \rangle, c, d, J \rangle \rangle$ such that:

- $a \in A$ is the agent disclosing the joint action.
- $\langle \langle a_1, a_2, \ldots, a_n \rangle, c, d, J \rangle \in G$ is the joint action being disclosed.

The disclosing agent must be either the initiator or partner of the joint action, i.e., $a = a_1$ or $a = a_n$. Let D be the set of all disclosures in $\mathcal{N}$.

A MAN has an associated environment containing the history of all events (i.e., partner proposals, joint actions, and disclosures). The history is the only part of the *environment* that we model in this work.[2] Agents have different perceptions of the environment, since they can only observe those events in which they are involved: being a seeker or a proposer in a partner proposal, being the initiator or partner of a joint action, or being a mediator of a disclosed joint action.

Definition 6 Given a multiagent network $\mathcal{N} = \langle A, \eta, C \rangle$, an environment is a tuple $e = \langle F', G', D' \rangle$ such that:

- $F' \subseteq F$ is a set of partner proposals.
- $G' \subseteq G$ is a set of joint actions.
- $D' \subseteq D$ is a set of disclosures.

Let E be the set of all environments for $\mathcal{N}$.

A global environment would contain all events that occurred during the system execution. Although the global environment might not be stored in any place it is a useful mathematical construct. On the other hand, there is a local environment that each agent can observe. These local environments are partial views of the global environment, thus, the global environment is the union of all the agents' local environments. Furthermore, an agent is said to have observed a joint action if it is part of its local environment.

Definition 7 Given a multiagent network $\mathcal{N} = \langle A, \eta, C \rangle$, a *global environment* $e = \langle F', G', D' \rangle$, and an agent a, the *local environment* of agent a is $e|_a = \langle F'', G'', D'' \rangle$, such that:

- $F'' = \{\langle a', a'', A' \rangle \in F' \mid a = a' \vee a = a'' \vee a \in A'\}$
- $G'' = \{\langle \langle a_1, a_2, \ldots, a_n \rangle, c, d, J \rangle \in G' \mid a = a_1 \vee a = a_n\}$
- $D'' = \{\langle a', \langle p, c, d, J \rangle \rangle \in D' \mid a \in \mu(p)\}$

Let $\theta : E \times A \to 2^G$ be the *observed joint action function*, where $\theta(e, a)$ is the set of all joint actions observed by agent a from the global environment $e = \langle F', G', D' \rangle$, i.e., given $e|_a = \langle F'', G'', D'' \rangle$ as defined above, $\theta(e, a) = G'' \cup \{g \in G' \mid \langle a', g \rangle \in D''\}$.

At the beginning of a MAN execution, the environment is always empty (i.e., $e_0 = \langle \emptyset, \emptyset, \emptyset \rangle$), thus, no agent has observed any joint actions (i.e., $\theta(e_0, a) = \emptyset$). Whenever a joint action is executed in an environment $e \in E$, the set J of previously executed joint actions is

---

[2] An environment, in general, could contain other pieces of information (e.g., sensor readings) that we do not consider.

the intersection of the joint actions observed from the environment by the interacting agents up to the moment of execution, i.e., $\forall \langle \langle a_1, a_2, \ldots, a_n \rangle, c, d, J \rangle \in G'$ $(J = \theta(e, a_1) \cap \theta(e, a_n))$.

The multiagent networks described in this work are normative. This means that an agent may be forbidden to execute some actions against another agent depending on the environment. In a MAN, the system designer defines a normative behaviour function that describes which actions an agent is permitted to execute in a joint action with another agent given the set of commonly observed joint actions. These commonly observed joint actions form part of the joint action so that any mediator can verify the partner agent's abidance to the normative behaviour if the joint action is disclosed.

Definition 8 A *normative behaviour* is defined as a function $v : 2^G \times A \times A \rightarrow 2^C$. Given the agents $a, a' \in A$, with the commonly observed joint actions $J \subseteq G$, $v(J, a, a')$ is the set of actions that a is permitted to execute in a joint action with a'. A joint action $g = \langle \langle a_1, a_2, \ldots, a_n \rangle, c, d, J \rangle$ is a *norm violation* when either $c \notin v(J, a_1, a_n)$ or $d \notin v(J, a_n, a_1)$. Our model allows agents to refuse to interact with norm violating agents. Therefore, it is always permitted to execute the void action against an agent that has executed a norm violation, i.e., $\texttt{void} \in v(J, a, a')$ if there exists $\langle \langle a'_1, a'_2, \ldots, a'_n \rangle, c', d', J' \rangle \in J$ such that either $c' \notin v(J', a'_1, a'_n) \wedge a' = a'_1$ or $d' \notin v(J', a'_n, a'_1) \wedge a' = a'_n$ hold.

4 Behavioural model

The previous section has introduced our model of a multiagent network. In this section we propose a behavioural model for the system, and we define some behavioural properties.

Executing joint actions and disclosure follow a specific algorithm in the current model which consists of three parts:

1. A path is *constructed* that links an initiator and a partner.
2. The initiator and partner agents execute a *joint action* through the constructed path.
3. Either the initiator or partner agent discloses the previously executed joint action.

4.1 Functional model

In the presented behavioural model we define functions describing the behaviour among agents. We assume deterministic agents, thus, we may define functions that describe the system's behaviour. We cannot learn these functions from observations, since the only way to define these functions is to have access to the internals of all agents in the system. Nonetheless, we may have approximations of these behaviours that allow us to see whether they satisfy specific properties (see Sect. 4.5).

Definition 9 Given a multiagent network $\mathcal{N} = \langle A, \eta, C \rangle$, its *behaviour* is a tuple $\langle \alpha, \pi, \delta \rangle$, where:

- $\pi : A \times A \times E \rightarrow F \cup \{\bot\}$ is a *potential partners* function that models the agents' mediation behaviour. Given an initiator agent $a \in A$, a mediator agent $a' \in A$, and an environment $e \in E$, $\pi(a, a', e)$ is either a partner proposal from a' to a or the empty event.
- $\alpha : A \times A \times E \rightarrow G \cup \{\bot\}$ is an *action execution* function that models the agents' action execution behaviour. Given an initiator agent $a \in A$, a partner agent $a' \in A$, and an environment $e \in E$, $\alpha(a, a', e)$ is either the joint action executed by a and a' in the environment e or the empty event.

– $\delta : A \times A \times E \rightarrow D \cup \{\perp\}$ is a *disclosure* function that models the agents' disclosure behaviour. Given an initiator agent $a \in A$, a partner agent $a' \in A$, and an environment $e \in E$, $\delta(a, a', e)$ is either the disclosure uttered by any of the two participants (see Sect. 4.4) or the empty event. We assume that agents do not disclose joint actions more than once, i.e., $\delta(a_1, a_n, \langle F', G', D' \rangle) \notin D'$.

Let B be the set of all system behaviours. The modelling of the system behaviour with functions is needed to prove the analytical results in Sect. 4.5. As a notation abuse, we define an agent behaviour as the system behaviour when one of the agent input variables is fixed to a specific agent.

4.2 Constructing a path

In the first stage of the joint action execution process an initiator selects a path that leads to a partner with which to interact. Not all paths in the network fulfil the properties needed in order to be part of a joint action because they depend on the the network environment and the system behaviour.

Definition 10 Given a multiagent network $\mathcal{N} = \langle A, \eta, C \rangle$, an environment $e \in E$, and a behaviour $\langle \alpha, \pi, \delta \rangle \in B$, a path $p = \langle a_1, a_2, \ldots, a_n \rangle$ is said to be *socially feasible* when each agent a_{i+1} in the path is proposed as a potential partner to the initiator by the previous agent in the path (i.e., a_i),i.e., $\forall 1 < i \leq n \ (\pi(a_1, a_i, e) = \langle a_1, a_i, A' \rangle \wedge a_{i+1} \in A')$.

During the path search process, the initiator agent will query agents for potential partners which are returned through partner proposal illocutions. The proposed partners must be a subset of the neighbours of the agent being queried. In order to construct the path, any graph search method may be used (see Sect. 5.2 for examples of search methods).

4.3 Executing a joint action

The second stage in the joint action execution process is to create the joint action. The joint action contains the feasible path that was constructed in the previous stage, the actions executed by the initiator and partner agents, and the set of joint actions observed by both of them from the environment. A joint action is executed because the initiator constructed a socially feasible path towards the partner. Nonetheless, both agents have the ability of executing the `void` action, which makes the joint action not of mutual consent.

By taking a strong stance on speech act theory, all actions are be executed through message passing. The messages containing the joint action are sent from one agent to another through the joint action path. In order for the selected actions to be private to the interacting agents the messages that contain the joint action are encrypted. This can be done by having a public key infrastructure (PKI) in which the public key of an agent is its identifier. Nonetheless, the use of a PKI involves some centralisation. This is why we choose to use a web-of-trust (WOT) approach implemented via OpenPGP through which we achieve the same results as using a PKI but without any centralisation. When executing a joint action, the message containing it would be encrypted using the recipient's public key. The encrypted messages would be passed along the joint action path, but none of the mediators should be able to decipher their content, thus keeping them private.

4.4 Disclosing joint actions

Disclosure is the process through which either the initiator or partner agents make the contents of a joint action observable to the mediator agents. In the previous section we have seen that the mediator agents have access to the encrypted message containing the joint action. This message has been encrypted using the public key of the destination agent, which is known to all. Therefore, if either the initiator or partner agent decide to disclose the contents of the executed joint action, they only need to send the decrypted contents to the mediators. The mediators can easily test whether the disclosed joint action contents are truthful by encrypting it using the public key of the recipient agent and verifying that the encrypted message matches the previous one that was sent through them. Since the original encrypted joint action is only known to the path mediators, only they can test its validity. Therefore, disclosure of joint actions is limited to the path's mediators. Furthermore, the mediators can test whether any of the actions was a norm violation by using the normative behaviour function ν.

The environment is updated as joint actions are executed and disclosed. In our MAN model the initiator and partner agents can disclose joint actions that have been previously executed by them only once. The environment is updated to include these illocutions whenever they are uttered.

4.5 Behavioural properties

In this section the properties of the proposed behavioural model are shown. They establish why a set of agents exhibiting certain behavioural properties enforce the norm by discouraging norm violations.

Definition 11 Given a multiagent network $\mathcal{N} = \langle A, \eta, C \rangle$ with a normative behaviour function ν, and given an environment $e \in E$, an agent a is a *norm violator* with respect to an agent a', noted as **violator**(a, a', e), if a executed a forbidden action in any of the joint actions observed by a'.

$$\textbf{violator}(a, a', e) \leftrightarrow \exists \langle \langle a_1, a_2, \ldots, a_n \rangle, c, d, J \rangle \in \theta(e, a')$$
$$((a_1 = a \wedge c \notin \nu(J, a_1, a_n)) \vee (a_n = a \wedge d \notin \nu(J, a_n, a_1)))$$

There are potentially many types of agent behaviours. We will discuss the properties of some of them: avoiding, blocking, protecting, and informing.

Definition 12 Given a multiagent network $\mathcal{N} = \langle A, \eta, C \rangle$, and a normative behaviour function ν, an action execution function α is said to be *violator avoiding* for an agent $a \in A$ when a executes the `void` action against norm violators, i.e., **violator**$(a', a, e) \wedge ((\alpha(a, a', e) = \langle p, \texttt{void}, d, J \rangle) \vee (\alpha(a', a, e) = \langle p', c', \texttt{void}, J' \rangle))$. Let the predicate **avoiding**(α, a) hold when the action execution function α is violator avoiding for agent a. A behaviour is said to be violator avoiding for agent a if its action execution function is.

For the following proofs we define the function $\tau : 2^G \times A \times A \rightarrow 2^G$. Given a set of joint actions $G' \subseteq G$ and two agents $a, a' \in A$, $\tau(G', a, a')$ is the set of the given joint actions that were executed by the given agents, i.e., $\tau(G', a_i, a_j) = \{\langle \langle a_1, a_2, \ldots, a_n \rangle, c, d, J \rangle \in G' \mid (a_1 = a_i \wedge a_n = a_j) \vee (a_1 = a_j \wedge a_n = a_i)\}$. We also define the function $\rho : 2^G \rightarrow 2^G$. Given a set of joint actions $G' \subseteq G$, and a normative behaviour function ν, $\rho(G')$ is the subset of the given joint actions which are norm violations and of mutual consent (see Definition 4), i.e., $\rho(G') = \{\langle \langle a_1, a_2, \ldots, a_n \rangle, c, d, J \rangle \in G' \mid (c \notin \nu(J, a_1, a_n) \vee d \notin \nu(J, a_n, a_1)) \wedge c \neq \texttt{void} \wedge d \neq \texttt{void}\}$.

Lemma 1 *Given a multiagent network* $\langle A, \eta, C \rangle$*, and a normative behaviour function* ν*, with an environment* $e \in E$*, let* $\langle \alpha, \pi, \delta \rangle \in B$ *be the system's behaviour. If the action execution function is violator avoiding for agents* a_i *and* a_j*, then the number of norm violations in mutually consented joint actions by agents* a_i *and* a_j *is at most 1, i.e.,* **avoiding**$(\alpha, a_i) \wedge$ **avoiding**$(\alpha, a_j) \rightarrow |\tau(\rho(\theta(e, a_i)), a_i, a_j)| \leq 1$.

Proof We proceed by reductio ad absurdum. Let us assume that **avoiding**$(\alpha, a_i) \wedge$ **avoiding**$(\alpha, a_j) \wedge |(\tau(\rho(\theta(e, a_i)), a_i, a_j)| > 1$. Let $g = \langle\langle a_1, a_2, \ldots, a_n \rangle, c, d, J \rangle$ be the joint action in $\tau(\rho(\theta(e, a_i)), a_i, a_j)$ executed the latest. This is known because by construction g contains all other joint actions in $\tau(\rho(\theta(e, a_i)), a_i, a_j)$ (see the discussion after Definition 7).

From the assumption we deduce that $\tau(\rho(\theta(e, a_i)), a_i, a_j) \backslash \{g\}$ cannot be empty. Since the action execution function is violator avoiding for agents a_i and a_j, and $\tau(\rho(\theta(e, a_i)), a_i, a_j) \backslash g \subseteq J$, thus containing at least one norm violation by agent a_i or a_j, then the joint action $\alpha(a_i, a_j, e')$, where e' is the environment at the moment of execution, should contain at least one `void` action. Therefore, g cannot be a joint action of mutual consent, i.e., $g \notin \rho(\theta(e, a_i))$, which is a contradiction. Hence **avoiding**$(\alpha, a_i) \wedge$ **avoiding**$(\alpha, a_j) \rightarrow |\tau(\rho(\theta(e, a_i)), a_i, a_j)| \leq 1$ as we wanted to prove. □

Theorem 1 *If the behaviour* $\langle \alpha, \pi, \delta \rangle$ *of a multiagent network* $\mathcal{N} = \langle A, \eta, C \rangle$*, with a normative behaviour function* ν*, contains an action execution function that is violator avoiding for all agents (i.e.,* $\forall a_i \in A$ *(***avoiding**$(\alpha, a_i))$*), then there exists an upper bound to the total number of potential norm violations in mutually consented joint actions that can be executed. This upper bound is* $|A|(|A| - 1)/2$.

Proof The total number of norm violations in a multiagent network is equal to the sum of the norm violations happening between each pair of agents. Therefore, one can calculate the total number of norm violations by adding the potential norm violations between each pair of agents.

$$\frac{\sum_{a_i, a_j \in A, i \neq j} \tau(\rho(\theta(e, a_i)), a_i, a_j)}{2}$$

Given that all agents in the system have a violator avoiding behaviour, Lemma 1 applies to all pairs of agents. Therefore,

$$\frac{\sum_{a_i, a_j \in A, i \neq j} \tau(\rho(\theta(e, a_i)), a_i, a_j)}{2} \leq \frac{|A|(|A| - 1)}{2}$$

which proves the theorem. □

The following result may be proved in much the same way as Lemma 1 and Theorem 1:

Corollary 1 *Given a multiagent network* $\langle A, \eta, C \rangle$ *with a normative behaviour function* ν*, and a behaviour* $\langle \alpha, \pi, \delta \rangle$ *that is violator avoiding for a group of agents* $A' \subseteq A$ *(i.e.,* $\forall a_i \in A'$ *(***avoiding**$(\alpha, a_i))$*), then there exists an upper bound to the total number of potential norm violations in mutually consented joint actions that can be executed by an agent when the other agent in the joint action is part of* A'*. This upper bound is* $|A'|(|A| - 1)/2$.

When the avoiding behavioural property is combined with other behavioural properties, the enforcement capabilities grow by reducing the number of potential norm violations.

Definition 13 Given a multiagent network $\langle A, \eta, C \rangle$ with a normative behaviour function ν, a potential partners function π is said to be *blocking* for agent $a \in A$ when the partner proposals it returns contain the empty set if the querier is a norm violator, i.e., **violator**$(a_j, a, e)) \wedge \pi(a_j, a, e) = \langle a_j, a, A' \rangle \rightarrow A' = \emptyset$. Let the predicate **blocking**(π, a) hold when the function π is blocking for agent a. A behaviour is said to be blocking for agent a if its potential partners function is.

It can be easily shown that when the system's behaviour is avoiding and blocking for all agents, a smart norm violator would execute forbidden actions against agents in the network whose neighbours are accessible through some other path. Therefore, the upper bound to the number of potential norm violations would remain the same as if all agents had an avoiding behavioural property alone. Consequently, in order to lower the upper bound we explore other behavioural properties.

Definition 14 Given a multiagent network $\langle A, \eta, C \rangle$ with a normative behaviour function ν, a potential partners function π is said to be *protecting* for agent $a \in A$ when the partner proposals it returns never contain norm violators, i.e., $\forall a_j \in \eta(a)$ (**violator**$(a_j, a, e)) \wedge \pi(a_k, a, e) = \langle a_k, a, A' \rangle \rightarrow a_j \notin A'$). Let the predicate **protecting**(π, a) hold when the function π is protecting for agent a. A behaviour is said to be protecting for agent a if its potential partners function is.

It is also straightforward to prove that when the system's behaviour is avoiding and protecting for all agents, a smart norm violator would execute forbidden actions against non-neighbouring agents first (e.g., by using a depth first search algorithm). Therefore, the number of potential norm violations shall remain the same as if all agents have an avoiding behavioural property alone. Even when all agents use a behaviour with avoiding, blocking, and protecting properties a depth first search for joint action partners to violate would maintain the upper bound. Therefore, other behavioural properties are needed to lower this upper bound.

Definition 15 Given a multiagent network $\langle A, \eta, C \rangle$ with a normative behaviour function ν, a disclosure function δ is said to be *informing* for agent $a \in A$ when it returns disclosures of norm violating joint actions executed against it, i.e., $\delta(a, a', \langle F', G', D' \rangle) = \langle a, g \rangle \rightarrow (g = \langle \langle a, a_2, \ldots, a' \rangle, c, d, J \rangle \wedge d \notin \nu(J, a', a)) \vee (g = \langle \langle a', a_2, \ldots, a \rangle, c, d, J \rangle \wedge c \notin \nu(J, a', a))$. Let the predicate **informing**(δ) hold when the function δ is informing. A behaviour is said to be informing for agent a if its disclosure function is.

Finally, it is obvious that if all agents in the system have a norm violation with avoiding and informing properties, a smart norm violator would only execute forbidden actions against agents in a breadth first manner (i.e., first through paths where it is a violator to all the mediators). In this way, the upper bound to the number potential norm violations remains the same as for a plain avoiding property.

Definition 16 A behaviour is full blocking for a given agent when it combines avoiding, blocking, protecting, and informing behavioural properties. Let the predicate **fullBlocking**$(\langle \alpha, \pi, \delta \rangle, a)$ hold when the behaviour $\langle \alpha, \pi, \delta \rangle$ is full blocking for agent a, i.e., **fullBlocking**$(\langle \alpha, \pi, \delta \rangle, a) \equiv$ **avoiding**$(\alpha, a) \wedge$ **blocking**$(\pi, a) \wedge$ **protecting**$(\pi, a) \wedge$ **informing**(δ, a).

For the following proofs we define the function $\tau : 2^G \times A \times A \rightarrow 2^G$. Given two agents $a_i, a_j \in A$ and a set of joint actions $G' \subseteq G$, $\tau(G', a_i, a_j)$ is the set of joint actions executed by the first agent through a path where the second agent appears beside it,

i.e., $\tau(G', a_i, a_j) = \{\langle\langle a_1, a_2, \ldots, a_{n-1}, a_n\rangle, c, d, J\rangle \in G' | (a_1 = a_i \wedge a_2 = a_j) \vee (a_n = a_i \wedge a_{n-1} = a_j)\}$. We also define the function $\rho : 2^G \times A \to 2^G$. Given an agent $a \in A$, a set of joint actions $G' \subseteq G$, and a normative behaviour function ν, $\rho(G', a)$ is the subset of the given joint actions in which the given agent selected the norm violating action in mutually consented joint actions, i.e., $\rho(G', a) = \{\langle\langle a_1, a_2, \ldots, a_n\rangle, c, d, J\rangle \in G' | ((c \notin \nu(J, a_1, a_n) \wedge a_1 = a) \vee (d \notin \nu(J, a_n, a_1) \wedge a_n = a)) \wedge c \neq \texttt{void} \wedge d \neq \texttt{void}\}$. Note that $\rho(\theta(e, a'), a)$ not being empty implies **violator**(a, a', e) but the opposite implication does not hold, since the **violator** predicate takes into account all joint actions with norm violations whereas ρ only takes into account those joint actions that are of mutual consent.

Lemma 2 *Let $\mathcal{N} = \langle A, \eta, C\rangle$ be a multiagent network with a normative behaviour function ν, a behaviour $b = \langle\alpha, \pi, \delta\rangle$, and an environment $e \in E$, where a_j and a_k are two neighbouring agents, i.e., $a_j \in \eta(a_k)$. If the behaviour is full blocking for each agent $a_i \in A$, i.e., $\forall a_i \in A$ (**fullBlocking**(b, a_i)), then the number of mutually consented joint actions executed by a_j in which it executed a norm violation through a path where a_k appears beside it, is at most 1, i.e., **fullBlocking**$(b, a_i) \to |\tau(\rho(\theta(e, a_j), a_j), a_j, a_k)| \leq 1$.*

Proof We proceed by reductio ad absurdum. Let us assume that $\forall a_i \in A$(**fullBlocking** $(b, a_i)) \wedge |(\tau(\rho(\theta(e, a_j), a_j), a_j, a_i)| > 1)$. Let $g = \langle\langle a_1, a_2, \ldots, a_{n-1}, a_n\rangle, c, d, J\rangle$ be the joint action in $\tau(\rho(\theta(e, a_j), a_j), a_j, a_i)$ executed de latest. This is known because by construction g contains all other joint actions in $\tau(\rho(\theta(e, a_j)), a_j, a_i)$ (see the discussion after Definition 7). From Definition 10, the paths of all executed joint actions must be feasible given the environment at the time of execution.

If $n = 2$ then we follow a similar reasoning as that of Lemma 1. Given that $\tau(\rho(\theta(e, a_j), a_j), a_j, a_i) \setminus \{g\}$ is not empty and α is avoiding for agent a_j, a_j's action would have been $\texttt{void}$ and the executed joint action not of mutual consent, which brings about a contradiction.

Otherwise, when $n > 2$ there are two options to consider: (i) paths of the form $\langle a_j, a_i, \ldots, a_n\rangle$ or (ii) paths of the form $\langle a_1, \ldots, a_i, a_j\rangle$. In both cases from the assumption it follows that $\tau(\rho(\theta(e, a_j), a_j), a_j, a_i) \setminus \{g\} \neq \emptyset$. Furthermore, since the disclosure function is informing for all agent in the network, then all agents in the path of a norm violation have observed the joint action, i.e., $\forall\langle\langle a_1, a_2, \ldots, a_n\rangle, c, d, J\rangle \in \rho(\theta(e, a_j), a_j), \forall k \in 1, \ldots, n$ ($\langle\langle a_1, a_2, \ldots, a_n\rangle, c, d, J\rangle \in \theta(e, a_k)$). Therefore, $\tau(\rho(\theta(e', a_k), a_j), a_j, a_i) \neq \emptyset$, where e' is the environment right before the execution of g.

Since the observed potential partners function is blocking and protecting for all agents, the path in the former option would not be feasible because $\pi(a_j, a_i, e')$ would contain an empty partner set, since a_j is a violator with respect to a_i's observed joint actions and π is blocking for agent a_i. In the latter option the path would not be feasible because $\pi(a_1, a_i, e')$ would not contain a_j since it is a violator with respect to a_i's observed joint actions and π is protecting for agent a_i.

All of the possible cases bring about a contradiction. Hence, $\forall a_i \in A$(**fullBlocking** $(b, a_i) \to |\tau(\rho(\theta(e, a_j), a_j), a_j, a_k)| \leq 1$ as we wanted to prove. $\square$

Theorem 2 *In a multiagent network $\mathcal{N} = \langle A, \eta, C\rangle$ with a normative behaviour function ν, and a behaviour $b = \langle\alpha, \pi, \delta\rangle$ which is full blocking for all agents (i.e., $\forall a_i \in A$ (**fullBlocking**(b, a_i))), there exists an upper bound to the total number of potential norm violations in mutually consented joint actions that can be executed. This upper bound is twice the number of links of the network, i.e., $\sum_{a_i \in A} |\eta(a_i)|$.*

Proof From Lemma 2 follows that if all agents have a full blocking behaviour, one single agent a is able to violate the norm at most once for each neighbour it has (i.e., $|\eta(a)|$).

Therefore, the total norm violations will be less than the sum of all agent neighbours (i.e., $\sum_{a \in A} |\eta(a)|$) which is twice the number of links in the network. □

In multiagent networks where the average neighbours per agent is less than half the population, it pays for agents to exhibit a full blocking behaviour, as opposed to a simpler avoiding behaviour, since less norm violations are possible. On the other hand, for densely connected networks an avoiding behaviour is sufficient. For such networks disclosing and storing all the norm violations is useless.

Interestingly, Theorem 2 implies that the structure of the multiagent network has an impact on norm enforcement. More densely connected networks are more prone to norm violations. On a single agent scale, agents with more neighbours can get away with more norm violations. Furthermore, with more neighbours comes a higher risk of becoming a victim of norm violations.

Nonetheless, as mentioned earlier in an open system it is highly probable that not all agents will exhibit the same behavioural properties. The following corollary generalises the results of full blocking behaviours when just a subset of agents exhibit them.

Corollary 2 *Given a multiagent network $\mathcal{N} = \langle A, \eta, C \rangle$ with a normative behaviour function v, a behaviour $b = \langle \alpha, \pi, \delta \rangle$, a subset of agents $A' \subseteq A$ that form a connected component, and a function $\phi : A \times 2^A \to 2^P$, where $\phi(a, A')$ is the subset of elements in A' reachable from a through a path of agents not in A'. If the behaviour is full blocking for all agents in this connected component (i.e., $\forall a_i \in A$ (**fullBlocking**(b, a_i))), then there exists an upper bound to the total number of potential norm violations in mutually consented joint actions that can be executed by one of the agents in the joint action when the other agent is part of A'. This upper bound is $\sum_{a \in A'} |(\eta(a) \cap A') \cup \phi(a, A')| + \sum_{a \notin A'} |\phi(a, A')|$.*

Proof It is easily seen that agents in the enforcing component A' will only be able to execute norm violations against the other agents in A' once through each neighbour in A', and if it has neighbours outside A' it will be able to execute norm violations against other agents in A' as many times as agents in A' it can reach through a path of agents outside A'. Whereas, those agents outside A' will only be able to execute a norm violation against agents in A' as many times as agents in A' it can reach through a path of agents outside A'. □

5 The scenario

Section 4 showed analytical results for multiagent networks where all agents exhibit the same type of enforcement behaviour. In a system where agents are assumed to be autonomous, one cannot expect that all agents will exhibit the same behaviour. Even though some of the analytical results have been generalised for subsets of the network that did exhibit the same enforcement behaviour (see Corollaries 1 and 2), we would like to verify how different types of behaviours would work in plural societies. Another issue with the analytical model is that it assumes that each executed joint action is atomic, meaning that no change in the environment can occur through the process of finding a socially feasible path. This may not be a valid assumption in real scenarios where the path may be found to be feasible, but stops being so before the joint action is eventually executed through the path. This may happen when an agent receives disclosed contents of a joint action that is a norm violation after the creation of a path leading to it or coming from it, but before the joint action is executed. For the reasons above, we have run experiments that test the following hypotheses in a simulated environment without the restrictions of uniform societies and atomic execution:

Hypothesis 1 Stricter enforcement behaviours reduce the number of norm violations.

Hypothesis 2 A larger ratio of agents with enforcing behaviours reduces the number of norm violations.

Hypothesis 3 The network topology influences the number of norm violations.

The scenario that has been used in the experiments follows the model described in the previous sections. The simulated environment consists of a multiagent network where agents take turns to start an interaction by first searching for a feasible path, then executing a joint action through it, and finally going through a disclosure stage in which they may or may not send the joint action contents to the path mediators.

5.1 Agents

As seen in Sect. 3, the system has a behaviour which is modelled via three functions. There is a potentially infinite number of possible behaviours. Nonetheless, in this work we do not aim to cover all the different behaviours, only a reduced set of coarse grained behaviours to test how the enforcement techniques work against norm violators.

The agents in our experiments can be classified under one of the three following types: meek, violator, and enforcer. Each type having a behaviour with different properties. Some of these properties were already defined in Sect. 4.5, such as violator avoiding, blocking, protecting, and informing. The rest are defined below:

Definition 17 Given a multi-agent network $\mathcal{N} = \langle A, \eta, C \rangle$ with a normative behaviour function ν, and a behaviour $\langle \alpha, \pi, \delta \rangle$, we define the following behavioural properties for an agent $a \in A$:

1. *Disclosing*: Will always report all its neighbours as potential partners, i.e., $\pi(a_i, a, e) = \langle a_i, a, A' \rangle \to A' = \eta(a)$.
2. *Friendly*: Always consents to act jointly, i.e., $(\alpha(a, a_i, e) = \langle \langle a, a_2, \ldots, a_i \rangle, c, d, J \rangle \to c \neq \texttt{void}) \wedge (\alpha(a_i, a, e) = \langle \langle a_i, a_2, \ldots, a \rangle, c, d, J \rangle \to d \neq \texttt{void})$.
3. *Abiding*: Always abides by the norm, i.e., $(\alpha(a, a_i, e) = \langle \langle a, a_2, \ldots, a_i \rangle, c, d, J \rangle \to c \in \nu(J, a, a_i)) \wedge (\alpha(a_i, a, e) = \langle \langle a_i, a_2, \ldots, a \rangle, c, d, J \rangle \to d \in \nu(J, a, a_i))$.
4. *Violating*: May violate the norm, i.e., $\exists a_i \in A, \exists e \in E \ ((\alpha(a, a_i, e) = \langle \langle a, a_2, \ldots, a_i \rangle, c, d, J \rangle \to c \notin \nu(J, a, a_i)) \vee (\alpha(a_i, a, e) = \langle \langle a_i, a_2, \ldots, a \rangle, c, d, J \rangle \to d \notin \nu(J, a, a_i)))$.
5. *Hiding*: Will avoid joint actions with those that disclose norm violations, i.e., $(\alpha(a, a_i, \langle F', G', D' \rangle) = \langle \langle a, a_2, \ldots, a_i \rangle, c, d, J \rangle \wedge \exists \langle a_i, \langle \langle a'_1, \ldots, a, \ldots, a'_n \rangle, c', d', J' \rangle \rangle \in D' ((a'_1 = a_i \wedge d' \notin \nu(J', a'_n, a'_1)) \vee (a'_n = a_i \wedge c' \notin \nu(J', a'_1, a'_n))) \to c = \texttt{void}) \vee (\alpha(a_i, a, \langle F', G', D' \rangle) = \langle \langle a_i, a_2, \ldots, a \rangle, c, d, J \rangle \wedge \exists \langle a_i, \langle \langle a'_1, \ldots, a, \ldots, a'_n \rangle, c', d', J' \rangle \rangle \in D' ((a'_1 = a_i \wedge d' \notin \nu(J', a'_n, a'_1)) \vee (a'_n = a_i \wedge c' \notin \nu(J', a'_1, a'_n))) \to d = \texttt{void})$.
6. *Discreet*: Never discloses the joint action contents to mediators, i.e., $\delta(a, a_i, e) = (\langle a_i, g \rangle \vee \perp) \wedge \delta(a_i, a, e) = (\langle a_i, g \rangle \vee \perp)$.

The agent type depends on which of the previous behavioural properties hold. As mentioned earlier the three main categories of agents we consider are: meeks, violators, and enforcers. A behaviour $\langle \alpha, \pi, \delta \rangle$ is of type *meek* for a given agent if it is disclosing, friendly, abiding, and discreet. On the other hand a behaviour is of type *enforcer* for a given agent if it is avoiding and abiding. Furthermore, an enforcer behaviour may also be informing, blocking, or protecting (all of which are enforcement properties). An enforcement behaviour b_e is

stricter than another $b_{e'}$ if b_e satisfies all the enforcement properties that $b_{e'}$ does and more. Finally, a behaviour is of type *violator* if it is disclosing, friendly, violating, and discreet. Violator agents in our simulations try to maximise the number of times they violate the norm. Furthermore, violator agents may also exhibit a hiding behaviour and will avoid selecting an enforcer agent as partner.

One can envision other more fine-grained behavioural functions for the different types of agents, such as enforcers that forgive violators by removing blockages after a certain number of rounds following the norm violation. The number of rounds after which forgiveness is granted could be made dependant on the number of known norm violations, making for an even more fine-grained enforcement behaviour. One could also think of more sophisticated violator behaviours, such as violators that block enforcers or violators that do not violate against agents that have exhibited blocking or protecting behaviours. Finally, all these behaviours could be probabilistic (e.g., if implemented to depend on the reputation of the other agent). Nonetheless, we have chosen to leave the study of these variations for future work, as our aim is to show that certain enforcement techniques reduce the number of norm violations for some applications and set a limit to the amount of norm violations possible in certain scenarios, rather than to find the optimal enforcement behaviour for any application.

5.2 Variables

In order to execute the simulations we had to give values to the different variables that define our experimental scenario. Table 1 shows the list of variables considered, the range of values they may take, and the values we have used for the experiments. Some of these variables are specifically mentioned in the hypotheses. Hypothesis 2 makes a reference to the ratio of enforcing agents. The higher the value of this variable, the smaller the number of norm violations. Hypothesis 1 has an implicit reference to the three variables on the type of enforcement being used. Simulations where one of these variables is set to true have fewer norm violations than simulations where they were set to false. Finally, Hypothesis 3 references the network topology explicitly by expecting some topologies to allow different numbers of norm violations.

It is quite straightforward to see that the number of rounds and the ratio of violating agents will also have an impact in the number of norm violations. Not so obvious is the relation

Table 1 Simulation variables

Name	Range	Selected values
Number of agents	$\mathbb{N}_{>1}$	10,20,100,200
Network topology		Tree, small-world, free-scale, random
Number of rounds	$\mathbb{N}_{>0}$	10,20,100,200
Agent ordering		Random
Ratio of enforcing agents	[0,100]%	10, 30, 50%
Enforcement via blocking	$\mathbb{B}$	True, false
Enforcement via protecting	$\mathbb{B}$	True, false
Enforcement via informing	$\mathbb{B}$	True, false
Ratio of violating agents	[0,100]%	10, 30, 50%
Violating by avoiding enforcers	$\mathbb{B}$	True,false
Partner search strategy		Random, BFS, DFS

between the type of violator behaviour (both by choosing whether or not to avoid enforcers, and in the way they search for feasible paths), or the agent order which dictates their position in the network and the order in which they attempt to interact. Furthermore, one can presume that the total number of agents will not have an impact in the number of norm violations. Nonetheless, the experiments have not been designed to test any of these latter hypotheses.

For the variables taking natural numbers as values we have selected four points from a pseudo-logarithmic scale. For the variables representing ratios we have selected three points in a linear scale. In our simulations, agents cannot both enforce and violate the norm. Therefore, the enforcing agent ratio creates a constraint for the violating agent ratio and vice-versa (e.g., a MAN with 80% of enforcers can have at most 20% of violators). In order to bypass this constraint, the selected values are always below 50%. Furthermore, 0% has not been selected as a value, since societies without enforcers or violators are not relevant to test our hypotheses. For variables that do not have a numerical range we have selected those values that we deem interesting: the network topologies that have been chosen are sufficiently different among each other, and most can be found in real societies, and partner search strategies that have been chosen are those that had been considered in Sect. 4.5, and which can maximise the number of norm violations done by agents (Breadth First Search and Depth First Search) plus a fully random search (norm abiding agents will always follow a random search strategy). Finally, the agent order has been chosen at random, although having a finite set of permutations of agents as possible orders.

The topologies we have selected are defined below. A *tree* is a connected undirected graph without cycles. Trees tend to have a large diameter and an average clustering coefficient[3] of 0. Trees are abstractions of highly hierarchical structures. A *random* graph is one that has been generated through some random process. Random graphs have a small diameter and a low average clustering coefficient. A *small-world* graph is one which has a small diameter but an average clustering coefficient orders of magnitude higher than those of a random graph with the same order and size. Finally, a *scale-free* graph is one where the number of neighbours follows a power-law distribution. These types of graphs have small diameter and low average clustering coefficient. They are neither as structured as trees nor as unstructured as random graphs. Networks presenting the small-world property or the free-scale property can be found in many social systems.

5.3 Feasible path search algorithm

Algorithm 1 is executed by the initiator agent in order to find a socially feasible path leading to a partner. First, if the current mediator is different than the initiator and it is selected as partner, then the algorithm returns the path together with the visited agent set passed as parameters to the function (line 1). Otherwise the initiator queries the current mediator for the set of potential partners (line 1) and the function iterates through the set of non-visited potential partners until either a feasible path is found or the set is exhausted. If the set is exhausted but no feasible path has been found, then the algorithm returns an empty path together with an updated set of visited agents (line 1). Otherwise it returns the path from the recursive call (line 1).

Algorithm 1 defines an abstraction of the search for socially feasible paths. Nonetheless, agents can implement different search strategies, as explained previously (random, breadth-first, or depth-first). These strategies are implemented through particular definitions of the functions `get_element_from` and `select_as_partner`. The implementation

---

[3] A graph's average clustering coefficient is the probability that any two neighbours of a given agent are also neighbours.

Algorithm 1: Feasible path search—$\texttt{path}(a_1, a_i, V, p)$

Input: $a_1 \in A$ - initiator agent
Input: $a_i \in A$ - current mediator
Input: $V \in 2^A$ - visited agents
Input: $p \in P$ - feasible path
Output: feasible path
Output: set of visited agents

```
1  request aᵢ the potential partners for a₁;
2  ⟨a₁, aᵢ, A'⟩ ← π(a₁, aᵢ, e);
3  N ← A' \ V;
4  if aᵢ ≠ a₁ ∧ select_as_partner(aᵢ, N) then
5      return (p, V);
6  else
7      while N ≠ ∅ ∧ ¬stop do
8          aⱼ ← get_element_from(N);
9          N ← N \ {aⱼ};
10         V ← V ∪ {aⱼ};
11         (p', V) ← path(a₁, aⱼ, V, p · aⱼ);
12         if p' ≠ ⟨⟩ then
13             stop ← ⊤;
14         end
15     end
16     if N = ∅ ∧ ¬stop then
17         return (⟨⟩, V);
18     else
19         return (p', V);
20     end
21 end
```

---

of these functions for the breadth-first and depth-first searches is straightforward and will not be discussed. The random search implements the $\texttt{get\_element\_from}$ function by returning an agent in a purely random fashion, and it implements the $\texttt{select\_as\_part-ner}$ by returning true with a certain probability. In our experiments this probability has been set to 0.3, which constructs paths with length following a geometric distribution with mean 10/3. This is enough to guarantee that all agents can interact with one another.

Many different methods exist to generate the previously mentioned graphs. We have chosen the following: The generated tree structures are k-ary trees ($k = 9$), i.e., a tree with up to k children per node. Random graphs have been generated following the Erdös–Rényi model [14], in which the probability that an edge between two vertices exists is 0.1. Finally, the small world graphs have been generated following the Watts–Strogatz model [39] starting with a ring lattice of degree 10, and a rewiring probability of 0.02. Finally, scale-free graphs have been generated using the Barabasi–Albert model [5] with a size and order similar to that of the small-world and random graphs.[4]

The experiment consists of exhaustive simulations with all the possible combinations of variable values. The metric we have used to test the hypothesis is the norm violation rate, i.e., the ratio of forbidden actions executed per violator agent and round. This metric is a

---

[4] All networks are generated randomly using one of the previous methods, and the agents are placed randomly in a graph position.

Table 2 Tukey's test results

Variable	Value pair	Difference (%)	Lower bound (%)	Upper bound (%)
Enforcers	10–30%	8.83	8.71	8.95
Enforcers	30–50%	8.96	8.84	9.08
Enforcers	10–50%	17.79	17.67	17.91
Topology	Tree–random	13.37	13.22	13.52
Topology	Tree–small world	13.01	12.86	13.16
Topology	Tree–scale free	12.73	12.58	12.88
Topology	Scale free–random	0.64	0.49	0.79
Topology	Small world–random	0.36	0.21	0.51
Topology	Scale free–small world	0.28	0.13	0.43
Protecting	False–true	7.37	7.29	7.45
Blocking	False–true	3.05	2.97	3.13
Informing	False–true	0.26	0.18	0.34

normalisation of the total forbidden actions so that we can compare simulations with different number of agents and rounds.

6 Simulations

This section shows the results of the experiments that have been executed following the scenario specified in Sect. 5. For each of the hypotheses presented we explain the statistical analysis that has been realised on the experimental data and the results of such analysis.

In order to test the three hypotheses we have executed a factorial experiment. The experiment consists of running a number of simulations (20 in total) for each of the parameter combinations in Table 1.[5] Before executing the statistical significance test we verified that the resulting data had a normal distribution through a Quantile–Quantile test, which is a precondition for certain statistical tests.

In order to test if there was a significant relationship between the independent variables, i.e., the parameters in the simulation, and the dependent variable, i.e., percentage of the total potential interactions per violator agent that resulted in a norm violation (from now on we will refer to this as the norm violation rate), we ran an analysis of variance (ANOVA) with the experimental data. The results of the ANOVA test demonstrate that all the independent variables were statistically significant with a p-value under 0.001. This information alone serves to support Hypothesis 3, since the topology variable proved to be significant. Data showing which variable values brought about lower norm violation rates is needed in order to support the other two hypotheses.

In order to verify which variable values did better for each of the variables that interested us with respect to the hypotheses, we ran post-hoc comparisons using Tukey's test (see the results in Table 2). One test indicated that higher percentages of enforcers implied a smaller mean of norm violation rates, thus supporting Hypothesis 2. Furthermore, another Tukey test indicated that the best network topology in reducing the norm violation rate was the random topology. Close behind were the small world and scale free topologies, and, lagging behind,

---

[5] There are 27, 648 total combinations. In total 552, 960 simulations were executed.

the tree topology was the one to do worst. Finally, Tukey tests on the enforcement behaviours showed that the three tested behaviours helped to reduce the norm violation rate. This information supports Hypothesis 1, since stricter enforcement behaviours reduced the norm violation rate. Out of the three behavioural properties, the property that produced the best results was the protecting property which lowered the norm violation rate by an average of 7.37%. Next came the blocking property which lowered the norm violation rate by an average of 3.05%, and last was the informing property that barely lowered the norm violation rate, a meagre 0.26%.

The Tukey test results on different topologies were a surprise to us as they countered our intuitions. Experiments on prior work [12,13] had showed that tree networks were more efficient in reducing norm violation rates than other topologies. This, together with the analytical results obtained in Theorem 2, made us think that the tree network would still be the one to achieve the best enforcement performance. Nonetheless, the results from the experiments showed that it is random networks that have the best enforcement performance and tree networks have the worst. Such results are brought about by the fact that trees have the highest betweenness centrality. Agents in a high betweenness centrality position will collect more joint action contents through disclosure. In our prior work, only enforcer agents benefited from their high betweenness centrality, which made tree networks more efficient for enforcement. In the current scenario, violator agents also collect information to find out who the enforcers are in order to avoid them. Therefore, they also benefit from high betweenness centrality positions, which makes trees worse for enforcement. Furthermore, our simulations restricted the maximum percentage of enforcers to 50%, making it easy for violators to find non-enforcers with which to execute norm violating actions.

Another result from the Tukey test that surprised us was the low reduction brought about by the informing behavioural property. At first glance, we thought that disclosing contents of norm violations would be a good enforcement behaviour. Nonetheless, the slight improvement brought about by the informing behavioural property made us look into this in more detail. Further tests showed that in simulations were the number of rounds is larger than the number of agents, the informing behavioural property actually increases the mean norm violation rate by 0.40% on average. Furthermore, when the number of rounds is equal to the number of agents there is no significant difference between results with and without the informing behavioural property. Finally, when the number of rounds is smaller than the number of agents the norm violation rate is decreased by 1.15%. Table 3 shows the results of the Tukey tests with the data subsets mentioned before. This is due to the fact that violators, in our simulations, avoid enforcers and these are detected when they disclose joint action contents. It is during bootstraping (when the number of rounds is small and agents had no time to interact with one another) that disclosure allows enforcers to quickly discover all violators. As the simulation continues, it is the violators who end up discovering all enforcers through their informing behavioural property, and can thus easily avoid them. This tendency could be changed if those enforcers disclosing the joint action contents could select a subset

Table 3 Informing variable Tukey test results for subsets of the simulation data

Data set	Value pair	Difference (%)	Lower bound (%)	Upper bound (%)
Rounds > agents	False–true	−0.40	−0.54	−0.26
Rounds = agents	False–true	−0.09	−0.26	0.08
Rounds < agents	False–true	1.15	1.04	1.26

of the mediator agents to which the contents is sent. This way enforcers would avoid being discovered by violators and the enforcement benefits of disclosure be improved.

7 Applications

This section presents two applications that could make use of our model and enforcement techniques. In order to apply these techniques, the applications need to have an objective norm function which is shared by all, the norm violating behaviours should not be too sophisticated, and norm violations should only affect those agents involved in the joint action.

The first application is a distributed forum for information sharing in which a group of agents forms a network of contacts through which they exchange information. In this forum there are norms specifying the valid information contents that can be shared. These norms must be verifiable objectively by any agent. Furthermore, agents violating these norms would either do it continuously in the same way as spammer agents in fora, or it would violate occasionally by mistake or because the recipient does not care for the specific norm violation. Both types of violators fit the specification of Sect. 5

The second application is for a network of hardware nodes (such as a sensor network) which co-operate by executing a specific protocol in order to manipulate information. The norm in this case would define the protocol to be followed. Such protocol ought to be objectively verifiable. Furthermore, all nodes are assumed to work well unless there is a hardware malfunction, in which case they will start breaking the protocol indefinitely.

In the following subsections we will show how our model and enforcement techniques are applied to the applications, and the results that can be achieved from these enforcement techniques.

7.1 Information sharing forum

In order to model the distributed information sharing forum application, we must start by defining the MAN $\mathcal{N} = \langle A, \eta, C \rangle$ with a normative behaviour function ν. Having A be the set of participating agents in the forum (let us assume it is 100 in total), η be the function that describes the connections among the agents (let it be a network with 500 links and the average degree is 10), C is the set of all content that can be shared, and ν is the function that specifies which contents are valid.

A joint action in this application means that one agent sends some information through a path of agents in the network to another agent. The action executed by the receiving agent indicates what it did with the information. When it executes the `void` action, it is telling the initiator that the information transfer has been immediately aborted. It is as if it had never been received. A norm in this application might specify that the maximum size of exchanged information is 10 MB, and that specific file types such as mp3 or text documents containing words from a previously established list of insults are not allowed. This norm predicate is objectively verifiable and is not influenced by the environment. Therefore, we will not mention the environment in the rest of this application. Furthermore, the `void` action is always permitted.

Let us assume that there are ten agents that are norm violators. Five of which are full spammers which will always send content that is not allowed by the norms. The other five will send invalid content rarely when they think they can get away with it. The results from Sect. 4.5 tell us that: if all agents in the application follow an avoidance behavioural property,

the maximum number of norm violating contents that can be sent is 9,900; if all agents exhibit a full blocking behaviour, this number is reduced to 1,000. Nonetheless, since there are only 10 violator agents, when all agents exhibit an avoidance behavioural property the upper bound is reduced to 990 (see Lemma 1) and when they all exhibit a full blocking behaviour it is reduced to 100 (see Lemma 2).

In case that the norm violators do not exhibit enforcement behaviours, the premises from the theorems would not hold but those of the corollaries would and we would apply their conclusions. Assuming that all norm abiding agents apply an avoidance behavioural property, then the maximum number of norm violating contents shared with norm abiding agents would be 900, and approximately 90 in case they all exhibited full blocking behaviours. This has been calculated assuming that all agents (including norm violators) have ten neighbours and probabilistically 9 of these neighbours will be norm abiding agents, and that all norm abiding agents form a connected component in the network.

When the agent's behaviours are not organised in any of the previous ways, the results from the simulations can still tell us how the system would behave. The application designer would then be interested in providing a free and accessible implementation of the information sharing agent that would embed a full blocking behaviour and leave other implementations to the users, thus, creating a cost for the implementation of other behaviours. Furthermore, when possible, the designer would be interested in organising the network randomly, since it was the random network that got the best enforcement performance.

7.2 Self-repair system

The self repairing network application is defined through the model as $\mathcal{N} = \langle A, \eta, C \rangle$, where A is the set of all nodes (let us assume a total of 100), η describes the connections among nodes, C defines the set of interaction data that can be sent, and a normative behaviour function ν that specifies the protocol for interaction.

In this application a joint action means that the initiator agent sends some data to the partner agents as part of a broader protocol. The protocol defined by the function ν is a simple query answer protocol where the void action can only be executed by the partner agent as long as there has been a protocol violation in the past. When the partner agent executes the void action it is letting the initiator know that it is not listening to its message. In this application the environment consists of the previous joint actions executed by the agents in the joint action.

In this application the system designer has complete control over the agents' behaviours and the network topology. The only reason for a violator to exist is due to hardware malfunction. Therefore, once an agent violates a norm, chances are that it will continue to misbehave until its hardware is repaired. Furthermore, the malfunctioning node has no capabilities to evade enforcement through sophisticated norm violation behaviours.

From the description of the application we deduce that it is in the best interest of the application designer to make all nodes have a full-blocking behaviour. Since agents that have failed will not have a hiding behavioural property to evade enforcement, a tree network might seem the most appropriate topology. Nonetheless, since a node that has failed will probably stop reporting its neighbours correctly (which is the same as if the agent would block everyone in the network), then a tree network is dangerous as it would easily compromise a whole sub-tree. Therefore, the best topology would be a random network with an average degree ensuring that the network forms a connected component.

When the self repairing application is designed as explained above, the upper bound in the number of failed protocols would be the number of links in the network. This number is equal to the number of agents times the tolerance to errors per agent. When the probability that edges exist in a random graph is $\frac{2\ln n}{n}$, where n is the number of vertices, the probability that the network is connected tends to 1 [14]. Therefore, to ensure that a random network is connected, the number of links would be $\ln n(n-1)$. Since our application comprises 100 agents, then the number of links would be at least 456. In order to make sure that errors in hardware do not split the network, the system designer should allow for more links (e.g., 500). Finally, in the previous setup the maximum number of protocol failures (i.e., norm violations) allowed would be 500, or an average of 5 per agent.

8 Discussion

We have provided a model for a multiagent system structured as a network where agents interact under a defined normative behaviour. Under this model, a set of enforcement techniques have been proposed to reduce the number of norm violations, namely: avoidance, blocking, protecting, and informing. Via analytical means we have shown that, when all agents (or a subset of them) exhibit behaviours with enforcement properties, the number of executed norm violations has an upper bound. Nonetheless, agents are autonomous and may decide to save up resources by not applying sanctions. In order to apply sanctions, agents must remember which other agents have executed a norm violating action in the past and take this information into account when helping to build socially feasible paths. If agents exhibiting this free-riding behaviour exist in the system, the analytical results in Sect. 4 do not always hold. However, the results of the experiments in Sects. 5 and 6 show us how such a system could be expected to behave. Firstly, the higher the number of agents using enforcement techniques the smaller the number of times violator agents would be able to execute norm violating actions. Secondly, using more of the enforcement techniques implies less norm violations. Nonetheless, we encountered results that challenged our intuitions. The experiments showed that disclosure of joint action contents does not always reduce norm violations, since overuse of disclosure can put violators under notice and help them avoid future enforcement. Therefore, disclosure should be restricted to trusted agents or be used only at bootstrap. Finally, the network structure is an important factor in reducing norm violations. Notwithstanding, we had expected tree networks to continue to be the best at reducing norm violations, as in previous work, and this did not happen. When violators use disclosure information to spot enforcers they also benefit from the high centrality positions possible in tree networks, making tree networks the worst topology for enforcement for sophisticated violators that manage to "climb the ladder". The dependence network formalism could be a good lead to study the power agents have to enforce norms.

Builders of an application, in which the enforcement techniques in this paper could be used, should try to get as many agents in the system to enforce the norm through the four techniques provided: avoiding, blocking, protecting, and informing. We have not aimed in this work to study what could motivate a selfish agent to act in specific ways. We have provided information about which behaviours benefit the society by reducing the number of norm violations. Future work could study how to create incentives so that agents exhibit the enforcement behaviours in order to diminish the potential incentives for executing norm violations. Furthermore, if builders have total control over the network topology, they should create a tree in which all the non-leaves would be under his control and would use all the full blocking behaviour. Otherwise, the tree network should be avoided. The impact of other

network parameters (e.g., clustering factor, diameter, number of links per agent, number of paths between agents, and position of agents) on norm enforcement should be studied. In order to help individual agents decide how to influence the network topology for its benefit. The study of the position of agents may prove specifically interesting. For instance, if an agent is a cut vertex in the graph and it uses the protecting behavioural property against its neighbours it can break the graph in two, not allowing abiding agents to interact. Furthermore, a violator that is in such a position may threaten to split the graph in order to avoid enforcement. These problems can be circumvented with new links.

It was out of the scope of this paper to treat network dynamics by which the network topology changes by adding and removing links, or by adding new agents to the system. Even so, agents should be extremely weary when adding new links in order not to allow agents to execute more norm violations through them. Future work may deal with this issue by adding another type of sanction in which an agent is sanctioned because it is too promiscuous in adding new links through which norm violations are executed. Another possibility is to mark new links as "high risk" until they have proven to be trustworthy. A high risk link is treated differently in that no neighbours are returned to any agent coming through it. Nonetheless, joint actions would not be avoided, since it is through them that trustworthiness would be gained.

The enforcement techniques in this work are meant for scenarios in which the agents do not know a priori with which agent to share information. Notwithstanding, they could also be applied to scenarios in which the partner agent is known and a feasible path in the network is to be found. In such a case routing mechanisms used in networking could be used in order to send the requests through the social networks. Since the partner agent would be known from the start of the process, the protecting behaviour would have to be modified but others could also be added.

Some applications that would benefit from our works have been provided in Sect. 7. These applications have to fulfil the restrictions imposed by the model, such as the objectivity of the norm and bareness of the agent behaviours available. Nonetheless, other applications that do not fulfil these restrictions could also benefit from the concepts defined here. In order to achieve this, future work should study the impact of subjective norms, where each agent decides what is satisfactory for it, and how more sophisticated enforcement behaviours could achieve better enforcement performance against more sophisticated violating behaviours.

References

1. Ågotnes, T., Van der Hoek, W., Rodríguez-Aguilar, J. A., Sierra, C., & Wooldridge, M. (2007). On the logic of normative systems. In *Proceedings of the 20th international joint conference on artificial intelligence (IJCAI 07)* (pp. 1175–1180). AAAI Press.

2. Aldewereld, H., Dignum, F., García-Camino, A., Noriega, P., Rodríguez-Aguilar, J. A., & Sierra, C. (2006). Operationalisation of norms for usage in electronic institutions. In *AAMAS'06: Proceedings of the 5th international joint conference on autonomous agents and multiagent systems* (pp. 223–225). New York, NY: ACM.

3. Axelrod, R. (1985). *The evolution of cooperation*. New York: Basic Books.

4. Axelrod, R. (1986). An evolutionary approach to norms. *The American Political Science Review, 80*, 1095–1111.

5. Barabasi, A. L., & Albert, R. (1999). Emergence of scaling in random networks. *Science, 286*(5439), 509–512.

6. Broersen, J., Dignum, F., Dignum, V., & Meyer, J.-J.C. (2004, May). Designing a deontic logic of deadlines. In *7th International workshop of deontic logic in computer science (DEON'04)* (pp. 43–56), Portugal.

7. Carpenter, J., Matthews, P., & Ong'ong'a, O. (2004). Why punish: Social reciprocity and the enforcement of prosocial norms. *Journal of Evolutionary Economics, 14*(4), 407–429.

8. Castelfranchi, C. (2000). Engineering social order. In *ESAW'00: Proceedings of the 1st international workshop on engineering societies in the agent world* (Vol. 1972, pp. 1–18). Springer.

9. Castelfranchi, C., Conte, R., & Paoluccci, M. (1998). Normative reputation and the costs of compliance. *Journal of Artificial Societies and Social Simulation, 1*(3).

10. Cranefield, S. (2005, February). *A rule language for modelling and monitoring social expectations in multi-agent systems*. Technical Report 2005/01, University of Otago.

11. Delgado, J. (2002). Emergence of social conventions in complex networks. *Artificial Intelligence, 141*(1), 171–185.

12. de Pinninck, A. P., Sierra, C., & Schorlemmer, M. (2007). Friends no more: Norm enforcement in multi-agent systems. In *Proceedings of the 6th international joint conference on autonomous agents and multi-agent systems* (pp. 1–3). ACM.

13. de Pinninck, A. P., Sierra, C., & Schorlemmer, M. (2008). Distributed norm enforcement: Ostracism in multi-agent systems. In *Computable models of the law, Lecture Notes in Artificial Intelligence* (Vol. 4884, pp. 275–290). Springer.

14. Erdös, P., & Rényi, A. (1960). *On the evolution of random graphs* (Vol. 5, pp. 17–61). Publ. Math. Inst. Hung. Acad. Science.

15. Esteva, M., Padget, J., & Sierra, C. (2002). Formalizing a language for institutions and norms. In *ATAL'01: Revised papers from the 8th international workshop on intelligent agents VIII, Lecture Notes in Artificial Intelligence* (Vol. 2333, pp. 348–366). Springer.

16. Esteva, M., Rosell, B., Rodriguez-Aguilar J. A., & Arcos, J. L. (2004). AMELI: An agent-based middleware for electronic institutions. In *Proceedings of the 3rd international joint conference on autonomous agents and multiagent systems (AAMAS'04)* (pp. 236–243). IEEE Computer Society.

17. Fon, V., & Parisi, F. (2005). The behavioral foundations of retaliatory justice. *Journal of Bioeconomics, 7*(1), 45–72.

18. García-Camino, A., Rodríguez-Aguilar, J.-A., Sierra, C., & Vasconcelos, W. (2007, May). Norm-oriented programming of electronic institutions: A rule-based approach. In *Coordination, organizations, institutions, and norms in agent systems II, Lecture Notes in Computer Science* (Vol. 4386, pp. 177–193). Springer-Verlag.

19. Grizard, A., Vercouter, L., Stratulat, T., & Muller, G. (2007). A peer-to-peer normative system to achieve social order. In *Coordination, organizations, institutions, and norms in agent systems II* (Vol. 4386, pp. 274–289). Springer.

20. Hales, D. (2002). Group reputation supports beneficent norms. *Journal of Artificial Societies and Social Simulation, 5*(4).

21. Hübner, J. F., Sichman, J. S., & Boissier, O. (2006). S-$MOISE^+$: A middleware for developing organized multi-agent systems. In *Coordination, organizations, institutions, and norms in multi-agent systems, Lecture Notes in Computer Science* (Vol. 3913, pp. 64–78). Springer.

22. Jackson, M. O. (2003). Mechanism theory. In U. Devigs (Ed.), *Optimization and operations research, the encyclopedia of life support science*. Oxford, UK: EOLSS Publishers.

23. Kittock, J. E. (1994). The impact of locality and authority on emergent conventions: Initial observations. In *AAAI'94: Proceedings of the 12th national conference on artificial intelligence* (Vol. 1, pp. 420–425). Menlo Park, CA: American Association for Artificial Intelligence.

24. Marín, R. H., & Sartor, G. (1999). Time and norms: A formalisation in the event-calculus. In *IC-AIL'99: Proceedings of the 7th international conference on artificial intelligence and law* (pp. 90–99). New York, NY: ACM Press.

25. Maskin, E. S., & Sjöström, T. (2002). Implementation theory. In K. J. Arrow, A. K. Sen, & K. Suzumura (Eds.), *Handbook of social choice theory and welfare*. Amsterdam: North-Holland.

26. Minsky, N. H. (1991). Law-governed systems. *Software Engineering Journal, 6*(5), 285–302.
27. Nisan, N. (2007). Introduction to mechanism design (for computer scientists). In N. Nisan, T. Rough-garden, E. Tardos, & V. V. Vazirani (Eds.), *Algorithmic game theory*. Cambridge, UK: Cambridge University Press.
28. Padget, J. A., & Bradford, R. J. (1999). A pi-calculus model of a spanish fish market—preliminary report. In *AMET'98: Selected papers from the 1st international workshop on agent mediated electronic trading on agent mediated electronic commerce* (pp. 166–188). London, UK: Springer.
29. Parkes, D. C. (2001, May). *Iterative combinatorial auctions: Achieving economic and computational efficiency*. PhD thesis, University of Pennsylvania, Department of Computer and Information Science.
30. Pujol J. M., Delgado, J., Sangüesa, R., & Flache, A. (2005). The role of clustering on the emergence of efficient social conventions. In *IJCAI'05: Proceedings of the 19th international joint conference on artificial intelligence* (pp. 965–970).
31. Robertson, D. (2005). A lightweight coordination calculus for agent systems. In *Declarative agent languages and technologies II* (Vol. 3476, pp. 183–197). Springer.
32. Savarimuthu, B. T. R., Purvis, M., Cranefield, S., & Purvis, M. (2007). Role model based mechanism for norm emergence in artificial agent societies. In *Proceedings of the international workshop on coordination, organization, institutions, and norms (COIN)*. Honolulu, HI.
33. Sergot, M. (2001). A computational theory of normative positions. *ACM Transactions on Computational Logic, 2*(4), 581–622.
34. Sergot, M., & Robert, C. (2006). The deontic component of $n\mathcal{C}+$. In *8th International workshop on deontic logic in computer science (DEON'06), Lecture Notes in Computer Science* (Vol. 4048, pp. 222–237). Springer.
35. Taylor, M. (1982). *Community, anarchy & liberty*. Cambridge: Cambridge University Press.
36. Vazquez-Salceda, J., Aldewereld, H., & Dignum, F. (2004) Implementing norms in multiagent systems. In G. Lindemann, J. Denzinger, I. J. Timm, & R. Unland (Eds.), *Multiagent system technologies* (LNAI 3187, pp. 313–327). Springer.
37. von Wright, G. H. (1951). Deontic logic. *Mind, 60*, 1–15.
38. Walker, A., & Wooldridge, M. (1995). Understanding the emergence of conventions in multi-agent systems. In *Proceedings of the 1st international conference on multi-agent systems* (pp. 384–389). San Francisco, CA: MIT Press.
39. Watts, D. J., & Strogatz, S. H. (1998). Collective dynamics of small-world networks. *Nature, 393*(6684), 440–442.
40. Xiong, L., Liu, L., & Ieee Computer Society. (2004). Peertrust: Supporting reputation-based trust for peer-to-peer electronic communities *IEEE Transactions on Knowledge and Data Engineering, 16*, 843–857.
41. Yarbrough, B. V., & Yarbrough, R. M. (1999). Governance structures, insider status, and boundary maintenance. *Journal of Bioeconomics, 1*, 289–310.
42. Yolum, P., & Singh, M. (2002). Flexible protocol specification and execution: Applying event calculus planning using commitments. In *AAMAS'02: Proceedings of the 1st international joint conference on autonomous agents and multiagent systems* (pp. 527–534). ACM Press.
43. Younger, S. (2004). Reciprocity, normative reputation, and the development of mutual obligation in gift-giving societies. *Journal of Artificial Societies and Social Simulation, 7*(1).