Received August 10, 2019, accepted August 28, 2019, date of publication September 2, 2019, date of current version September 18, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2939019

Norm-Based Behavior Regulating Technique for Multi-Agent in Complex Adaptive Systems

MOAMIN A. MAHMOUD[1], MOHD SHARIFUDDIN AHMAD[1], AND SALAMA A. MOSTAFA[2]
[1]College of Computing and Informatics, Universiti Tenaga Nasional, Kajang 43000, Malaysia
[2]Faculty of Computer Science and Information Technology, Universiti Tun Hussein Onn Malaysia, Parit Raja 86400, Malaysia

Corresponding author: Moamin A. Mahmoud (moamin@uniten.edu.my)

ABSTRACT In a complex adaptive system, norms are widely used to regulate agents' behavior within a society in which the agents are not explicitly given the norms of different host systems, but instead detect and adapt the norms autonomously. Thus, failing to adopt a host's norms resulting in deprived of accessing resources and services that would severely affect their performance. Few attempts have been made to overcome this problem but proposed solutions lack accuracy in identifying the norms of a domain. Consequently, this paper demonstrates a new effective technique called Regulative Norms Detection Technique (RNDT) to detect norms by analyzing odd events that trigger reward or penalty. Several tests have been conducted on agents exploiting the technique to detect norms in a virtual domain under varying environmental settings. The tests' results show that the RNDT achieve well although the rate of success relies on the environmental variables settings. Specifically, the result shows that the rate of adopting the domain's norms using RNDT is 78.0%. The rate of rewarded agents in three testing cycles increased from 30% to 70%, the rate of neutral agents also increased from 35% to 45%. The most noticeable change is in the penalty case in which 35% of agents are penalized in cycle 1, while in cycle 2 and 3 the rate is maintained at 0.0%.

INDEX TERMS Social norms, normative systems, norm detection, agent-based simulation.

I. INTRODUCTION

Software agents have been widely utilized to simulate normative systems due to their autonomous, self-interested, rational abilities [36]–[39], and social abilities [40]–[43]. Interactions between a software agent and a human user are many in everyday situations such as access to information, entertainment, and purchases [46]. Therefore, regulating agents' behavior is extremely important [47]. There are several ways to control agent behavior such as Morality, Sincerity, Emotion [48], however, the most popular way is using Norms [49]–[51].

The concepts of norms and normative systems are widely used as efficient means to control and normalize agents' behavior within a social network due to their capability to coordinate agents' interactions [1]–[3], [33]. Generally, Norms are agreed upon rules set by population indicating expected actions to perform that are either prohibitive, obligatory or permissive [4]–[6]. However, to persuade agents to comply with norms, normative systems should be able to extend rewards or penalties [7]–[9].

The associate editor coordinating the review of this manuscript and approving it for publication was Xiwang Dong.

Norms are classified into two types, Conventional norms and Essential norms [11]–[13]. Conventional norms are natural, arise without any enforcement mechanism [14]–[16], while Essential norms mitigate collective action issues when an individual and collective interests have conflict [13], [17]. Three types of essential norms are suggested by literature as shown in Figure 1, regulative norms, constitutive norms, procedural norms [18].

By means of three norms types, obligations, prohibitions and permissions, the regulative norms regulating agents' activities through determining the optimal and sub-optimal behavior [16], [22].

While the constitutive norms revise the normative system by controlling the creation of institutional norms [20]. And finally, the procedural norms which are also known as instrumental norms [20], given to agents who act on normative system roles intending to achieve the social order, particularly in terms of substantive norms [21]. On the other hand, norms detection plays an important role in updating an agent's norms by identifying potential norms of a social network through some detection mechanisms exploiting observation of or interaction with, other agents to According

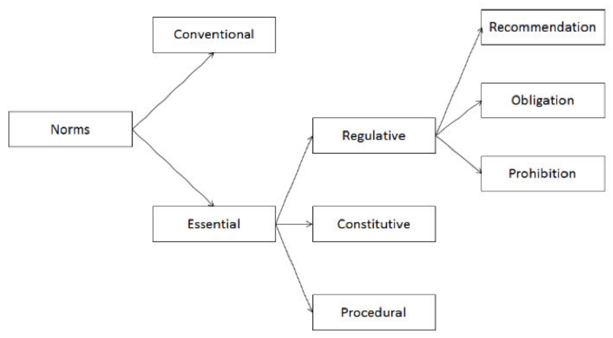


FIGURE 1. **Norms taxonomy.**

to [10], [24], [25], norm detection id one of the main issues faces researchers in building a normative multi-agent system

Few interesting identification techniques have been recently proposed by literature to mitigate the challenges [27]–[32], however, the reasons for the limitation of these studies are either due to the exploited approach or mechanisms or the technical work itself. Therefore, this paper presents a new technique to identify regulative norms and subsequently determine a normative protocol. The normative protocol as "the order of occurrence of events or protocols that are related to a set of norms" [27]. For example, a restaurant normative protocol could be arrive, order, eat, pay, tip and depart. The norms are detected by analyzing odd events that trigger reward or penalty. Several tests have been conducted on agents exploiting the technique to detect norms in a virtual domain under varying environmental settings.

II. RELATED WORK

Research issues on norm detection have been addressed by several studies [27], [28], [29], [31], [32], [44], [45]. Epstein [29] has proposed an imitation model that adopting the behavior of the majority. The process goes through two stages, in the first stage agents initially are given the norms, and in the second stage agents are able to update their norms by observing other agents' actions.

In other work, Lòpez y Lòpez [44], three strategies are proposed to trigger agents to adopt a norm of other agents, the strategies are a simple imitation, reasoned imitation, and reciprocation. In simple imitation, an agent follows a straightforward strategy as part of its desire to conform to other agents. In this strategy, the agent adheres to a norm if other surrounding agents adhere too. Similarly, if other agents violate the norm, the agent violates it too. In reasoned imitation strategy, an agent discovers the behavior of other agents as well as the behavior of its promoters (the agents in charge of the reward) and defenders (the agents in charge of penalty). Four cases relating to discovered behavior are: (i) an agent follows a norm and gets a reward, (ii) the agent follows the norm but does not get a reward, (iii) the agent does not follow the norm and gets a penalty, and (iv) the agent does not follow the norm and does not get a penalty. In reciprocation strategy, agents share common benefits; an agent follows a norm of other beneficiary agents on the condition that other agents fulfilled the norm for which the agent is a beneficiary and vice versa.

Symeonidis et al. developed an algorithm to express an agent's actions called K-Profile. It is utilized using data mining to anticipate agents' behaviors to identify the patterns from a log file and determine the actions of agents. The mechanism entails offline and online processes. The offline processes include (i) observing agent actions, (ii) processing the agent behavior dataset, and (iii) applying data mining algorithm to extract the behavior profiles for storage. The online processes include (i) observing the agent actions, (ii) processing the agent behavior vector, (iii) comparing the current action sequence against the stored profile, and (iv) finally, producing recommendation.

Another work by Andrighetto *et al.* [31] proposed a norm innovation theory within a project called Emergence in the Loop EMIL. The EMIL project consists of three main components, EMIL-M, EMIL-S, and EMIL-T. EMIL-M performs a general model of norm-innovation, EMIL-S is a simulation platform to test EMIL-M by conducting experiments, and EMIL-T evaluates the performance of the model.

Sen and Airiau [32] developed social interaction-based norms learning approach within a social network where every agent learns via interaction with other neighbor agents. They use three learning algorithms, Q-learning which can be used to learn in pure strategies, WoLFPHC (Win or Learn Fast - policy hill climbing) which can be used to learn in mixed strategies, and FP which can be used to learn in a game theory.

Savarimuthu *et al.* [26], [27]: developed two norm identification algorithms based on relation-sequences and frequently occurring events called Obligation Norm Identification (ONI) to identify obligation norms, and Prohibition Norm Inference (PNI) to identify prohibition norms. The PNI algorithm identifies prohibition type norms only but requires a sanctionable action by an agent. The authors exemplified a park scenario, in which an agent, X, observes another agent, Y, being sanctioned because of littering in the park. X then collects the sanctioned event and adds it to its belief base. By using association rule mining and a threshold to identify the candidate norm, X identifies the prohibited norm. The ONI algorithm only identifies obligation type norms and also requires a sanctionable action by an agent.

III. THE RNDT TECHNIQUE

The detection technique is implemented using the concept of a precondition, post-condition and the agent's belief. When the agent observes rewarded or penalized events on other agents or on itself, the observation is considered as a precondition and the event's data is stored in its Record Base, which is a component of its belief base. Based on this belief, it detects the norm that caused the exceptional event which is the post-condition. Briefly, if R is the Record Base, F is an event then,

TABLE 1. The PNDT components.

Component	Description
Agent's Belief Base	The Agent's Belief Base entails three sub-components, log file, and norm model base. The log file contains the records of all observed events and represents the history log. While the Norms Model Base is the storage for all identified norms normative protocols, which triggers rewards or penalties.
Observation Process	The observation is a process of keeping watch events of nearby agents. Observing (events) rely on the availability of other nearby agents. In data collection, the agent does not collect all the data in the domain, only the relevant data.
The Regulative Norm Mining Algorithm	The RMNA mines log file to detect a domain's norms and its normative protocol. It consists of five phases Grouping, formatting, filtering, extracting norms, determining normative protocol.
Verification Process	The verification process is a process that ensures that the identified regulative norms; recommendation, obligation, and prohibition, are valid norms that are being practiced within an agent society. When the regulative norms have been extracted and identified, the agent starts the verification process on each norm. The verification process is implemented in two approaches depending on whether the extracted norms have been previously verified or otherwise. • The first verification approach is by checking the norm in the Norm Model Base then the identified norm is considered as verified. • If the identified norm is not found, then the agent performs the verification process by repeating the entire observation, but this time only for specific events. In this case, the agent acquires, at least, three historical episodes with events that include the norm to verify. Based on our judgment, three historical episodes are used to guarantee accurate verification results. These three historical episodes must show the same result.
Updating Process	The updating process, UP, updates the agent's belief base. The updating process is quite straight-forward, in which the agent simply updates the identified and verified regulative norms and normative protocols at the predefined structure

Precondition:	observe $(\alpha_\kappa, (f_\kappa \in \vartheta_\chi)) \Longrightarrow$ record $(\alpha_\kappa, f_\kappa)$
Belief:	belief $(\alpha_\kappa, (f_\kappa \in \vartheta_\chi))$
Postcondition:	launch $(\alpha_\kappa$, RNDT$) \Longrightarrow$ detect $(\alpha_\kappa, \eta_\kappa)$

The Regulative Norms Detection Technique (RNDT) RNDT framework consists of five main components Agent's Belief Base (AB), Observation Process (OP), Regulative Norms Mining Algorithm (RNMA), verification process (VP), and updating process (UP).

We use the approach of signaling events [27], which assumes that an agent is able to sense its surrounding events and recognize rewarded and penalized events. A rewarded event represents a positive signal and a penalized event represents a negative signal. A positive signal can be any type of rewards e.g., thanking with a smile, while a negative signal can be any type of punishment, e.g., a growl. The next component is RNMA which considers the major contribution of this paper, it is developed to enable agents to analyze the collected events in the loge file and subsequently detects the regulative norms and determine the normative protocol. Having identified a domain's norms, the VP components confirms the detected norms. Finally, the UP component updates the norm model-based on new norms and normative protocol. Figure 1 shows the RNDT framework.

Table 1 presents the PNDT components, however, the rest of this paper focuses on the major contribution which is the Regulative Norm Mining Algorithm (RNMA).

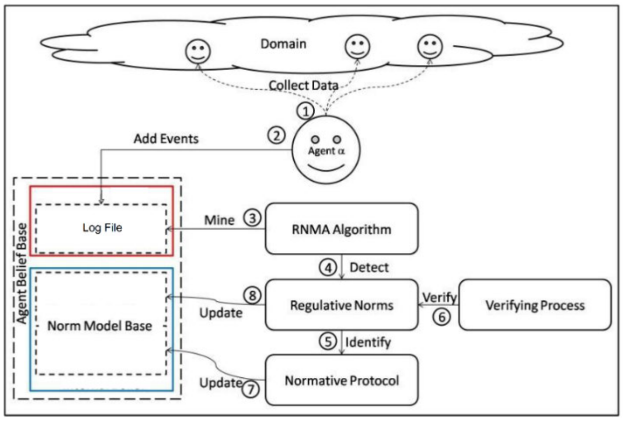


FIGURE 2. The RNDT framework.

As shown in Figure 2,

- An agent first collect data from a domain by observing other agents' actions,
- The agent then stores the collected events to its log file,
- It starts mining the loge file using the RNMA algorithm,
- It then detects the regulative norms and determines the normative protocol.
- Next, it verifies the detected norms, (7) updates its Norms Model base with new regulative norms and normative protocol.

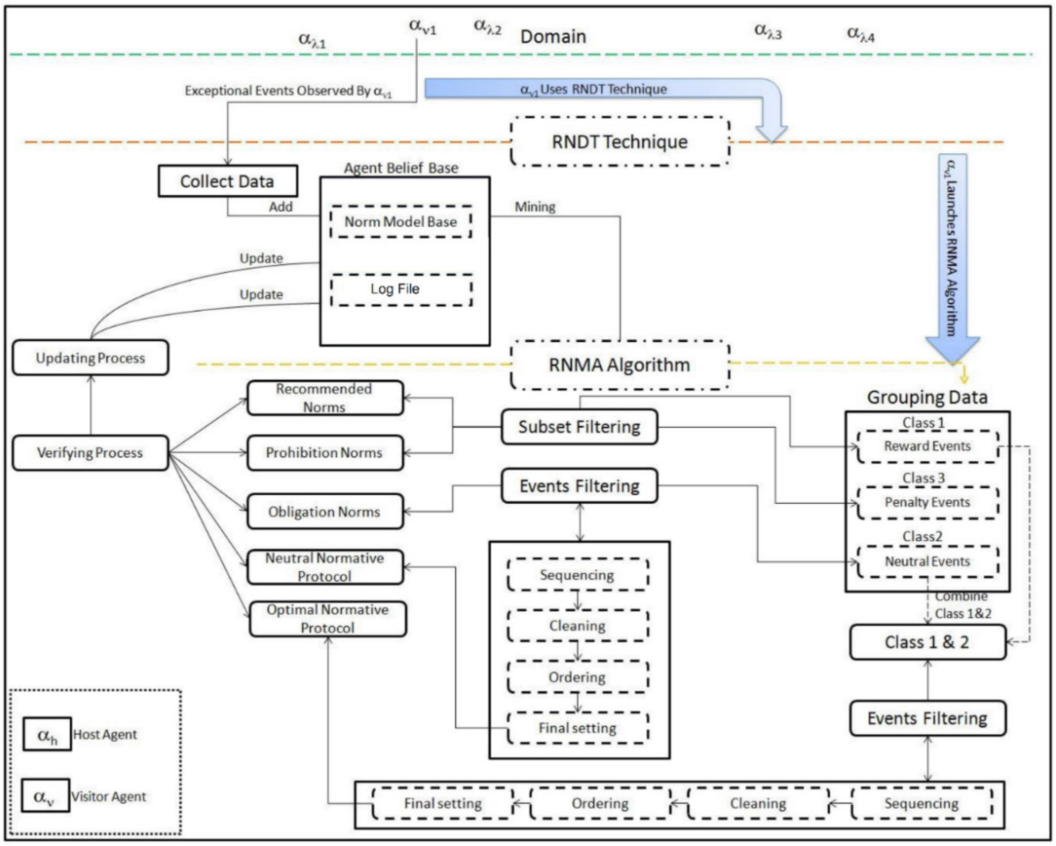


FIGURE 3. The RNDT process flow.

- Any updating made on steps (7) or (8) updates the agent's belief.

Figure 3 shows the complete RNDT process.

IV. THE REGULATIVE NORM MINING ALGORITHM (RNMA)

The following definitions are required to facilitate the presentation of the RNMA. These definitions provide the fundamental idea and theory underlying the norm mining process.

Definition 1: A Convention, CNV, is a type of norm that is expected to be used by every agent in the domain.

Definition 2: A Recommendation Norm, N_{REC} a type of norm that if an agent exercises it, it gets a reward, but if it does not, it is not penalized.

Definition 3: An Obligation Norm, N_{OBL} is a type of norm that if an agent exercises it, it avoids a penalty, but if it does not do so, it is penalized.

Definition 4: A Prohibition Norm, N_{PRO} is a type of norm that if an agent exercises it, it is penalized, but if it does not do so, it avoids the penalty.

Definition 5: All episodes, A_{EPS}, refer to the entire episode of a specific history.

Definition 6: Subset episodes, S_{EPS}, refer to the episodes which have the same sequence of events of superset episodes in a specific history.

To explain the RMNA algorithm process, we present an example of the elevator scenario in detecting the obligation, prohibition, and recommendation norms.

A set of commonly practiced norms in a typical elevator domain are inspired by real practice, which is: (arrive; wait; enter; greet; litter; excuse; depart).

It is also assumed that some host agent is $A_\lambda = \{\alpha_{\lambda 1}, \alpha_{\lambda 2}, \ldots \alpha_{\lambda n}\}$ are penalized, Π, due to performing prohibition norms i.e. litter, or not acting an obligation norm excuse, while others are rewarded, Ω, for acting a recommended action such as greeting. The collected events from other agents $\alpha_{\lambda 1}$ to $\alpha_{\lambda 13}$ are presented as follows,

$(\alpha_{\lambda 1})$	= (enter, excuse, depart)
$(\alpha_{\lambda 2})$	= (arrive, wait, enter, excuse)
$(\alpha_{\lambda 3}, \alpha_\sigma)$	= (wait, enter, litter, excuse, Π)
$(\alpha_{\lambda 4})$	= (arrive, wait, enter, excuse, depart)
$(\alpha_{\lambda 5}, \alpha_\sigma)$	= (arrive, wait, enter, depart, Π)
$(\alpha_{\lambda 6}, \alpha_\sigma)$	= (enter, depart, Π)
$(\alpha_{\lambda 7})$	= (wait, enter, excuse)
$(\alpha_{\lambda 8}, \alpha_\sigma)$	= (arrive, wait, enter, litter, excuse, depart, Π)
$(\alpha_{\lambda 9}, \alpha_\sigma)$	= (wait, enter, litter, Π)
$(\alpha_{\lambda 10}, \alpha_\sigma)$	= (arrive, wait, enter, greet, excuse, depart, Ω)
$(\alpha_{\lambda 11})$	= (arrive, wait, enter)

$$(\alpha_{\lambda 12}, \alpha_{\sigma}) = (\text{wait, enter, depart}, \Pi)$$
$$(\alpha_{\lambda 13}, \alpha_{\sigma}) = (\text{greet, excuse, depart}, \Omega)$$

The RNMA phases implementation are as follows,

Phase 1: Grouping the data, GrDt, of Record Base (RB) into three classes. Class 1, C1, involves the rewarded events. Class 2, C2, groups the neutral events (no rewards or penalties). Class 3, C3, involves penalized events.

If V_Ω are rewarded events; V_Π are penalized events; and V_N are neutral events, then,

$$\begin{aligned} RB &= \{V_\Omega, V_N, V_\Pi\} \\ C_1 &= RB/\{V_N, V_\Pi\} \\ C_2 &= RB/\{V_\Omega, V_\Pi\} \\ C_3 &= RB/\{V_N, V_\Pi\} \\ GrDt &= C_1(V_\Omega), C2(V_N), C3(V_\Pi) \end{aligned} \quad (1)$$

Applying Eq.1 on the episodes of the elevator example:

Classes	Episodes
$C_1(V_\Omega)$ (Class 1)	• (arrive, wait, enter, greet, excuse, depart) • (greet, excuse, depart)
$C_2(V_N)$ (Class 2)	• (enter, excuse, depart) • (arrive, wait, enter, excuse) • (arrive, wait, enter, excuse, depart) • (wait, enter, excuse) • (arrive, wait, enter)
$C_3(V_\Pi)$ (Class 3)	• (wait, enter, litter, excuse) • (arrive, wait, enter, depart) • (enter, depart) • (arrive, wait, enter, litter, excuse, depart) • (wait, enter, litter) • (wait, enter, depart)

Phase 2: Events filtering (E – episode) is applied to the C2 (Class 2), which is the neutral events by applying the following process:

$$E - \text{episode}: \quad Q(C_2) \rightarrow L(C_2) \rightarrow D(C_2) \rightarrow C_{F2} \quad (2)$$

Table 2 shows the filtering process details,

Applying Sequencing $Q(C_2)$ on Class 2 of the elevator scenario:

$Q(C_2) = (\text{enter}_{[1]}, \text{excuse}_{[2]}, \text{depart}_{[3]})$
$(\text{arrive}_{[1]}, \text{wait}_{[2]}, \text{enter}_{[3]}, \text{excuse}_{[4]})$
$(\text{arrive}_{[1]}, \text{wait}_{[2]}, \text{enter}_{[3]}, \text{excuse}_{[4]}, \text{depart}_{[5]})$
$(\text{wait}_{[1]}, \text{enter}_{[2]}, \text{excuse}_{[3]})$
$(\text{arrive}_{[1]}, \text{wait}_{[2]}, \text{enter}_{[3]})$

Applying Filtering $L(C_2)$ on $Q(C_2)$ of the scenario:

$L(C_2) = (\text{enter}_{[3]}, \text{excuse}_{[4]}, \text{depart}_{[5]}, \text{arrive}_{[1]}, \text{wait}_{[2]})$

Applying Ordering $D(C_2)$ on $L(C_2)$ of the scenario:

$D(C_2) = (\text{arrive}_{[1]}, \text{wait}_{[2]}, \text{enter}_{[3]}, \text{excuse}_{[4]}, \text{depart}_{[5]})$

Getting C_{F2} from $D(C_2)$ of the scenario:

$C_{F2} = (\text{arrive, wait, enter, excuse, depart})$

TABLE 2. The filtering process.

Process	Description
Sequencing, Q:	Numbers each episode's event as a sequence from 1 to k (1, 2, 3, 4, ..., k). For example, the episode (a, b, c, d) becomes ($a_{[1]}$, $b_{[2]}$, $c_{[3]}$, $d_{[4]}$).
Cleaning, L:	Select the event, which has the highest sequence number from the repeated events and removing the rest. For example, given these two episodes: ($a_{[1]}$, $b_{[2]}$, $c_{[3]}$, $d_{[4]}$), and ($b_{[1]}$, $d_{[2]}$) we remove $b_{[1]}$ and $d_{[2]}$ retain $b_{[2]}$ and $d_{[4]}$, since they hold the highest sequence number for repeated events.
Ordering, D:	After getting the highest number of sequence for non-repeated set, the set is ordered from 1 to n (counting up). For example, the episode ($b_{[2]}$, $a_{[1]}$, $d_{[4]}$, $c_{[3]}$) becomes ($a_{[1]}$, $b_{[2]}$, $c_{[3]}$, $d_{[4]}$) upon ordering the sequence.
Final setting, C_{F2}	Get the final set of Class 2 ordering to use it in the next steps. For example, the final set of ordering episode ($a_{[1]}$, $b_{[2]}$, $c_{[3]}$, $d_{[4]}$) is (a, b, c, d),

C_{F2}represents the identity class, it can be used to detect all the types of regulative norms and identify the normative protocol of the domain.

Phase 3: Subset filtering (F – episodes) is applied on A – episodes of C_1 and C_3 by removing the S – episodes if the same event sequence exists in both subset and super episodes.

$$\begin{aligned} &F - \text{episodes}\ (C_1) \\ &\quad = A - \text{episodes}\ (C_1) \setminus S - \text{episodes}(C_1) \Longrightarrow C_{F1} \end{aligned} \quad (3)$$
$$\begin{aligned} &F - \text{episodes}\ (C_3) \\ &\quad = A - \text{episodes}\ (C_3) \setminus S - \text{episodes}\ (C_3) \Longrightarrow C_{F3} \end{aligned} \quad (4)$$

The subset filtering removes all the subsets from C_1 and C_3 to get the final set of Class 1 (C_{F1}) and Class 3 (C_{F3}).

This step filters the unnecessary episodes data of Class 1 and Class 3 to determine the candidate episodes and to reduce the cost of extracting the regulative norms. Getting C_{F1} from Class 1 in the scenario (Eq. 3),

A-episodes(C_1)
(arrive, wait, enter, greet, excuse, depart);
(greet, excuse, depart)

S-episodes (C_1)
(greet, excuse, depart)

We obtain, C_{F1} = (arrive, wait, enter, greet, excuse, depart)

Getting C_{F3} from Class 3 in the elevator scenario (Eq. 4):

A-episodes(C_3)
(wait, enter, litter, excuse); (arrive, wait, enter, depart);
(enter, depart); (arrive, wait, enter, litter, excuse, depart);
(wait, enter, litter); (wait, enter, depart)

S-episodes (C_3)

(wait, enter, litter, excuse); (enter, depart);

(wait, enter, litter); (wait, enter, depart)

Phase 4: Extracting the regulative norms. Based on Definition 2, Recommendation norms can be extracted from C_{F1}, Obligation norms can be extracted from C_{F2} (the storage of neutral events, i.e., no rewarded and or penalized events), and Prohibition type norm must appear in C_{F3} only, the extraction process is as shown in Table 3.

Phase 5: Identify the normative protocol, NorProt., As a consequence of the RNMA algorithm, two types of NorProt are identified as shown in Table 4,

V. TESTING AND RESULTS

To validate the developed norm detection technique, we demonstrate a simulation study represented as a scenario of an agent in an elevator domain.

A. SIMULATION MODEL AND SETTINGS

Five testing variables are considered and classified into three categories, Task Condition, the Agent Ability, and Observed Domain. Each category is given the value of Low, Medium or High for testing. Table 5,

Table 6 shows each variable's values of different testing levels (Low, Medium, High).

Two types of agents are simulated in this scenario, host agents, α_λ; and visitor agents, α_ν. The following defines the norms of the elevator domain and assigns each norm type.

Enacted Norms (Normative Protocol, $\aleph_P$): *wait*, *enter*, *greet*, *litter*, *excuse*, *depart.*

Conventions (Γ):	*wait*, *enter*, *depart*
Recommendation (Ω):	*greet*
Obligation norms (Π):	*excuse*
Prohibition Norms (Ψ):	*litter*

B. SIMULATION AND RESULTS

Experiment 1: When the visitor agents are equipped with the RNDT.

- **Test No. 1**: This test determines the rate of norm detection when the visitor agents are equipped with RNDT.

Settings

A_σ:	1 agent	
A_v:	10 agents	
$A_{\lambda S}$:	10 agents	Medium
C_N:	3 cycles	Medium
G_S:	$5 * 5$	Low
O_L:	1 cell	Low
E_O:	50%	Medium

Run No.	Run 1	Run 2	Run 3	Run 4	Run 5	Mean
A_{vD}	7	9	6	9	8	78.0%

The above result shows the success rate of visitor agents in detecting the norms in five runs. The mean rate of norms detection by the visitor agents equals 78%.

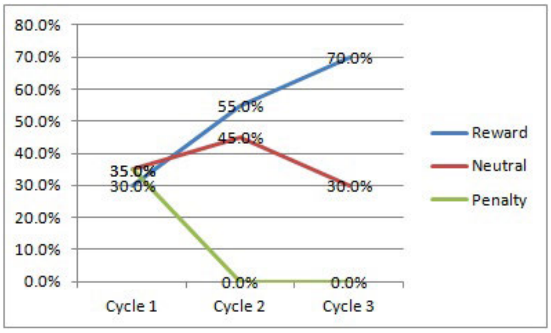


FIGURE 4. The Result of updating norms in each cycle, the rate of rewarded agents increased gradually from 30% to 70% and neutral agents also increased from 35% to 45% while the penalty case decreased from 35% to 0.0%.

- **Test No. 2**: this test demonstrates the updating progress of norms in each cycle

Settings

A_σ:	1 agent	
A_v:	10 agents	
$A_{\lambda S}$:	10 agents	Medium
C_N:	3 cycles	Medium
G_S:	$10 * 10$	Medium
O_L:	1 cell	Low
E_O:	50%	Medium

Cycle No.	Cycle 1	Cycle 2	Cycle 3
Rate of rewarded agents	30.0%	55.0%	70.0%
Rate of neutral agents	35.0%	45.0%	30.0%
Rate of penalized agents	35.0%	0.0%	0.0%

The above results show considerable progress of norms update. The rate of rewarded agents in the three cycles increases gradually from 30% in cycle 1 to 55% in cycle 2 and 70% in cycle 3. The rate of neutral agents also increases in cycle 2 from 35% to 45% but the rate decreases to 30% in cycle 3 due to 15% of the agents manage to get rewards. The most noticeable change is in the penalty case in which 35% of agents are penalized in cycle 1, while in cycle 2 and 3 the rate is maintained at 0.0%. Figure 4 shows the rate of progress when the visitor agents use the RNDT for norm detection.

Figure 5 shows the rate of reward and penalty for each type of agent (host and visitor) in each cycle. The figure clearly shows that the visitor agents have made progress because they are equipped with the RNDT. In cycle 1, the rewarded visitor agents are 0.0% and the number increases to 50% in cycle 2 and 80% in cycle 3. The rate of neutral and penalized visitor agents in cycle 1 is 30% and 70% respectively. In cycle 2, it is 50% and 0.0% and in cycle 3 it is 20% and 0.0% respectively.

Figure 6 shows the rate of detection of the three regulative norms greet (recommendation), excuse (obligation), and litter (prohibition) in each cycle and for each type of agent. In cycle 1, the rate of visitor agents that adopted greet norm is 0.0%, in cycle 2 the rate increases to 50% and in cycle 3 it reaches 80%. The excuse norm is adopted by 70% of visitor

TABLE 3. **The extraction process.**

Norm	Description	Formulation	Elevator Scenario	Discussion
Recommendation	An agent detects a Recommendation norm by finding the set-theoretic complement between the episodes of C_{F1} and the segments from C_{F2} class that have the same first and last event of C_{F1}, $(V_\Omega C_{F2})$.	Extracting the Recommendation Norm, $\text{If } ((C_{F1} \setminus V_\Omega C_{F2}) = \varnothing)$ $\Rightarrow (\text{RcmN} = \varnothing),$ $\text{else } ((C_{F1} \setminus V_\Omega C_{F2}) = \text{Norm})$ (8) $\Rightarrow (\text{Norm} \in \text{RcmN})$ which means that if C_{F1} set without $V_\Omega C_{F2}$ set equals $\varnothing$, then no reward events appear in the C_{F1}, otherwise, the results represent the Recommendation norm.	C_{F1} = (arrive, wait, enter, greet, excuse, depart) C_{F2}= (arrive, wait, enter, excuse, depart) $V_\Omega C_{F2}$ = (arrive, wait, enter, excuse, depart) Therefore,from Eq.8,((arrive, wait, enter, greet, excuse, depart \arrive, wait, enter, excuse, depart) = greet)) $\Rightarrow$ (greet $\in$ RcmN)	The RNMT detects *greet* as a recommendation norm; *excuse* as an obligation norm and *litter* as a prohibition norm. Given the identified regulative norms, the agent then infers the normative protocol for the scenario.
Obligation	An agent detects an Obligation norm by finding the set-theoretic complement between the episodes $V_\Omega C_{F2}$ and C_{F3} that have the same first and last event of $V_\Omega C_{F2}$.	Extracting the obligation norm, $\text{If } ((V_\Omega C_{F2} \setminus C_{F3}) = \varnothing)$ $\Rightarrow (\text{ObligN} = \varnothing),$ $\text{else } ((V_\Omega C_{F2} \setminus C_{F3}) = \text{Norm})$ $\Rightarrow (\text{Norm} \in \text{ObligN})$ which means that if $V_\Omega C_{F2}$ set without C_{F3} set equal $\varnothing$, then no obligation norms appear in the $V_\Omega C_{F2}$, otherwise, the results represent the obligation norms.	Identifying the Obligation norms in the elevator example: C_{F2} = (arrive, wait, enter, excuse, depart) C_{F3} = (arrive, wait, enter, depart) (arrive, wait, enter, litter, excuse, depart) C_{F3} has two episodes • Episode1 C_{F3} = (arrive, wait, enter, depart) C_{F2}= (arrive, wait, enter, excuse, depart) $V_\Omega C_{F2}$= (arrive, wait, enter, excuse, depart) Therefore, from Eq.9, (((arrive, wait, enter, excuse, depart) \arrive, wait, enter, depart) = excuse)) $\Rightarrow$ (excuse $\in \Pi$) • Episode2 C_{F3}= (arrive, wait, enter, litter, excuse, depart) C_{F2}= (arrive, wait, enter, excuse, depart) $V_\Omega C_{F2}$= (arrive, wait, enter, excuse, depart) Therefore, from Eq.9, ((arrive, wait, enter, excuse, depart \arrive, wait, enter, litter, excuse, depart) = $\varnothing$)) $\Rightarrow$ (ObligN = $\varnothing$)	
Prohibition	An agent detects a prohibition norm by finding the set-theoretic complement between each episode in C_{F3} and episode of $V_\Omega C_{F2}$	Extracting the prohibition norm, $\text{If } ((C_{F3} \setminus V_\Omega C_{F2}) = \varnothing)$ $\Rightarrow (\text{ProhbN} = \varnothing),$ $\text{else } ((C_{F3} \setminus V_\Omega C_{F2}) = \text{Norm})$ $\Rightarrow (\text{Norm} \in \text{ProhbN}).$ which means that if C_{F3} set without $V_\Omega C_{F2}$ set equal $\varnothing$, then no prohibition norm appears in the C_{F3}, otherwise, the results represent the prohibition norm of penalized events.	Identifying the Prohibition norms in the elevator example: C_{F2} = (arrive, wait, enter, excuse, depart) C_{F3} = (arrive, wait, enter, depart) (arrive, wait, enter, litter, excuse, depart) C_{F3} has two episodes • Episode1 C_{F3} = (arrive, wait, enter, depart) C_{F2}= (arrive, wait, enter, excuse, depart) $V_\Omega C_{F2}$= (arrive, wait, enter, excuse, depart) Therefore, from Eq.10, ((arrive, wait, enter, depart \arrive, wait, enter, excuse, depart) = $\varnothing$)) $\Rightarrow$ (ProhbN = $\varnothing$) • Episode2 C_{F3} = (arrive, wait, enter, litter, excuse, depart) C_{F2}= (arrive, wait, enter, excuse, depart) $V_\Omega C_{F2}$= (arrive, wait, enter, excuse, depart) Therefore, from Eq.10, ((arrive, wait, enter, litter, excuse, depart \arrive, wait, enter, excuse, depart) = litter)) $\Rightarrow$ (litter $\in$ ProhbN)	

agents in cycle 1 and increases to 100% in cycle 2 and 3. The litter norm is practiced by 60% of the visitor agents in cycle 1 but is reduced to 0.0% in cycle 2 and 3.

Test 3 shows that about 78% of the visitor agents have successfully updated their norms and enhanced their performance by getting rewards or avoiding penalties. In this test, we equipped the visitor agents with the RNDT to prove that after a certain cycle, the visitor agents perform better than the host agents.

Test 4 complements the results of Test 2 by revealing the increasing rate of the visitor agents getting rewards from 30% to 70% and a corresponding decreasing rate of the

TABLE 4. NorProt identification.

Type	Description & Formulation	Elevator Scenario
Neutral Normative Protocol, NuNorProt	A normative protocol that does not contain recommendation and prohibition norms. The agent identifies the NuNorProt from CF2 (neutral episode) because CF2 contains the set of norms that if exercised by the agent, are not rewarded or penalized. $\mathrm{NuNorProt} = \mathrm{C}_{F2}$ (11)	Identifying the Neutral Normative Protocol in the elevator scenario: From Eq.11, $\mathrm{NuNorProt} = \mathrm{C}_{F2} = (\mathrm{arrive}, \mathrm{wait}, \mathrm{enter}, \mathrm{excuse}, \mathrm{depart})$
Optimal Normative Protocol, OptNorProt,	A normative protocol that contains recommendation norms but without prohibition norms. The agent can identify the NuNorProt by filtering events, which is mentioned in Step 2. Following the complement of C_{F1} and C_{F2} events filtering, sequencing, filtering, and ordering are performed. Then the final set represents the set of norms that are rewarded. $(\mathrm{C}_{F1}, \mathrm{C}_{F2}) \longrightarrow L(\mathrm{C}_{F1}, \mathrm{C}_{F2}) \longrightarrow D(\mathrm{C}_{F1}, \mathrm{C}_{F2}) \longrightarrow \mathrm{OptNorProt}$ (12)	Identifying the Optimal Normative Protocol in the elevator scenario: From Eq.12, C_{F1} = (arrive, wait, enter, greet, excuse, depart) C_{F2} = (arrive, wait, enter, excuse, depart) • Sequencing, Q (arrive$_{[1]}$, wait$_{[2]}$, enter$_{[3]}$, greet$_{[4]}$, excuse$_{[5]}$, depart$_{[6]}$) (arrive$_{[1]}$, wait$_{[2]}$, enter$_{[3]}$, excuse$_{[4]}$, depart$_{[5]}$) • Filtering, L (arrive$_{[1]}$, wait$_{[2]}$, enter$_{[3]}$, greet$_{[4]}$, excuse$_{[5]}$, depart$_{[6]}$) • Ordering, D (arrive$_{[1]}$, wait$_{[2]}$, enter$_{[3]}$, greet$_{[4]}$, excuse$_{[5]}$, depart$_{[6]}$) • Final setting OptNorProt = (arrive, wait, enter, greet, excuse, depart) predefined structure

TABLE 5. Variables' description.

Category	Variable	Description
Task Condition	Cycle Time	It is defined as the time given to a number of event cycle for the domain. For example, in the elevator domain, the event cycle could be wait, enter, excuse, depart. Each host agent enacts these events. A visitor agent observes and learns those events in a number of cycles, which could be one cycle or more.
Agent Ability	Observation Limit	Each visitor agent has a limit of observation to monitor the host agents' behaviors. This study assumes that the visitor agents in the grid are able to observe the surrounding host agents within the observation threshold limit. The unit of measuring the limit is a cell of the grid in the North, South, East and West direction, i.e., a visitor agent is able to observe the host agents located one cell beside it. A mathematical model is developed to determine the visible cells for nearby host agents.
	Grid Size	The domain is simulated as a two-dimensional grid and the grid size represents the domain size. Figure 3 shows an example of a 10*10 grid with host and visitor agents randomly distributed within the gird.
Domain	Population Density	This is the number of Host Agents in the grid. If there are many agents, then the density is considered high.
	Occurrence Rate of Exceptional Events	This variable determines the frequency of exceptional events happening in the domain.

agents getting penalties from 35% to 0.0% within the three cycles. Figure 5 clearly shows that the visitor agents enhanced their performance until they become more norms-compliant than the host agents. In cycle 1, 0.0% of the visitor agents are rewarded, while in cycle 2, 50% and in cycle 3, 80%. However, this result can be more or less based on environmental variables settings.

Experiment 2: In this experiment, the environmental variables' effects on norm detection by the RNDT are observed. The variables are Observation Limit ($\mathrm{O_L}$), Grid Size ($\mathrm{G_S}$),

TABLE 6. **Variables' values.**

Variables	Low	Medium	High
Observation Limit (O_L)	1 Cell	2 Cells	3 Cells
Grid (Domain) Size (G_S)	5*5	10*10	20*20
Cycle Time (C_N)	2 Cycles	3 Cycles	4 Cycles
Occurrence of Exceptional Events (E_O)	25%	50%	75%
Population Density of Domain Agents ($A_{\lambda S}$)	5 Agents	10 Agents	20 Agents

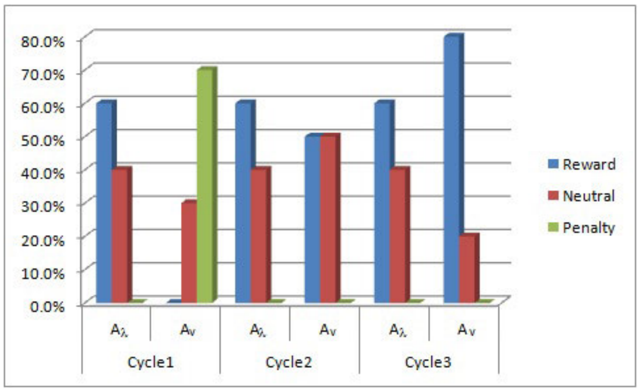


FIGURE 5. **The rate of rewarded, neutral and penalized agents in each cycle and for each type of agents (host and visitor) when they are equipped with RNDT.**

Cycle Time (C_N), Occurrence of Exceptional Events (E_O), Population Density of Host Agents ($A_{\lambda S}$).

- **Test No. 3**: This test measures the effect of Exceptional events Occurrence (E_O) on norm detection.

Settings

A_σ:	1 agent	
A_v:	10 agents	
$A_{\lambda S}$:	5 agents	Low
C_N:	2 cycles	Low
G_S:	$20 * 20$	High
O_L:	1 cell	Low
E_O:	N%	(Low, Medium, High)

Run No.	Run 1	Run 2	Run 3	Run 4	Run 5	Mean
N =25% - Low						
A_{vD}	1	3	3	2	1	20.0%

Run No.	Run 1	Run 2	Run 3	Run 4	Run 5	Mean
N =50% - Medium						
A_{vD}	1	3	3	3	4	28.0%

Run No.	Run 1	Run 2	Run 3	Run 4	Run 5	Mean
N =75% - High						
A_{vD}	4	3	4	3	2	32.0%

Figure 7 shows the effect of Exceptional events Occurrence (E_O) on norm detection. The rate of detection increases from 20% when E_O is Low to 28% when E_O is Medium and reaches 32% when E_O is High.

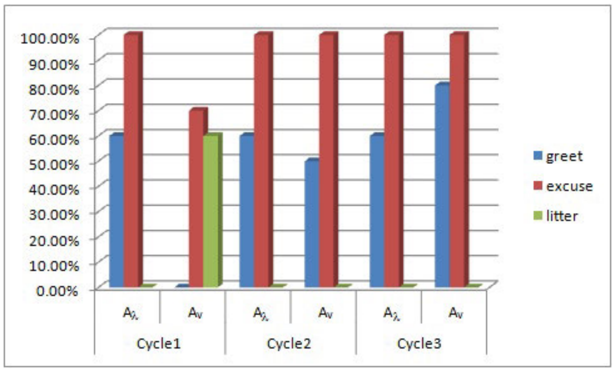


FIGURE 6. **The rate of occurrence of the three regulative norms greet (recommendation, increased from 0% to 80%), excuse (obligation, increased from 70% to 100%), and litter (prohibition, decreased from 60% to 0%).**

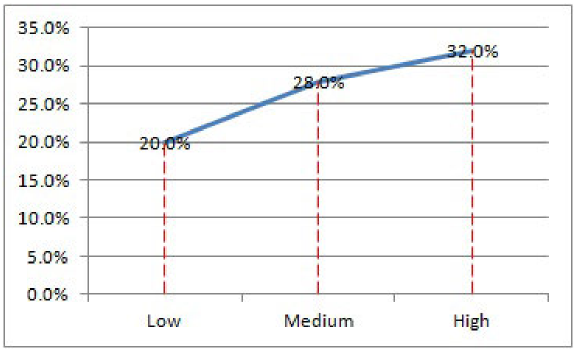


FIGURE 7. **The effect of exceptional events occurrence (E_O) on norm detection, the rate of detection increased from 20% when E_O is low to 28% when E_O is medium and reached 32% when E_O is high.**

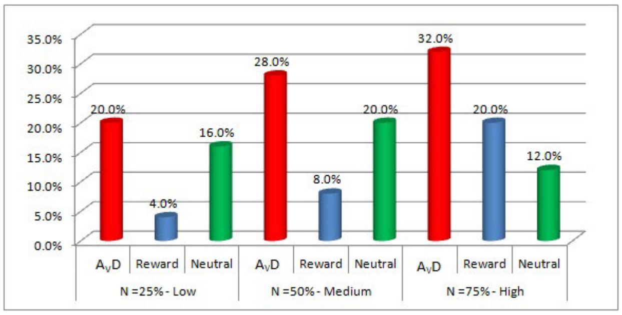


FIGURE 8. **The rate of rewarded and neutral visitor agents of exceptional events (E_O) test, in the low case, the total A_{vD} is 20%, in the medium case, the total A_{vD} is 28%, in the high case, the total A_{vD} is 32%.**

Figure 8 shows the rate of rewarded and neutral visitor agents in each case (Low, Medium, High) with regard to Exceptional events Occurrence (E_O). In the Low case, the total A_{vD} is 20%, which comprise of 4% that manage to get rewards and 16% manage to avoid penalties. While in the Medium case, the total A_{vD} is 28%, which comprise of 8% that manage to get rewards and 20% manage to avoid penalties. The best results are shown in the High case, in which the

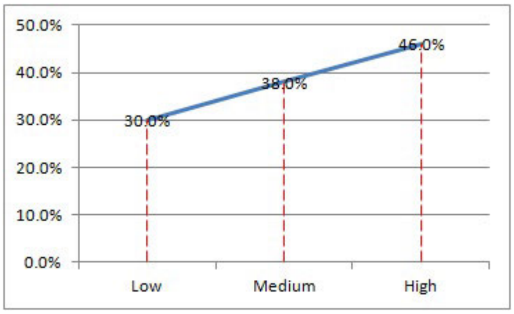


FIGURE 9. Observation limit (O_L) test, the rate of detection increased from 30% to 46% (Low: 30%, Medium: 38%, High: 46%).

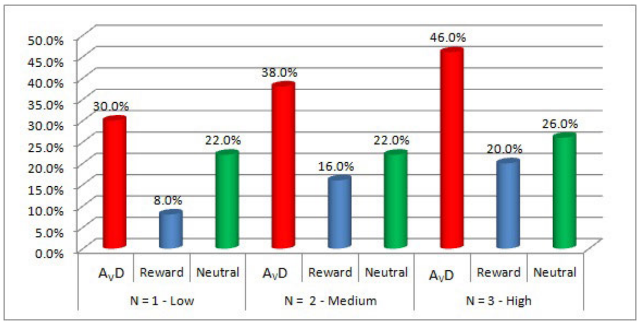


FIGURE 10. The rate of rewarded and neutral visitor agents of observation limit (O_L) test, in the Low case, the total A_{vD} is 30%, in the Medium case, the total A_{vD} is 38% in the High case, the total A_{vD} is 46%.

total A_{vD} is 32%, which comprises 20% that manage to get rewards and 12% manage to avoid penalties.

- This test measures the effect of Observation Limit (O_L) on norm detection.

Settings

A_σ:	1 agent	
A_v:	10 agents	
$A_{\lambda S}$:	5 agents	Low
C_N:	2 cycles	Low
G_S:	$20 * 20$	High
O_L:	N cell	(Low, Medium, High)
E_O:	50%	Medium

Run No.	Run 1	Run 2	Run 3	Run 4	Run 5	Mean
N =1 - Low						
A_{vD}	2	3	3	4	3	30.0%

Run No.	Run 1	Run 2	Run 3	Run 4	Run 5	Mean
N =2 - Medium						
A_{vD}	3	5	4	4	3	38.0%

Run No.	Run 1	Run 2	Run 3	Run 4	Run 5	Mean
N =3- High						
A_{vD}	6	6	3	5	3	46.0%

Figure 9 shows the effect of Observation Limit (O_L) on norm detection. The rate of detection increased from 30% when O_L is Low to 38% when O_L is Medium and reaches 46% when O_L is High.

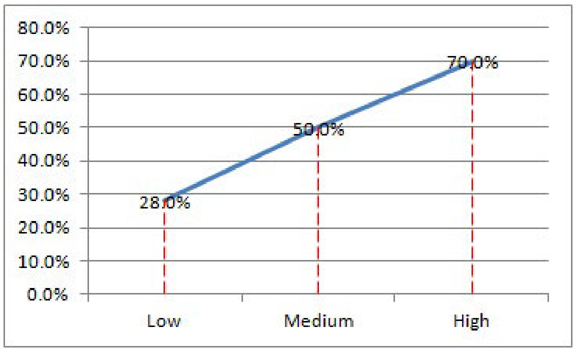


FIGURE 11. Cycle Time (C_N) test, the rate of detection increased from 28% when C_N is Low to 50% when it is Medium and reaches 70% when it is High.

Figure 10 shows the rate of rewarded and neutral visitor agents in each case (Low, Medium, High) with regard to Observation Limit (O_L). In the Low case, the total A_{vD} is 30%, which comprise of 8% that manage to get rewards and 22% that manage to avoid penalties. While in the Medium case, the total A_{vD} is 38% which comprises 16% that manage to get rewards and 22% that manage to avoid penalties. The best results are demonstrated in the High case, in which the total A_{vD} is 46% which comprises 20% that manage to get rewards and 26% that manage to avoid penalties.

- **Test No. 5**: This test measures the effect of Cycle Time (C_N) on norm detection.

Settings

A_σ:	1 agent	
A_v:	10 agents	
$A_{\lambda S}$:	5 agents	Low
C_N:	N cycles	(Low, Medium, High)
G_S:	$20 * 20$	High
O_L:	1 cell	Low
E_O:	50%	Medium

Run No.	Run 1	Run 2	Run 3	Run 4	Run 5	Mean
N =2 - Low						
A_{vD}	3	3	2	3	3	28.0%

Run No.	Run 1	Run 2	Run 3	Run 4	Run 5	Mean
N =3 - Medium						
A_{vD}	4	6	6	5	4	50.0%

Run No.	Run 1	Run 2	Run 3	Run 4	Run 5	Mean
N =4- High						
A_{vD}	9	7	7	5	7	70.0%

Figure 11 shows the impact of C_N. The rate of detection increases from 28% when C_N is Low to 50% when C_N is Medium and reaches 70% when C_N is High.

Figure 12 shows the rate of rewarded and neutral visitor agents in each case (Low, Medium, High) with regard to Cycle Time (C_N). In the Low case, the total A_{vD} is 28% which comprise of 6% that manage to get rewards and 22% that manage to avoid penalties. While in the Medium case, the total A_{vD} is 50% which comprises 30% that manage to

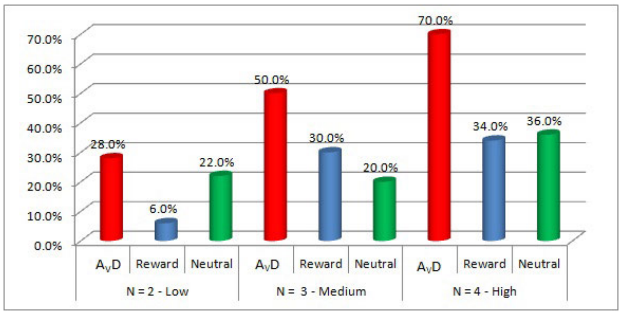


FIGURE 12. **The rate of rewarded and neutral visitor agents of cycle time (C_N) test, in the Low case, the total A_{vD} is 28%, in the Medium case, the total A_{vD} is 50%, in the High case, the total A_{vD} is 70%.**

get rewards and 20% that manage to avoid penalties. The best results are demonstrated in the High case, in which the total A_{vD} is 70% which comprises 34% that manage to get rewards and 36% that manage to avoid penalties.

- **Test No. 6**: This test measures the effect of the Population density of Host Agents ($A_{\lambda S}$) on norm detection.

Settings

A_σ:	1 agent	
A_v:	10 agents	
$A_{\lambda S}$:	N agents	(Low, Medium, High)
C_N:	2 cycles	Low
G_S:	20 ∗ 20	High
O_L:	1 cell	Low
E_O:	50%	Medium

Run No.	Run 1	Run 2	Run 3	Run 4	Run 5	Mean
N =5 - Low						
A_{vD}	4	2	3	2	3	28.0%

Run No.	Run 1	Run 2	Run 3	Run 4	Run 5	Mean
N =10 - Medium						
A_{vD}	3	5	4	4	3	38.0%

Run No.	Run 1	Run 2	Run 3	Run 4	Run 5	Mean
N =20- High						
A_{vD}	6	9	7	4	7	66.0%

Figure 13 shows the impact of Population density on norms detection. The rate of detection increases from 28% when $A_{\lambda S}$ is Low to 38% when $A_{\lambda S}$ is Medium and reaches 66% when $A_{\lambda S}$ is High.

Figure 14 shows the rate of rewarded and neutral visitor agents in each case (Low, Medium, High) with regard to the Population density of Host Agents ($A_{\lambda S}$). In the Low case, the total A_{vD} is 28% which comprise of 8% that manage to get rewards and 20% that manage to avoid penalties. While in the Medium case, the total A_{vD} is 38% which comprises 14% that manage to get rewards and 24% that manage to avoid penalties. The best results are demonstrated in the High case, in which the total A_{vD} is 66% which comprise of 56% that manage to get rewards and 10% that manage to avoid penalties.

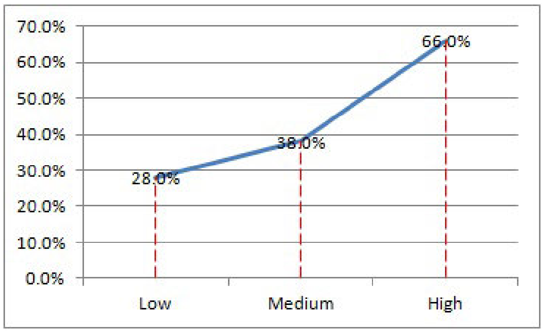


FIGURE 13. **Population density of Host Agents ($A_{\lambda S}$) test, the rate of detection increases from 28% and reached 66% when $A_{\lambda S}$ is High.**

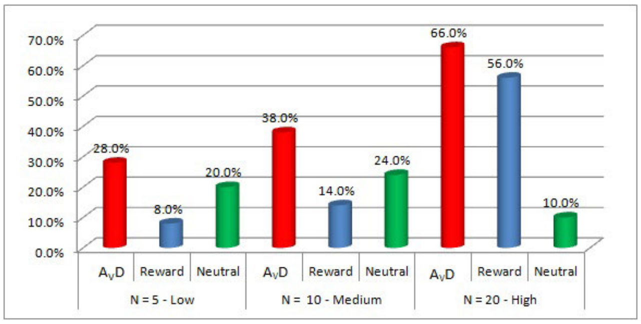


FIGURE 14. **The rate of rewarded and neutral visitor agents of population density of Host Agents ($A_{\lambda S}$) test, in the Low case, the total A_{vD} is 28%, in the Medium case, the total A_{vD} is 38%, in the High case, the total A_{vD} is 66%.**

- **Test No. 7**: This test measures the effect of Grid Size (G_S) on norm detection.

Settings

A_σ:	1 agent	
A_v:	10 agents	
$A_{\lambda S}$:	5 agents	Low
C_N:	2 cycles	Low
G_S:	N ∗ M	(Low, Medium, High)
O_L:	1 cell	Low
E_O:	50%	Medium

Run No.	Run 1	Run 2	Run 3	Run 4	Run 5	Mean
N =5*5 - Low						
A_{vD}	7	8	9	8	6	76.0%

Run No.	Run 1	Run 2	Run 3	Run 4	Run 5	Mean
N =10*10 - Medium						
A_{vD}	4	7	7	5	6	58.0%

Run No.	Run 1	Run 2	Run 3	Run 4	Run 5	Mean
N =20*20- High						
A_{vD}	3	4	3	4	2	32.0%

Figure 15 shows the effect of Grid Size (G_S) on norm detection. The rate of detection decreases from 76% when G_S is Low to 58% when G_S is Medium and reaches 32% when G_S is High. Consequently, Grid Size, G_S, is inversely proportional to norms detection.

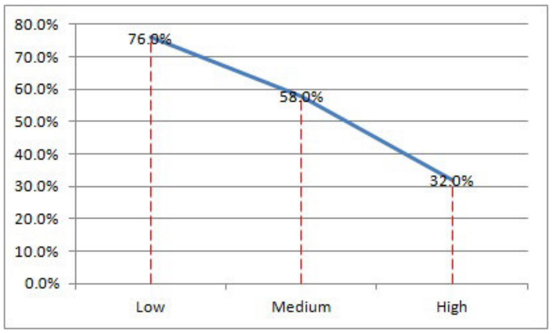


FIGURE 15. Grid size (G_S) test, the rate of detection decreases from 76% when G_S is Low to 58% when G_S is Medium and reaches 32% when G_S is High.

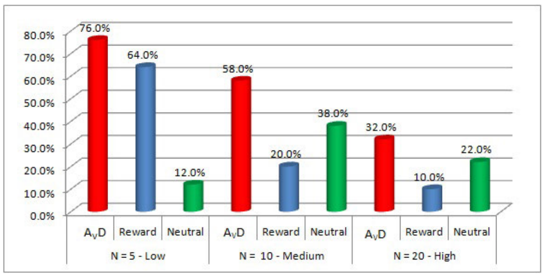
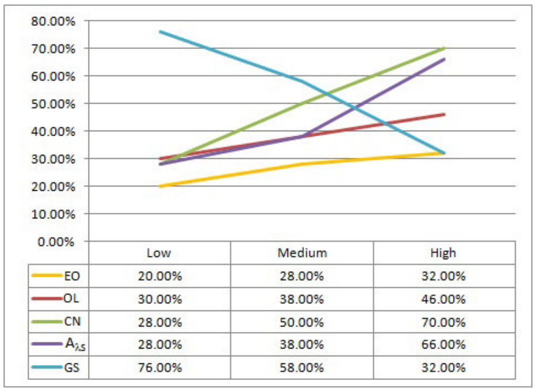


FIGURE 16. The rate of rewarded and neutral visitor agents of grid size (G_S) test, in the Low case the total A_{vD} is 76%, in the Medium case the total A_VD is 58%, in the High case, the total A_VD is 32%.



	Low	Medium	High
EO	20.00%	28.00%	32.00%
OL	30.00%	38.00%	46.00%
CN	28.00%	50.00%	70.00%
$A_{\lambda S}$	28.00%	38.00%	66.00%
GS	76.00%	58.00%	32.00%

FIGURE 17. Rate of norms detection among variables G_S (Low: 76%), $A_{\lambda S}$ (High: 66%), C_N (High: 70%), O_L (High: 46%), E_O (High: 32%).

Figure 16 shows the rate of rewarded and neutral visitor agents in each case (Low, Medium, High) with regard to Grid Size (G_S). The best results are demonstrated in the Low case, in which the total A_{vD} is 76% which comprise of 64% that manage to get rewards and 12% that manage to avoid penalties. While in the Medium case, the total A_VD is 58% which comprises 20% that manage to get rewards and 38% that manage to avoid penalties. In the High case, the total A_VD is 32% which comprise of 10% that manage to get rewards and 22% that manage to avoid penalties.

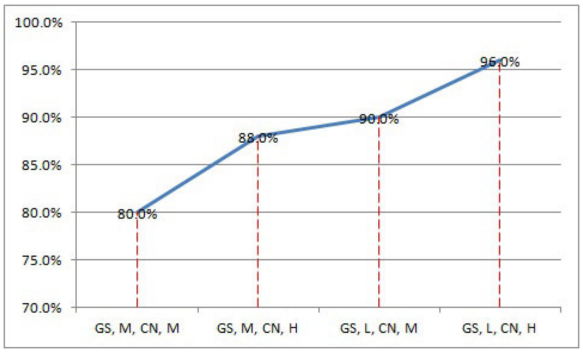


FIGURE 18. Grid size and cycle time test (G_S& C_N Medium): 80%, (G_S Medium & C_N High): 88%, (G_S Low & C_N Medium): 90%, (G_S Low & C_N High): 96%.

Figure 17 reveals some interesting results.

- **Test No. 8**: This test measures the effect of Grid Size, $G_{S,}$ and Cycle Time, C_N, on norms detection.

Settings

A_σ:	1 agent	
A_v:	10 agents	
$A_{\lambda S}$:	5 agents	Low
C_N:	X cycles	Test (Medium, High)
G_S:	$N * M$	Test (Low, Medium)
O_L:	1 cell	Low
E_O:	50%	Medium

Run No.	Run 1	Run 2	Run 3	Run 4	Run 5	Mean
M*N=10*10 – Medium (M), X =3 Cycles – Medium (M),						
A_{vD}	9	7	9	8	7	80.0%

Run No.	Run 1	Run 2	Run 3	Run 4	Run 5	Mean
M*N=10*10 – Medium (M), X =4 Cycles – High (H)						
A_{vD}	7	10	9	10	8	88.0%

Run No.	Run 1	Run 2	Run 3	Run 4	Run 5	Mean
M*N=5*5 – Low (L), X =3 Cycles – Medium						
A_{vD}	10	7	9	10	9	90.0%

Run No.	Run 1	Run 2	Run 3	Run 4	Run 5	Mean
M*N=5*5 – Low (L), X =4 Cycles – High (H)						
A_{vD}	10	9	10	10	9	96.0%

Figure 18 shows the effect of Grid Size, G_S, and Cycle Time, C_N, on norms detection. The rate increases from 80% when G_S and C_N are Medium to 88% when G_S is Medium and C_N is High and reaches 90% when G_S is Low and C_N is Medium. But the best result is 96% when G_S is Low and C_N is High.

Figure 19 shows the rate of rewarded and neutral visitor agents in each of the cases (G_S & C_N Medium; G_S Medium & C_N High; G_S Low & C_N Medium; G_S Low & C_N High). The best result is obtained when G_S is Low and C_N is High, in which the total A_{vD} is 96% and all of them manage to get rewards. The rest of the results are shown in Figure 32. The results clearly show that when there are two variables of strong positive effect, the agent's performance improves rapidly. In addition, the rewarded visitor agents reach the highest level and almost equal the total of A_{vD}.

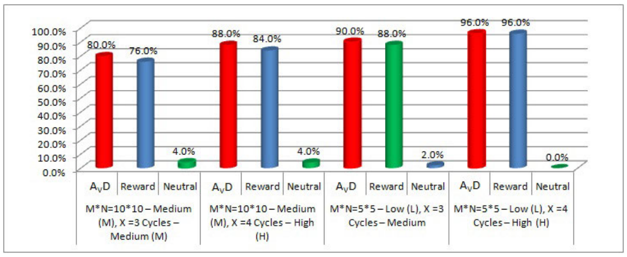


FIGURE 19. The rate of rewarded and neutral visitor agents (G_S& C_N Medium; G_S Medium & C_N High; G_S Low & C_N Medium; G_S Low& C_N High).

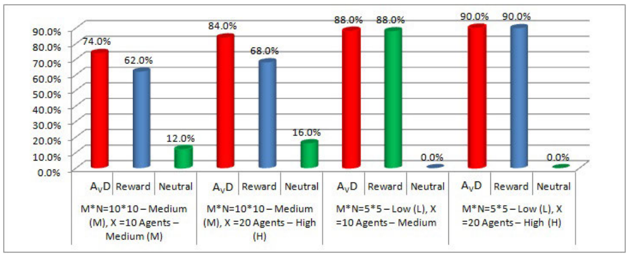


FIGURE 21. The rate of rewarded and neutral visitor agents (G_S & $A_{\lambda S}$ Medium; G_S Medium & $A_{\lambda S}$ High; G_S Low & $A_{\lambda S}$ Medium; G_S Low & $A_{\lambda S}$ High).

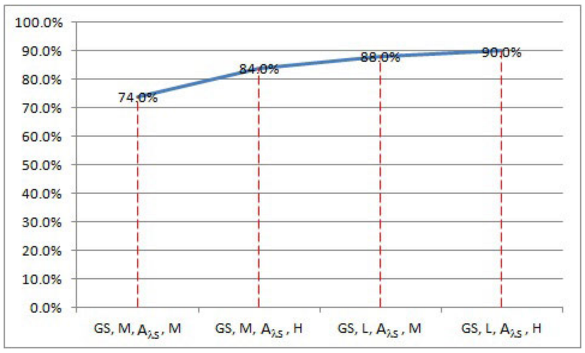


FIGURE 20. Grid size and population density test (G_S & $A_{\lambda S}$ Medium): 74%, (G_S Medium & $A_{\lambda S}$ High): 84%, (G_S Low & $A_{\lambda S}$ Medium): 88%, (G_S Low & $A_{\lambda S}$ High): 90%.

- **Test No. 9**: This test measures the effect of Grid Size, G_S, and Population Density of Host Agents, $A_{\lambda S}$, on norms detection.

Settings

A_σ:	1 agent	
A_v:	10 agents	
$A_{\lambda S}$:	X agents	Test (Medium, High)
C_N:	2 cycles	Low
G_S:	N ∗ M	Test (Low, Medium)
O_L:	1 cell	Low
E_O:	50%	Medium

Run No.	Run 1	Run 2	Run 3	Run 4	Run 5	Mean
M*N=10*10 – Medium (M), X =10 Agents– Medium (M),						
A_{vD}	8	7	7	9	6	74.0%

Run No.	Run 1	Run 2	Run 3	Run 4	Run 5	Mean
M*N=10*10 – Medium (M), X =20 Agents – High (H)						
A_{vD}	10	9	9	8	6	84.0%

Run No.	Run 1	Run 2	Run 3	Run 4	Run 5	Mean
M*N=5*5 – Low (L), X =10 Agents – Medium						
A_{vD}	8	10	9	7	10	88.0%

Run No.	Run 1	Run 2	Run 3	Run 4	Run 5	Mean
M*N=5*5 – Low (L), X = 20 Agents – High (H)						
A_{vD}	9	8	10	8	10	90.0%

Figure 20 shows the effect of Grid Size, G_S, and Population Density of Host Agents, $A_{\lambda S}$, on norms detection. The rate increases from 74% when G_S and $A_{\lambda S}$ are Medium to 84% when G_S is Medium and $A_{\lambda S}$ is High and reaches 88% when G_S is Low and $A_{\lambda S}$ is Medium. But the best result is 90% when G_S is Low and $A_{\lambda S}$ is High.

Figure 21 shows the rate of rewarded and neutral visitor agents in each of the cases (G_S & $A_{\lambda S}$ Medium; G_S Medium & $A_{\lambda S}$ High; G_S Low & $A_{\lambda S}$ Medium; G_S Low & $A_{\lambda S}$ High). The best result is obtained when G_S is Low and $A_{\lambda S}$ is High, in which the total A_{vD} is 90% and all of them managed to get rewards The rest of the results are shown in Figure 34.

- **Test No. 10**: This test measures the effect of Domain (Grid) Size, G_S, Cycle Time, C_N, and Population Density of Host Agents, $A_{\lambda S}$, on norms detection.

Settings

A_σ:	1 agent	
A_v:	10 agents	
$A_{\lambda S}$:	Y agents	Test (Medium, High)
C_N:	X cycles	Test (Medium, High)
G_S:	N ∗ M	Test (Low, Medium)
O_L:	1 cell	Low
E_O:	50%	Medium

Run No.	Run 1	Run 2	Run 3	Run 4	Run 5	Mean
M*N=10*10 – Medium (M), X =3 Cycles – Medium (M), Y = 10 Agents – Medium (M)						
A_{vD}	8	7	8	10	9	84.0%

Run No.	Run 1	Run 2	Run 3	Run 4	Run 5	Mean
M*N=5*5– Low (L), X =4 Cycles – High (H), Y = 20 Agents – High (H)						
A_{vD}	10	10	9	10	9	96.0%

Figure 22 shows the effect of Domain (Grid) Size, G_S, Cycle Time, C_N, and Population Density of Host Agents, $A_{\lambda S}$, on norms detection. The rate increases from 84% when G_S, $A_{\lambda S}$, and C_N are Medium to 96% when G_S is Low, and $A_{\lambda S}$ and C_N are High.

Figure 23 shows the rate of rewarded and neutral visitor agents in each of the cases (G_S, $A_{\lambda S}$, C_N Medium; G_SLow; $A_{\lambda S}$, C_N High). The best results are obtained when G_S is Low, $A_{\lambda S}$ and C_N are High, in which the total A_{vD} is 96% and all of them managed to get rewards. When G_S, $A_{\lambda S}$, and C_N are Medium, the total A_{vD} is 84% and all of them managed to get rewards.

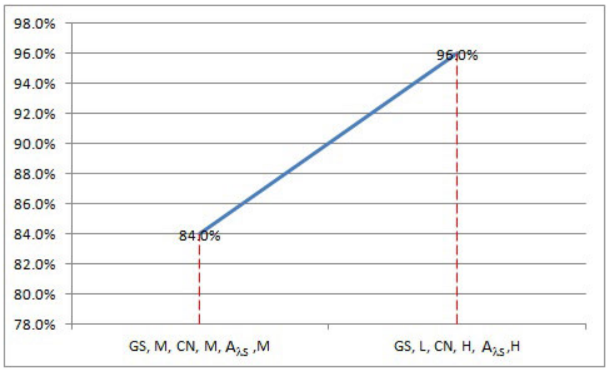


FIGURE 22. Grid Size, Cycle Time and Population Density Test (G_S & C_N & $A_{\lambda S}$) Medium: 84%, (G_S Low & C_N High & $A_{\lambda S}$ High): 96%.

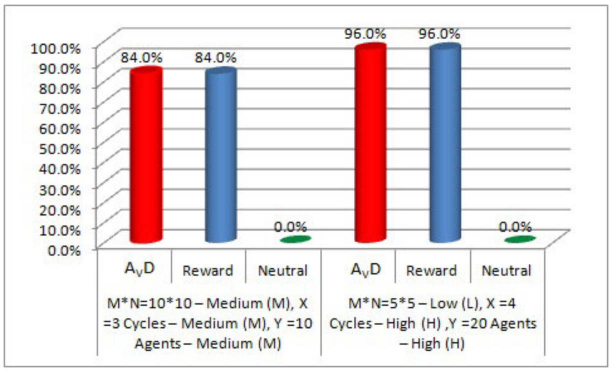


FIGURE 23. The rate of rewarded and neutral visitor agents in each case ((G_S & C_N & $A_{\lambda S}$ Medium; G_S Low & C_N High & $A_{\lambda S}$ High).

Tests 10, 11, and 12 show that when two or more variables are set in a strong positive effect, a high rate of visitor agents succeed in detecting the domain norms and high rate succeeds in gaining rewards.

VI. ANALYSIS OF THE RESULTS

In this simulation, we initially test the scenario when the visitor agents are equipped with the RNDT. The result shows that the rate of adopting the domain's norms is 78.0% of visitor agents succeeded in detecting the domain norms.

In Test 2 we demonstrated the updating progress of norms in each cycle, The rate of rewarded agents in the three cycles increased from 30% to 70%, the rate of neutral agents also increased from 35% to 45%. The most noticeable change is in the penalty case in which 35% of agents are penalized in cycle 1, while in cycle 2 and 3 the rate is maintained at 0.0%. Consequently, the visitor agents become more norms-compliant than the host agents. However, the optimal result of norms detection by exploiting the RNDT depends on the values of the environment variables.

In favorable environments, the RNDT is exploited to optimally detect the norms. For example, a small grid size is favorable for an agent to observe nearby host agents. Therefore, we conduct several experiments to study the cases that offer optimal results and those that do not meet the acceptable rate of detection.

Our objectives from conducting these experiments are to study the effect of the environmental variables settings on the rate of norm detection by a visitor agent and to discover the significant variables that can strongly improve the detection rate hence qualifying the agent to reward level. We set out the output of the experiment as follows:

- Using high positive effect variables: The variable Grid Size, G_S, (76%) is the most effective one followed by the Cycle Time, C_N, (70%) and Population Density of Host Agents, $A_{\lambda S}$, (66%). On the other hand, the Observation Limit, O_L, (46%) is a low positive effect variable while Exceptional Events, $E_{O,}$ (32%) is the lowest positive effect variable.
- The optimal environment entails at least two significant high positive effect variables. Consequently, we conduct experiments based on the combination of these significant variables: G_S, C_N, and $A_{\lambda S}$. The results show that the detection rate increased rapidly when two variables are in high positive effect: G_S (Low), C_N (High), $A_{\lambda S}$ (High) (96%); G_S (Low), C_N (High) (96%); G_S (Low), $A_{\lambda S}$ (High) (90%).
- Generally, the minimum requirements to have an optimal environment for detecting a domain's norms are when the Grid Size is Low and the Cycle Time is High or when Grid Size is Low and Population Density of Host Agents is High. These settings help an agent to collect enough events for analysis.

However, tests of norm detection show considerable results and interesting observations. The RNDT technique shows satisfactory results with minimum settings of the environmental variables. It also shows accurate results when determining the moral status of each detected norm by exploiting of the reward and penalty actions. But it still depends on the availability of sanctions by a third party in the domain. It is therefore considered somewhat inferior in that situation.

The developed technique could be exploited in different domains to adapt norms such as Public (Restaurant, Café, Mall, Park), Private (home), working (Organization, Institutions), Games through maximizing the utility of agents.

VII. DISCUSSION

In this section, we discuss the results of the elevator scenario and theoretically compare the PNI [26], ONI [27] and RNDT. Although the given scenario is not that complex, it does illustrate the challenges of regulative norms detection technique and the potential solution that could be applied via the regulative norms mining algorithm.

From the elevator scenario, we observe the following:

- The algorithm is not based on the association rule mining to filter the events. Instead, it filters based on the Set Theory. From our perspective, this is the most important part to overcome the problem of filtering based on the frequency of occurrences, which may omit some norms if they do not have high enough frequency to be candidate norms.

- The technique of removing the subsets does not omit any norms with any frequency under any situation. Basically, removing the subset of another set means we are removing some cases that are still available for testing. The case will not be omitted by removing the subset because it stills within the set.
- It is clearly shown that the algorithm succeeded in identifying the three types of regulative norms: recommendation (greet), obligation (excuse), and prohibition (litter). This success makes the usage of the algorithm very useful and can be used to detect the regulative norms in general.
- The example shows that the algorithm can be exploited for both reward and penalty actions to detect the regulative norms.
- The algorithm succeeded in identifying multiple norms; the example shows the algorithm detected three norms of the three types.
- The algorithm identifies two types of normative protocol in the example: neutral and optimal. This gives the agent the ability it needs to improve its performance by using the optimal protocol.
- As for the verification process, the example shows that the agent can verify the norms by referring to the Moral Base without the need to ask other agents. This is more useful in public domains to avoid any wrong information.
- From the above points and our reading of the literature, the RNDT is the first and novel technique that identifies the three types of regulative norms and two types of normative protocols while exploiting both reward and penalty actions.

Table 7 shows a comparison between PNI, ONI, and RNDT.

VIII. CONCLUSION AND FUTURE WORK

This paper presented a new mechanism to detect regulative norms and normative protocol of a society called RNDT. It enables the robot community and software agents to adapt to the different domains and assimilate new behaviors to improve performance.

Several tests have been conducted on agents exploiting the technique to detect norms in a virtual domain under varying environmental settings. To validate the technique, a virtual elevator scenario is created and several experiments are conducted. When agents are equipped with the RNDT, a high proportion of agents succeeded in detecting the domain's norms and identifying the normative protocol, the agents managed to get rewards and avoid penalties. However, the environmental variables settings affect the success of detection and the significant variables are Grid Size, Cycle Time and Population Density of Host Agents. When there are two significant variables in a strong positive effect, they offer optimal environment for norm detection. Briefly, the results show that the RNDT can offer immediate effect on norms detection and updating because the technique exploits sanctions to detect the norms.

TABLE 7. Comparison between PNI, ONI, RNDT.

#	Comparison Points	PNI [26]	ONI [27]	RNDT
1	Identify Obligation Norms	No	Yes	Yes
2	Identify Prohibition Norms	Yes	No	Yes
3	Identify Recommendation Norms	No	No	Yes
4	Identify regulative norms in general	No	No	Yes
5	Exploit penalty action to identify the norms	Yes	Yes	Yes
6	Exploit reward action to identify the norms	No	No	Yes
7	Exploit both reward and penalty actions to identify the norms	No	No	Yes
8	Can identify the norms with any occurrence	No	No	Yes
9	Can identify multiple norms	Not tested	Not tested	Yes
10	Can identify neutral normative protocol	Yes	Yes	Yes
11	Can identify optimal normative protocol	No	No	Yes
12	Can identify both optimal and neutral normative protocols	No	No	Yes
13	Verification process guaranteed in public domains	No	No	Yes

The results showed that the technique performs satisfactorily. The RNDT succeeded in detecting the regulative norms in general. It overcomes the problem of using the frequency of occurrences in detecting the candidate norms.

For the future work, we shall study the cost of adopting the detected norms and the ability of an agent to cover this cost, this concept is called norm assimilation. In which an agent may detect a norm "tipping" but it is not necessarily that the agent would adopt the norm. We shall investigate how the decision should be made.

REFERENCES

[1] H. Aldewereld, V. Dignum, and W. W. Vasconcelos, "Group norms for multi-agent organisations," *ACM Trans. Auton. Adapt. Syst.*, vol. 11, no. 2, 2016, Art. no. 15.

[2] M. A. Mahmoud, M. S. Ahmad, M. Z. M. Yusoff, and A. Mustapha, "Norms assimilation in heterogeneous agent community," in *Proc. Int. Conf. Princ. Pract. Multi-Agent Syst.* Cham, Switzerland: Springer, Dec. 2014, pp. 311–318.

[3] M. Alberti, A. S. Gomes, R. Gonçalves, J. Leite, and M. Slota, "Normative systems represented as hybrid knowledge bases," in *Proc. 12th Int. Conf. Comput. Logic Multi-Agent Syst.*, in Lecture Notes in Computer Science, 2011, pp. 330–346.

[4] Y. Zhang and J. Leezer, "Emergence of social norms in complex networks," in *Proc. Int. Conf. Comput. Sci. Eng.*, vol. 4, Aug. 2009, pp. 549–555.

[5] M. A. Mahmoud, M. S. Ahmad, A. Ahmad, M. Z. M. Yusoff, A. Mustapha, and N. H. A. Hamid, "Obligation and prohibition norms mining algorithm for normative multi-agent systems," in *Proc. KES-AMSTA*, May 2013, pp. 115–124.

[6] M. A. Mahmoud, M. S. Ahmad, A. Ahmad, M. Z. M. Yusoff, and A. Mustapha, "Norms detection and assimilation in multi-agent systems: A conceptual approach," in *Knowledge Technology Week*. Berlin, Germany: Springer, 2011, pp. 226–233.

[7] M. A. Mahmoud, M. S. Ahmad, A. Ahmad, A. Mustapha, M. Z. M. Yusoff, and N. H. A. Hamid, "Building norms-adaptable agents from potential norms detection technique PNDT," *Int. J. Intell. Inf. Technol.*, vol. 9, no. 3, pp. 38–60, 2013.

[8] J. Li, F. Meneguzzi, M. Fagundes, and B. Logan, "Reinforcement learning of normative monitoring intensities," in *Proc. Int. Workshop Coordination, Org., Inst., Norms Agent Syst.* Springer, Jul. 2015, pp. 209–223.

[9] R. Gonçalves and J. J. Alferes, "Specifying and reasoning about normative systems in deontic logic programming," in *Proc. 11th Int. Conf. Auton. Agents Multiagent Syst.*, Jun. 2012, pp. 1423–1424.

[10] C. D. Hollander and A. S. Wu, "The current state of normative agent-based systems," *J. Artif. Soc. Social Simul.*, vol. 14, no. 2, p. 6, 2011.

[11] H. Verhagen, "Norm autonomous agents, stockholm: Department of system and computer sciences," Roy. Inst. Technol., Stockholm Univ., Stockholm, Sweden, Tech. Rep., 2000.

[12] B. T. R. Savarimuthu, M. Purvis, and M. K. Purvis, "Social norm emergence in virtual agent societies (short paper)," in *Proc. 7th Int. Joint Conf. Auto. Agents Multi-Agent Syst.*, Dunedin, New Zealand, 2008, pp. 1521–1524.

[13] D. Villatoro, S. Sen, and J. Sabater-Mir, "Of social norms and sanctioning: A game theoretical overview," *Int. J. Agent Technol. Syst.*, pp. 1–15, 2010.

[14] M. A. Mahmoud, M. S. Ahmad, M. Z. M. Yusoff, and A. Mustapha, "A review of norms and normative multiagent systems," *Sci. World J.*, vol. 2014, Jul. 2014, Art. no. 684587.

[15] M. A. Mahmoud, A. Mustapha, M. S. Ahmad, A. Ahmad, M. Z. M. Yusoff, and N. H. A. Hamid, "Potential norms detection in social agent societies," in *Distributed Computing and Artificial Intelligence*. Cham, Switzerland: Springer, 2013, pp. 419–428.

[16] M. Mahmoud, M. S. Ahmad, and M. Z. M. Yusoff, "Development and implementation of a technique for norms-adaptable agents in open multi-agent communities," *J. Syst. Sci. Complex.*, vol. 29, no. 6, pp. 1519–1537, 2016.

[17] D. Villatoro, "Self-organization in decentralized agent societies through social norms," in *Proc. 10th Int. Conf. Auton. Agents Multiagent Syst.*, May 2011, pp. 1373–1374.

[18] P. Caire, "A normative multi-agent systems approach to the use of conviviality for digital cities," in *Proc. Int. Workshop Coordination, Org., Inst., Norms Agent Syst.*, 2007, pp. 245–260.

[19] G. Boella and L. van der Torre, "Regulative and constitutive norms in normative multiagent systems," in *Proc. 9th Int. Conf. Princ. Knowl. Represent. Reasoning*, Whistler, BC, Canada, 2004, pp. 255–265.

[20] G. Boella and L. van der Torre, "Substantive and procedural norms in normative multiagent systems," *J. Appl. Log.*, vol. 6, no. 2, pp. 152–171, Jun. 2008.

[21] F. L. Y. López, M. Luck, and M. D'Inverno, "A normative framework for agent-based systems," *Comput. Math. Org. Theory*, vol. 12, nos. 2–3, pp. 227–250, 2006.

[22] R. Rubino, A. Omicini, and E. Denti, "Computational institutions for modelling norm-regulated MAS: An approach based on coordination artifacts," in *Proc. AAMAS Workshops*, 2005, pp. 127–141.

[23] A. Peczenik, *On Law and Reason*. Springer, 2009.

[24] G. Boella, L. van der Torre, and H. Verhagen, "Ten challenges for normative multi-agent systems," in *Proc. Dagstuhl Seminar*, R. Bordini, Ed., 2008, pp. 1–11.

[25] R. Conte and F. Dignum, "From social monitoring to normative influence," *J. Artif. Soc. Social Simul.*, vol. 4, no. 2, p. 7, 2001.

[26] B. T. R. Savarimuthu, S. Cranefield, M. A. Purvis, and M. K. Purvis, "Identifying prohibition norms in agent societies," *Artif. Intell. Law*, vol. 21, no. 1, pp. 1–46, 2013.

[27] B. T. R. Savarimuthu, S. Cranefield, M. A. Purvis, and M. K. Purvis, "Obligation norm identification in agent societies," *J. Artif. Soc. Social Simul.*, vol. 13, no. 4, p. 3, 2010.

[28] J. Campos, M. López-Sánchez, and M. Esteva, "A case-based reasoning approach for norm adaptation," in *Proc. 5th Int. Conf. Hybrid Artif. Intell. Syst. (HAIS)*. Madrid, Spain: Springer, 2010, pp. 168–176.

[29] J. M. Epstein, "Learning to be thoughtless: Social norms and individual computation," *Comput. Econ.*, vol. 18, no. 1, pp. 9–24, 2001.

[30] F. López, "Social power and norms: Impact on agent behaviour," Ph.D. dissertation, Dept. Electron. Comput. Sci., Univ. Southampton, Southampton, U.K., 2003.

[31] G. Andrighetto, M. Campenni, F. Cecconi, and R. Conte, "The complex loop of norm emergence: A simulation model," in *Simulating Interacting Agents and Social Phenomena. Agent-Based Social Systems*, vol. 7, C. Cioffi-Revilla, G. Deffuant, and K. Takadama, Eds. Tokyo, Japan: Springer, 2010, pp. 19–35.

[32] S. Sen and S. Airiau, "Emergence of norms through social learning," in *Proc. IJCAI*, 2007, pp. 1507–1512.

[33] M. A. Mahmoud, M. S. Ahmad, and M. Z. M. Yusoff, "A norm assimilation approach for multi-agent systems in heterogeneous communities," in *Proc. Asian Conf. Intell. Inf. Database Syst.* Berlin, Germany: Springer, Mar. 2016, pp. 354–363.

[34] M. A. Mahmoud, M. S. Ahmad, A. Ahmad, M. Z. M. Yusoff, and A. Mustapha, "A norms mining approach to norms detection in multi-agent systems," in *Proc. Int. Conf. Comput. Inf. Sci. (ICCIS)*, vol. 1, Jun. 2012, pp. 458–463.

[35] M. A. Mahmoud, M. S. Ahmad, M. Z. M. Yusoff, and S. A. Mostafa, "A regulative norms mining algorithm for complex adaptive system," in *Proc. Int. Conf. Soft Comput. Data Mining*. Cham, Switzerland: Springer, 2018, pp. 213–224.

[36] S. A. Mostafa, M. S. Ahmad, A. Ahmad, M. Annamalai, and S. S. Gunasekaran, "A flexible human-agent interaction model for supervised autonomous systems," in *Proc. 2nd Int. Symp. Agent, Multi-Agent Syst. Robot. (ISAMSR)*, Aug. 2016, pp. 106–111.

[37] S. A. Mostafa, M. S. Ahmad, M. Annamalai, A. Ahmad, and S. S. Gunasekaran, "A dynamically adjustable autonomic agent framework," in *Advances in Information Systems and Technologies*. Berlin, Germany: Springer, 2013, pp. 631–642.

[38] S. A. Mostafa, R. Darman, S. H. Khaleefah, A. Mustapha, N. Abdullah, and H. Hafit, "A general framework for formulating adjustable autonomy of multi-agent systems by fuzzy logic," in *Proc. KES Int. Symp. Agent Multi-Agent Syst., Technol. Appl.* Cham, Switzerland: Springer, Jun. 2018, pp. 23–33.

[39] S. A. Mostafa, M. S. Ahmad, M. Annamalai, A. Ahmad, and S. S. Gunasekaran, "A conceptual model of layered adjustable autonomy," in *Advances in Information Systems and Technologies*. Berlin, Germany: Springer, 2013, pp. 619–630.

[40] S. A. Mostafa, M. S. Ahmad, A. Ahmad, M. Annamalai, and A. Mustapha, "A dynamic measurement of agent autonomy in the layered adjustable autonomy model," in *Recent Developments in Computational Collective Intelligence*. Cham, Switzerland: Springer, 2014, pp. 25–35.

[41] A. Ahmad, M. Zaliman, M. Yusof, M. S. Ahmad, M. Ahmed, and A. Mustapha, "Resolving conflicts between personal and normative goals in normative agent systems," in *Proc. IEEE 7th Int. Conf. Inf. Technol. Asia*, Jul. 2011, pp. 1–6.

[42] M. Ahmed, M. S. Ahmad, and M. Z. M. Yusoff, "Modeling agent-based collaborative process," in *Proc. Int. Conf. Comput. Collective Intell.* Berlin, Germany: Springer, Nov. 2010, pp. 296–305.

[43] M. Ahmed, M. S. Ahmad, and M. Z. M. Yusoff, "Mitigating human-human collaboration problems using software agents," in *KES International Symposium on Agent and Multi-Agent Systems: Technologies and Applications*. Berlin, Germany: Springer, Jun. 2010, pp. 203–212.

[44] F. L. Y. López, "Social powers and norms: Impact on agent behaviour," Ph.D. dissertation, Dept. Electron. Comput. Sci., Univ. Southampton, Southampton, U.K., 2003.

[45] A. Symeonidis and P. Mitkas, "A methodology for predicting agent behavior by the use of data mining techniques," in *Autonomous Intelligent Systems: Agents and Data Mining* (Lecture Notes in Computer Science), vol. 3505. Berlin, Germany: Springer, 2005, pp. 161–174.

[46] C. Burr, N. Cristianini, and J. Ladyman, "An analysis of the interaction between intelligent software agents and human users," *Minds Mach.*, vol. 28, no. 4, pp. 735–774, 2018.

[47] T. Bösser, "Autonomous agents," in *International Encyclopedia of the Social & Behavioral Sciences*, 2nd ed. 2015, pp. 309–313.

[48] S. A. Mostafa, M. S. Ahmad, A. Mustapha, and M. A. Mohammed, "A concise overview of software agent research, modeling, and development," *Softw. Eng.*, vol. 5, no. 1, pp. 8–25, 2017.

[49] F. Cunha, T. Sirqueira, M. Viana, and C. Lucena, "Extending BDI multiagent systems with agent norms world academy of science, engineering and technology, open science index 137," *Int. J. Comput. Inf. Eng.*, vol. 12, no. 5, pp. 302–309, 2018.

[50] Z. Shams, M. De Vos, J. Padget, and W. W. Vasconcelos, "Practical reasoning with norms for autonomous software agents," *Eng. Appl. Artif. Intell.*, vol. 65, pp. 388–399, Oct. 2017.

[51] L. Bölöni, T. S. Bhatia, S. A. Khan, J. Streater, and S. M. Fiore, "Towards a computational model of social norms," *PLoS ONE*, vol. 13, no. 4, 2018, Art. no. e0195331.

MOAMIN A. MAHMOUD received the bachelor's degree in mathematics from the College of Mathematics and Computer Science, University of Mosul, Iraq, in 2007, the master's degree in information technology from the College of Graduate Studies, Universiti Tenaga Nasional (UNITEN), Malaysia, in 2010, and the Ph.D. degree in information and communication technology from Universiti Tenaga Nasional (UNITEN), Malaysia, in 2013, where he joined as a Senior Lecturer with the Department of Software Engineering, in 2014. He has published more than 80 articles of indexed journals, book chapters, and conferences. He successfully supervised more than 40 undergraduate software engineering students, ten master's degree students in information technology, and three Ph.D. degree students in information and communication technology. His current research interests include artificial intelligence, distributed and autonomous systems, complex adaptive systems, and the IoT-based smart systems.

MOHD SHARIFUDDIN AHMAD received the B.Sc. degree in electrical and electronic engineering from Brighton Polytechnic, U.K., in 1980, the M.Sc. degree in artificial intelligence from Cranfield University, U.K., in 1995, and the Ph.D. degree from Imperial College London, U.K., in 2005. He started his career as a Power Plant Engineer specializing in process instrumentation and control, in 1980. He joined as a Principal Lecturer and the Head of the Department of Computer Science and Information Technology with Universiti Tenaga Nasional (UNITEN), where he has been an Associate Professor, since 2006. His research interests include applying constraints to develop collaborative frameworks in multi-agent systems, collaborative interactions in multi-agent systems, and tacit knowledge management using AI techniques.

SALAMA A. MOSTAFA received the B.Sc. degree in computer science from the University of Mosul, Iraq, in 2003, and the M.Sc. and Ph.D. degrees in information technology from Universiti Tenaga Nasional (UNITEN), Malaysia, in 2011 and 2016, respectively. He is currently a member of the Postdoctoral Staff with Universiti Tun Hussein Onn Malaysia, Malaysia. His research interests include software engineering, artificial intelligence and their integration, including software agents and intelligent autonomous systems.