Automated Synthesis of Normative Systems

Javier Morales[1,2] and Maite Lopez-Sanchez[1]
[1]MAiA Department
Universitat de Barcelona
Barcelona, Spain
jmorales@iiia.csic.es
maite@maia.ub.es

Juan A. Rodriguez-Aguilar
[2]Artificial Intelligence Research Institute (IIIA)
Spanish Council of Scientific Research (CSIC)
Campus UAB. Bellaterra, Spain
jar@iiia.csic.es

Michael Wooldridge
Dept. of Computer Science
University of Oxford
Oxford, United Kingdom
mjw@cs.ox.ac.uk

Wamberto Vasconcelos
Dept. of Computing Science
University of Aberdeen
Aberdeen, UK
wvasconcelos@acm.org

ABSTRACT

Normative systems (norms) have been widely proposed as a technique for coordinating multi-agent systems. The automated synthesis of norms for coordination remains an open and complex problem, which we tackle in this paper. We propose a novel mechanism called IRON (Intelligent Robust On-line Norm synthesis mechanism), for the on-line synthesis of norms. IRON aims to synthesise conflict-free norms without lapsing into over-regulation. Thus, IRON produces norms that characterise necessary conditions for coordination, without over-regulation. In addition to defining the norm synthesis problem formally, we empirically show that IRON is capable of synthesising norms that are effective even in the presence of non-compliance behaviours in a system.

Categories and Subject Descriptors

I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence —*Multiagent Systems*

Keywords

Norms - Normative Systems - Norm Synthesis

1. INTRODUCTION

Norms have been widely proposed as a technique for coordinating multi-agent systems (MAS). A norm can be understood as an established, expected pattern of behaviour [14]. Typically, these behavioural patterns impose constraints on the behaviour of individuals in order to avoid conflicts (e.g., collisions in a traffic scenario).

Since the seminal work of Shoham and Tennenholtz [12], the problem of norm synthesis (i.e., determining the set of norms that avoid conflicting states) has attracted considerable attention. We differentiate two strands of work tackling this problem: the *off-line* and *on-line* norm synthesis approaches. On the one hand, off-line approaches (such as [12, 4]) aim at synthesising norms for a MAS that constrain the behaviour of agents while ensuring the achievement of global system goals. Off-line approaches require detailed knowledge of a MAS, (e.g., its full state space), at design time. Some refinements to the basic approach have included the implementation costs of norms and multiple design goals with different priorities [1]. Following [12], the complexity of the norm synthesis problem is high (NP-complete). This has recently spurred research to better cope with the size of the state space [2].

Nonetheless, off-line design is not appropriate to cope with open MAS, whose composition and state space change with time. On-line norm synthesis approaches (such as e.g. [9]) try to overcome such limitations by synthesising norms that regulate a MAS at run-time instead of at design time. More recently, norm emergence has become a popular technique for on-line norm synthesis (e.g., [6, 8, 10, 11, 13]). It does not require any global state representation or centralized control. Instead, it considers that agents collaboratively choose their own norms out of a space of possible norms. A norm is considered to have emerged when a majority of agents adopt it and abide by it. Approaches based on norm emergence suffer from several drawbacks. Firstly, convergence is highly sensitive to the initial conditions in the MAS. Secondly, there is the assumption that agents collaborate during the norm synthesis process and that agents are endowed with the necessary machinery to participate in the emergence process. Third, regarding the norm synthesis process, although the utility of norms is eventually considered, there are further aspects that, to the best of our knowledge, have not been taken into account yet. On the one hand, it is not considered whether a synthesised norm is truly necessary or not (because its regulation is subsumed by another norm). Thus, it might be the case that a norm within a normative system is not really necessary, and hence leads to over-regulation: agents must handle more norms than needed. On the other hand, the generalisation of a set of norms into a more general one is not considered either as part of the norm synthesis process.

Against this background, we propose a novel mechanism called IRON (Intelligent Robust On-line Norm synthesis machine), for the on-line synthesis of norms. IRON produces norms for the agents in a MAS that characterise necessary conditions for coordination, while avoiding over-regulation. IRON synthesises norms that are both *effective* and *necessary*. Furthermore, we endow IRON with the capability of generalising norms. By generalising norms and discarding unnecessary norms, we allow IRON to yield *concise* normative systems. Finally, we empirically show that IRON successfully synthesises norms that are effective and necessary, even in the presence of non-compliance behaviours in a MAS. IRON is resilient to a very high percentage of violations (up to 50%).

The paper is organised as follows. Section 2 formally introduces the norm synthesis problem that we tackle in this paper. Section 3 details IRON and Section 4 offers its empirical evaluation. Section 5 draws some conclusions and sets paths to future research.

2. PROBLEM STATEMENT

Consider a MAS composed of a set of agents Ag, and a finite set of actions $Ac = \{ac_1, \ldots, ac_n\}$ that these agents can perform. Let

Appears in: *Proceedings of the 12th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2013), Ito, Jonker, Gini, and Shehory (eds.), May, 6–10, 2013, Saint Paul, Minnesota, USA.*

S be the set of all possible states of the system, and let $C \subseteq S$ be a set of *conflicting* states. For instance, consider a traffic scenario in which the agents are cars. Each state of the traffic network would correspond to each state of the MAS, and conflicting states would correspond to those states containing e.g., collisions or traffic jams.

We will use a language $\mathcal{L}$ to describe the states of a MAS. This language, to be more formally defined later, is a logical language containing the standard classical connectives, and a notion of consequence defined for it via a relation "$\models$". Given a state $s \in S$, we let $\tau(s)$ denote an expression in $\mathcal{L}$ that describes the state. Using the mapping τ, we can capture the notion of *partial view* of some state s as a sub-expression of $\tau(s)$. For instance, if $\mathcal{L}$ is a first-order language and s stands for a state of a traffic network, $\tau(s)$ would be composed of the predicates describing the state. A partial view of s could be a given junction, and its description a subset of the predicates in $\tau(s)$. Hereafter, ν_s^i will stand for i-th *partial view* of state s. We assume that views are correct, but incomplete.

From the agent perspective, each agent has her own local perception of the state of the MAS she is part of. For instance, an agent at a junction has her own *local view* of the state of the MAS. A local view is an agent's internal representation of a partial view (i.e., its belief). Agents express their local views in terms of an *agent language* $\mathcal{L}_{Ag}$. Therefore, while $\mathcal{L}$ is the MAS language representing a global, external observer's perspective, $\mathcal{L}_{Ag}$ is the agent language representing a local, individual perspective. Details of $\mathcal{L}_{Ag}$ are not significant for now, however, we will assume as above that we have a notion of consequence defined for the language via a relation $\models$. Henceforth, given an agent $ag \in Ag$, we will refer to her local view as her *context*, and we denote it by $c(ag)$.

Now we are ready to introduce our notion of norm, which establishes obligations, permissions and/or prohibitions to an individual agent whenever some pre-conditions are fulfilled. Such preconditions are expressed by means of language $\mathcal{L}_{Ag}$, namely in terms of an agent's point of view.

DEFINITION 1. *A* norm *is a pair $\langle \varphi, \theta(ac) \rangle$ where $\varphi \in \mathcal{L}_{Ag}$ stands for the precondition of the norm, $ac \in Ac$ is an action, and $\theta \in \{obl, perm, prh\}$ is a deontic operator.*

An agent $ag \in Ag$ evaluates whether a norm $n = \langle \varphi, \theta(ac) \rangle$ applies as follows. We say that the context of ag, $c(ag)$, satisfies the pre-condition of norm n iff $c(ag) \models \varphi$. Then, norm n applies to agent ag and the deontic expression $\theta(ac)$ will hold for her. For instance, within a traffic scenario, consider a norm that establishes an obligation to stop for an agent that sees a car to its left that is heading towards its right. We represent this norm as $\langle left(>), obl(stop) \rangle$ where precondition $left(>)$ is a proposition that is true if there is a car heading to the right of the agent evaluating the norm and it is located at its left, $stop$ stands for the action to consider, and obl is the deontic operator. So, if agent ag's context is $left(>)$, then, $c(ag) \models \varphi$ holds and $obl(stop)$ applies to the car.

In this paper, we focus on a particular type of MAS, namely norm-aware multi-agent system (NA-MAS). A NA-MAS is a MAS whose agents have their actions regulated by some *normative system* (set of norms) they are aware of. Moreover, the MAS itself can assess whether and to whom norms in the normative system apply. Formally:

DEFINITION 2. *A Norm-aware Multi-agent System (NA-MAS) is a tuple $\langle Ag, Ac, \Omega, \mathcal{L}_{Ag}, S \rangle$, where:* (i) *$Ag$ is a set of agents;* (ii) *Ac is a set of agent actions;* (iii) *Ω is a normative system, whose norms are expressed in the agent language $\mathcal{L}_{Ag}$; and* (iv) *S is a set of states.*

Given a NA-MAS, our aim is to generate a normative system that satisfactorily avoids lapsing into conflicting states while avoiding over-regulation. With this aim, given a normative system we must be able to measure: (i) the effectiveness of its norms in preventing conflicts; and (ii) whether its norms are necessary or whether they include redundancy. Furthermore, since the state of a NA-MAS changes as agents interact, our goal is to find a *stable* normative system, namely a set of norms that are sufficiently effective and necessary for a given period of time. We measure this sufficiency by applying specific thresholds (α_{eff} and α_{nec}) over effectiveness and necessity measurement functions (μ_{eff} and μ_{nec}). Finally, we are ready to define the problem that we address in this paper.

DEFINITION 3. *Given a NA-MAS $M = \langle Ag, Ac, \Omega, \mathcal{L}_{Ag}, S \rangle$, a set of conflicting states $C \subseteq S$ and functions μ_{eff}, μ_{nec} to assess the effectiveness and necessity of a normative system, the norm synthesis problem (NSP) is that of finding a normative system $\bar{\Omega}$ such that $\mu_{eff}(\bar{\Omega}, M, C, t) \geq \alpha_{eff}$ and $\mu_{nec}(\bar{\Omega}, M, C, t) \geq \alpha_{nec}$ for all $t \in [t_{begin}, t_{end}]$, where $\alpha_{eff}, \alpha_{nec} \in [0, 1]$ are thresholds that establish a satisfaction degree for both effectiveness and necessity, and $[t_{begin}, t_{end}]$ is a time interval.*

3. IRON: A NORM SYNTHESIS MECHANISM

In this section we introduce the Intelligent Robust On-line Norm synthesis mechanism (IRON), a norm synthesis approach aimed at solving the norm synthesis problem formalised by definition 3. With this aim, given a NA-MAS, IRON operates by continuously iterating the following steps: (1) it monitors the NA-MAS operation; (2) it decides upon the addition of brand new norms to the current (initially empty[1]) normative system; (3) it evaluates whether the effectiveness and necessity of the normative system are within expected thresholds; (4) if required, it refines the normative system; and (5) it makes the normative system available to the agents. Notice therefore that IRON continuously searches for a normative system *on-line*, while agents in the system are operating.

IRON is based on four components: (i) a grammar to synthesise new norms; (ii) the normative network (a data structure to represent normative systems and explored norms); (iii) a set of operators that make it possible to transform one normative system into another; and (iv) a strategy that specifies when to use such operators. We describe below each component in detail, and explain IRON's architecture and computational model.

3.1 Grammar for norm synthesis

Our approach employs a grammar to synthesise candidate norms of the form $\langle \varphi, \theta(ac) \rangle$ (cf. Def 1). We have adapted our grammar from [5], using as building blocks *atomic formulae* of the form $p^n(\tau_1, \ldots, \tau_n)$, p being an n-ary predicate symbol and $\tau_1, \ldots, \tau_n$ terms of $\mathcal{L}_{Ag}$.

$$\begin{array}{lll}
\text{Norm} & ::= & \langle \text{LHS, RHS} \rangle \\
\text{LHS} & ::= & \text{LHS \& LHS} \mid \alpha \\
\text{RHS} & ::= & \theta(Ac) \\
\theta & ::= & obl \mid perm \mid prh \\
Ac & ::= & ac_1 \mid ac_2 \mid \ldots \mid ac_n \\
\alpha & ::= & p^n(\tau_1, \ldots, \tau_n)
\end{array}$$

If we consider further our traffic scenario, we can employ the grammar above to synthesise:

[1]The approach would also work if the normative system is initialised with a set of norms provided at design time.

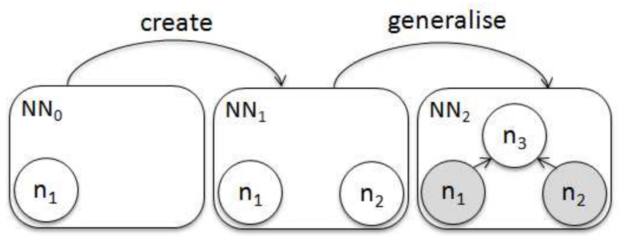

Figure 1: Evolution of a normative network along time.

$$n_1 : \langle \mathit{left}(>) \& \mathit{front}(>) \& \mathit{right}(>),\ \mathit{obl}(\mathit{stop}) \rangle$$
$$n_2 : \langle \mathit{left}(>) \& \mathit{front}(>) \& \mathit{right}(\text{-}),\ \mathit{obl}(\mathit{stop}) \rangle$$

Norm n_1 enforces a car to stop (hence giving way) if there is a car heading towards its right to its left, front and right positions. Norm n_2 enforces a car to stop if there is a car heading towards its right to its left and front, but there is nothing (indicated by "-") to its right. Notice that norms n_1, n_2 only differ in their right position.

3.2 A representation of normative systems

Since IRON will continuously synthesise norms in search for a satisfactory normative system, it must be able to differentiate between the norms that are currently part of the normative system and those that are not (i.e., they have been explored but they are not currently active). For this purpose, IRON employs a graph-based data structure, which we call a *normative network*, to represent normative systems. A normative network is a graph whose nodes stand for norms and whose edges stand for relationships (generalisations in this paper) between norms. Norms in a network may be either *active* or *inactive*. We consider that a normative network represents a normative system as its active norms.

Figure 1 illustrates the evolution of a normative network (and its corresponding normative system) over time points t_0, t_1, t_2. At instant t_0 the normative network NN_0 has a single active norm n_1 (represented as a white circle) and $\Omega_0 = \{n_1\}$. At instant t_1 a new norm, n_2, is added to NN_0, yielding NN_1 and $\Omega_1 = \{n_1, n_2\}$. Finally, at instant t_2, norms n_1 and n_2 are generalised as norm n_3 and deactivated (represented as grey circles) giving rise to NN_2 and $\Omega_2 = \{n_3\}$. In fact, Figure 1 illustrates the way IRON performs the norm synthesis process. In general the process will consist of continuously operating over (applying changes to) the normative network according to some strategy until it finds a normative system that solves the norm synthesis problem (NSP).

Now we are ready to offer a formal definition of the normative network employed by IRON.

DEFINITION 4. *A Normative Network (NN) is a tuple $\langle \mathcal{N}, R_G, \Delta, \delta \rangle$ where:* (i) *$\mathcal{N}$ is a set of norms;* (ii) *$R_G \subseteq \mathcal{N} \times \mathcal{N}$ is a* generalisation *relationship between norms;* (iii) *$\Delta = \{\mathit{active}, \mathit{inactive}\}$ is the set of possible states of a norm;* (iv) *$\delta : \mathcal{N} \rightarrow \Delta$ is a function that returns the state of a norm $n \in \mathcal{N}$.*

Since IRON considers that the current normative system is composed of the norms that are currently active in the normative network, we define $\Omega = \{n \mid n \in \mathcal{N} \wedge \delta(n) = \mathit{active}\}$.

The Normative Network definition above considers generalisation as the only relationship between the norms in $\mathcal{N}$. Given two norms n, n' we say that n' is *generalised* by n if the applicability conditions of n' are more restrictive than the applicability conditions of n and if both modalities and actions are equal. Formally:

DEFINITION 5. *Given norms $n = \langle \varphi, \theta(a) \rangle$, $n' = \langle \varphi', \theta(a) \rangle$, n' is* generalised *by n, denoted by $n' \subset n$, iff $\varphi' \neq \varphi$ and $\varphi' \models \varphi$.*

Consider again a traffic scenario and the following norm:

$$n_3 : \langle \mathit{left}(>) \& \mathit{front}(>)\ ,\ \mathit{obl}(\mathit{stop}) \rangle$$

Notice that the applicability condition of n_3 is more general than those of n_1 and n_2 above because a car is obliged to stop if it finds a car heading towards its right to its left and to the front, but no matter what it perceives to its right position. Therefore, n_3 generalises both n_1 and n_2.

In general, if n_j is *generalised* by n_i, then we also say that n_i is *specialised* by n_j. If there exists at least some $n_k \in \mathcal{N}$ such that $n_j \subset n_k \subset n_i$ we say that n_i is an *ancestor* of n_j, otherwise n_i is a *father* of n_j. If n_j is not *generalised* by n_i, we denote it by $n_j \not\subseteq n_i$.

3.3 Operators for normative networks

IRON will search for a normative system that solves the NSP by transforming an initial normative network over time, hence moving from one normative system to another. With this aim, our norm synthesis mechanism implements a collection of normative network operators. Each operator transforms IRON's normative network $\langle \mathcal{N}, \mathcal{R}, \Delta, \delta \rangle$ into another one $\langle \mathcal{N}', \mathcal{R}', \Delta', \delta' \rangle$. More precisely, IRON implements operators to perform:

- The *creation* of a new norm using a grammar (as described in Section 3.1) to add it to the normative system.
- The *deactivation* of a norm in the normative system.
- The *generalisation* of a set of norms in the normative system into a more general norm (e.g., considering the example above, generalising n_1, n_2 into n_3).
- The *specialisation* of a norm in the normative system into more specific norms. This operation reverses the result of a generalisation (e.g., n_3 can be specialised into n_1 and n_2).

Table 1 formally specifies each of these operators.

Operator	Specification
$\mathit{create}(NN, \mathcal{G}, \langle \nu^i_{s_{t-1}}, \nu^i_{s_t} \rangle)$	$n \leftarrow \mathit{synthesiseNorm}(\mathcal{G}, \langle \nu^i_{s_{t-1}}, \nu^i_{s_t} \rangle)$ $\mathcal{N}' \leftarrow \mathcal{N} \cup \{n\}$ $\delta'(n) \leftarrow \mathit{active}$ $NN' \leftarrow \langle \mathcal{N}', R_G, \Delta, \delta' \rangle$
$\mathit{deactivate}(NN, n)$	$\delta'(n) \leftarrow \mathit{inactive}$ $NN' \leftarrow \langle \mathcal{N}, R_G, \Delta, \delta' \rangle$
$\mathit{generalise}(NN, \mathit{parent}, \mathit{children})$	$\mathcal{N}' \leftarrow \mathcal{N} \cup \{\mathit{parent}\}$ $R'_G \leftarrow R_G \cup \{(\mathit{ch}, \mathit{parent}) \mid \mathit{ch} \in \mathit{children}\}$ $\delta'(\mathit{parent}) \leftarrow \mathit{active}$ $\delta'(\mathit{ch}) \leftarrow \mathit{inactive}$ for all $\mathit{ch} \in \mathit{children}$ $NN' \leftarrow \langle \mathcal{N}', R'_G, \Delta, \delta' \rangle$
$\mathit{specialise}(NN, \mathit{parent}, \mathit{children})$	$\delta'(\mathit{parent}) \leftarrow \mathit{inactive}$ for all $\mathit{child} \in \mathit{children}$ if $(\mathit{child}, \mathit{parent}) \in R_G$ $\delta'(\mathit{child}) \leftarrow \mathit{active}$ $NN' \leftarrow \langle \mathcal{N}, R_G, \Delta, \delta' \rangle$

Table 1: IRON **operators.**

3.3.1 *Creating norms*

The *create* operator synthesises a new norm, using grammar $\mathcal{G}$, aimed at preventing a conflicting state. The new norm is added to IRON's normative network, and it is also activated. Observe that a pair $\langle \nu^i_{s_{t-1}}, \nu^i_{s_t} \rangle$ stands for a transition between two views of a state at consecutive times. The differences between these views captures the local changes that occurred when a NA-MAS evolved from $t-1$ to t. Consider views $\langle \nu^i_{s_{t-1}}, \nu^i_{s_t} \rangle$ such that s_t is a conflicting state and $\nu^i_{s_t}$ contains some conflict (e.g., the collision of two cars at a particular junction). The create operator uses the function $synthesiseNorm$ to synthesise a new norm aimed at preventing the conflict at $\nu^i_{s_t}$. The implementation of this function is based on a Case-Based Reasoning (CBR) unsupervised approach along the lines of the one used in [7]. Thus, this approach is based on the following principle: if we can prevent a conflict at a given situation by enacting a norm, it is likely that we can prevent a conflict at a similar situation by means of a similar norm. Once this new norm is synthesised, it is added to the normative network ($\mathcal{N}' = \mathcal{N} \cup \{n\}$), and its state is set to active ($\delta'(n) = active$).

3.3.2 Deactivating norms

The implementation of the $deactivate$ operator sets the state of a given norm to *inactive*. Hence, although the norm remains in the normative network, it is no longer part of the normative system.

3.3.3 Generalising norms

The *generalise* operator generalises a set of norms (*children*) into a more general norm (*parent*) by: (i) adding the parent norm to the network; (ii) establishing new generalisation relations (R_G) between each generalised (child) norm and the parent norm in the normative network; (iii) setting the state of the parent to active and the children's to inactive. As a result, the child norms will no longer belong to the normative system, but the parent norm will do.

3.3.4 Specialising norms

The *specialise* operator undoes the result of a generalisation by setting to inactive the state of the parent (more general) norm and setting to active the state of its children. Thus, thereafter all the child norms become candidates to belong to the normative system, while the parent norm does not any longer.

3.4 A strategy to synthesise normative systems

Operators are invoked by following a specific strategy. Our proposal is to monitor the evolution of the system at regular time intervals (i.e., ticks) and apply operators under certain conditions. At every tick, IRON runs its strategy to perform three tasks: *(i)* synthesis of new norms; *(ii)* evaluation of the current normative system; and *(iii)* refinement of the normative system by means of specialisations, generalisations and deactivation of norms. Once the strategy finishes, it outputs the normative system represented by the normative network.

Algorithm 1 specifies IRON's strategy (Π), which requires as input: (i) a list of $views$ $[\langle \nu^0_{s_{t-1}}, \nu^0_{s_t} \rangle, \ldots, \langle \nu^m_{s_{t-1}}, \nu^m_{s_t} \rangle]$ where $\nu^i_{s_{t-1}}$ and $\nu^i_{s_t}$ stand for the i-th view of the state of a NA-MAS at time $t-1$ and at time t respectively; (ii) a normative network NN; (iii) a grammar $\mathcal{G}$; (iv) a function f_{apply} to check norm applicability (v) a function $f_{conflict}$ to detect conflicts; (vi) two evaluation functions μ_{eff}, μ_{nec} to assess the effectiveness and necessity of norms; and (vii) Θ, a set of satisfaction degree thresholds described below ($\Theta = \{\alpha_{eff}, \alpha_{nec}, \alpha^{deact}_{eff}, \alpha^{deact}_{nec}, \alpha^{gen}_{eff}, \alpha^{gen}_{nec}\}$).

3.4.1 Synthesis of new norms

The norm synthesis process starts searching for conflicts in all the received views. Thereafter, for each pair $\langle \nu^i_{s_{t-1}}, \nu^i_{s_t} \rangle$, the strategy detects whether there is a conflict in view $\nu^i_{s_t}$. Finally, for each detected conflict, the strategy synthesises a new norm in order to avoid it in the future. Function $conflictDetection$ (line 2) uses function $f_{conflict}$ to identify the pairs $\langle \nu^i_{s_{t-1}}, \nu^i_{s_t} \rangle$ in $views$ that

Algorithm 1 IRON functions

```
 1: function $\Pi(views, NN, \mathcal{G}, f_{apply}, f_{conflict}, \mu_{eff}, \mu_{nec}, \Theta, T)$
 2:     $conflicts \leftarrow conflictDetection(views, f_{conflict})$
 3:     for all $\langle \nu^i_{s_{t-1}}, \nu^i_{s_t} \rangle \in conflicts$ do
 4:         $NN \leftarrow create(NN, \mathcal{G}, \langle \nu^i_{s_{t-1}}, \nu^i_{s_t} \rangle)$
 5:     $appNorms \leftarrow normApplicability(NN, views, f_{apply})$
 6:     $(A, V) \leftarrow normCompliance(appNorms, f_{conflict})$
 7:     $U \leftarrow updateUtilities(A, V, \mu_{eff}, \mu_{nec})$
 8:     $P \leftarrow updatePerformances(U, T)$
 9:     for all $n \in norms(A, V)$ do
10:         if $isUnderperforming(n, P, \Theta)$ then
11:             $specialiseDown(NN, n, P, \Theta)$
12:         else
13:             $L \leftarrow validGeneralisations(NN, \mathcal{G}, n, P)$
14:             for all $generalisation \in L$ do
15:                 $parent \leftarrow getParent(generalisation)$
16:                 $children \leftarrow getChildren(generalisation)$
17:                 $NN \leftarrow generalise(NN, parent, children)$
18:     $\Omega \leftarrow \{n \in NN | \delta(n) = active\}$
19:     return $\Omega$
20:
21: function SPECIALISEDOWN$(NN, n, P, \Theta)$
22:     if $isLeaf(n)$ then
23:         $NN \leftarrow deactivate(NN, n)$
24:     else
25:         $children \leftarrow getChildren(n)$
26:         $NN \leftarrow specialise(NN, n, children)$
27:     for all $child \in children$ do
28:         if $isUnderPerforming(child, P, \Theta)$ then
29:             $specialiseDown(NN, child, P, \Theta)$
30:     return $L$
```

contain conflicts. New norms to avoid conflicts are created by the $create$ operator (line 4), which uses grammar $\mathcal{G}$ to synthesise new norms and add them to the normative network.

3.4.2 Norm evaluation

The strategy updates the effectiveness and necessity of norms by evaluating them individually. Given a particular view and an agent that is part of it, the agent may decide whether to follow the norm or violate it. In general, we will evaluate a norm depending on the outcome (e.g., a conflict) that either its application or violation lead to. Therefore, norm evaluation will solely consider the applicable norms that have been either applied or violated in the transition from two consecutive states in time.

Norm evaluation is performed by steps 5 to 7 in algorithm 1. Function $normApplicability$ (line 5) uses function f_{apply} to assess the norms in the normative network (NN) that were applicable at tick $t-1$. Thus, for each view $\langle \nu^i_{s_{t-1}}, \nu^i_{s_t} \rangle$, this function assesses the norms that are applicable at view $\nu^i_{s_{t-1}}$. Next, function $normCompliance$ (line 6) partitions the selected applicable norms into applied or violated norms. Moreover, it uses a conflict-detection function ($f_{conflict}$) to determine which norms led to conflicts during the last time step. As a result we obtain a partition of applicable norms into four multi-sets (sets that allow duplicate values): *(i)* applied norms that led to conflicts (A_C); *(ii)* applied norms that did not lead to conflict ($A_{\bar{C}}$); *(iii)* violated norms that led to conflicts (V_C); and *(iv)* violated norms that did not lead to conflict ($V_{\bar{C}}$). In the algorithm $A = (A_C, A_{\bar{C}})$ and $V = (V_C, V_{\bar{C}})$.

At this point we can evaluate norms. The strategy uses function $updateUtilities$ (line 7) to compute the effectiveness and necessity of each norm at time t. On the one hand, we measure the effectiveness of *applied* norms based on their outcomes. In fact, we evaluate the *cumulative* effectiveness of a norm according to the following principle: the higher the ratio of *successful applications* (applications not leading to conflicts) of a norm, the higher the effectiveness increase. We compute the effectiveness of norm n up to time t as:

$$\mu_{\mathit{eff}}(n,t) = (1-\alpha) \times \mu_{\mathit{eff}}(n,t-1) + \alpha \times r_{\mathit{eff}}(n,t) \quad (1)$$

where: $\mu_{\mathit{eff}}(n,t-1)$ stands for the effectiveness of n at time $t-1$; $0 \leq \alpha \leq 1$ is a parameter to trade off exploitation (of the effectiveness obtained so far until $t-1$) with exploration (of the effectiveness reward obtained from $t-1$ to t) in the cumulative effectiveness; and $r_{\mathit{eff}}(n,t)$ is a reward value based on the successful applications of norm n. We assess a norm's effectiveness reward as follows:

$$r_{\mathit{eff}}(n,t) = \frac{w_{A_{\bar{C}}} \times m_{A_{\bar{C}}}(n)}{w_{A_{\bar{C}}} \times m_{A_{\bar{C}}}(n) + w_{A_C} \times m_{A_C}(n)} \quad (2)$$

where $m_{A_{\bar{C}}}(n)$ stands for the number of applications of norm n that did not end up with conflicts, $m_{A_C}(n)$ stands for the number of applications of norm n that led to conflicts, and $w_{A_{\bar{C}}} > 0$, $w_{A_C} > 0$ are weights that measure the importance of successful applications and unsuccessful applications of n respectively. Notice therefore that our approach is akin to reinforcement learning.

On the other hand, we measure the necessity of *violated* norms based also on their outcomes. Analogously to the above-described approach, we assess the *cumulative* necessity of a norm according to the following principle: the higher the ratio of *harmful violations* (violations leading to conflicts), the more necessary the norm. We compute the necessity of norm n up to time t as:

$$\mu_{nec}(n,t) = (1-\beta) \times \mu_{nec}(n,t-1) + \beta \times r_{nec}(n,t) \quad (3)$$

where: $\mu_{nec}(n,t-1)$ stands for the necessity of n at time $t-1$; $0 \leq \beta \leq 1$ is a parameter to to trade off exploitation (of the necessity obtained so far until $t-1$) with exploration (of the necessity reward obtained from $t-1$ to t) in the cumulative necessity; and $r_{nec}(n,t)$ is a reward value based on the harmful violations of norm n. We assess a norm's necessity reward as follows:

$$r_{nec}(n,t) = \frac{w_{V_C} \times m_{V_C}(n)}{w_{V_C} \times m_{V_C}(n) + w_{V_{\bar{C}}} \times m_{V_{\bar{C}}}(n)} \quad (4)$$

where $m_{V_C}(n)$ stands for the number of violations of norm n that led to conflicts, $m_{V_{\bar{C}}}(n)$ stands for the number of violations of norm n that did not lead to conflicts, and $w_{V_C} > 0$, $w_{V_{\bar{C}}} > 0$ are weights that measure the importance of harmful applications and harmless applications of n respectively.

3.4.3 *Normative system refinement*

The last task of our strategy is the *normative system refinement*, which yields a new normative system by transforming the normative network via specialisations and generalisations. With this aim, the strategy keeps track of the effectiveness and necessity of the norms in the normative network during a period of time T. Then, the refinement task amounts to implementing the following rules:

- A norm is *specialised* (or *deactivated* if it has no children in the normative network) provided that either its effectiveness *or* necessity have not been good enough during T. This occurs when the effectiveness *or* necessity of some of its children have not been good enough either.
- A set of norms are *generalised* provided that: (i) they all relate to the very same norm (parent) in the normative network; (ii) they are all the possible child norms of the parent norm; (iii) their effectiveness *and* necessities have all been good enough during T.

On the one hand, we say that a norm n has not been good enough within period of time T, and hence can be specialised if any of the following conditions hold:

$$\bar{\mu}_{\mathit{eff}}(n,T) + \hat{\mu}_{\mathit{eff}}(n,T) < \alpha^{deact}_{\mathit{eff}} \quad (5)$$

$$\bar{\mu}_{nec}(n,T) + \hat{\mu}_{nec}(n,T) < \alpha^{deact}_{nec} \quad (6)$$

where: $\bar{\mu}_{\mathit{eff}}(n,T)$ and $\hat{\mu}_{\mathit{eff}}(n,T)$ stand for the average and deviation of the effectiveness of n within T; $\bar{\mu}_{nec}(n,T)$ and $\hat{\mu}_{nec}(n,T)$ stand for the average and deviation of the necessity of n within T; $\alpha^{deact}_{\mathit{eff}} \in [0,1]$ and $\alpha^{deact}_{nec} \in [0,1]$ stand for the *deactivation thresholds* for effectiveness and necessity and both $\alpha^{deact}_{\mathit{eff}}$, $\alpha^{deact}_{nec} \in \Theta$.

On the other hand, we say that a norm n has been good enough within T, and hence might be generalised to its parent if the following conditions hold:

$$\bar{\mu}_{\mathit{eff}}(n,T) - \hat{\mu}_{\mathit{eff}}(n,T) \geq \alpha^{gen}_{\mathit{eff}} \quad (7)$$

$$\bar{\mu}_{nec}(n,T) - \hat{\mu}_{nec}(n,T) \geq \alpha^{gen}_{nec} \quad (8)$$

where $\alpha^{gen}_{\mathit{eff}} \in [0,1]$ and $\alpha^{gen}_{nec} \in [0,1]$ stand for the *generalisation thresholds* for effectiveness and necessity and both $\alpha^{gen}_{\mathit{eff}}$, $\alpha^{gen}_{nec} \in \Theta$.

Next, we detail how algorithm 1 implements specialisations and generalisations respectively. We start considering the specialisation of a norm that is not performing well enough. This amounts to: (i) deactivating the norm along with its child norms that are not performing good enough either; and (ii) activating the child norms that are performing good enough. First, function $updatePerformances$ (line 8) computes the upper and lower effectiveness and necessity *performances* for each norm in the normative network, namely $\bar{\mu}_{\mathit{eff}}(n,T) + \hat{\mu}_{\mathit{eff}}(n,T)$, $\bar{\mu}_{\mathit{eff}}(n,T) - \hat{\mu}_{\mathit{eff}}(n,T)$, $\bar{\mu}_{nec}(n,T) + \hat{\mu}_{nec}(n,T)$, and $\bar{\mu}_{nec}(n,T) - \hat{\mu}_{nec}(n,T)$. Next, for each norm n that was either applied or violated during the last state transition, function $isUnderperforming$ (line 10) checks whether it satisfies the deactivation conditions (equations 5 and 6) or not. If so, the strategy calls function $specialiseDown$ (line 11) to specialise norm n in the normative network. If the norm is a leaf in the normative network (line 22), it is simply deactivated (line 23) using the *deactivate* operator. Otherwise, if the norm has children, the function triggers the specialisation down the normative network: first, it specialises the norm to its children using the *specialise* operator (line 26); second, the function calls itself recursively to specialise the children. Notice that lines 27-29 guarantee that those children whose effectiveness *or* necessity have not been good enough are deactivated.

Finally we consider the generalisation of a set of norms. In case a norm n must not be specialised (because it is good enough), the strategy considers whether the norm can be generalised (lines 12-17). Function $validGeneralisations$ (line 13) searches in the normative network for valid generalisations involving norm n. A valid generalisation is composed of a parent norm n' such that $(n, n') \in R_G$ and all possible siblings of n (obtained from the grammar $\mathcal{G}$) whose parent is also n' . Furthermore, each of the child norms in a valid generalisation must be active and satisfy the generalisation conditions of equations 7 and 8. For each valid generalisation, the strategy applies the *generalise* operator (line 17) to the parent and child norms.

3.5 Evaluating normative systems

We assess the effectiveness and necessity of a normative system Ω as a whole over a period of time $T = [t_\omega, t]$ based on the average effectiveness and average necessity of each of its norms over a period T. Then:

$$\mu_{eff}(\Omega, T) = \frac{\sum_{n\in\Omega} \bar{\mu}_{eff}(n,T)}{|\Omega|} \quad \mu_{nec}(\Omega, T) = \frac{\sum_{n\in\Omega} \bar{\mu}_{nec}(n,T)}{|\Omega|} \tag{9}$$

These measures are employed by IRON to determine whether a normative system Ω is good enough as a solution to the NSP in definition 3. This occurs whenever $\mu_{eff}(\Omega, T) \geq \alpha_{eff}$ and $\mu_{nec}(\Omega, T) \geq \alpha_{nec}$.

3.6 Architecture and computational model

We now have all components for the architecture and computational model of our norm synthesis system. Figure 2 illustrates the architecture of IRON. As we mentioned, IRON continuously searches *on-line* for a solution to the NSP, namely during the operation of a NA-MAS. We regard IRON as an external observer of agents' interactions. Moreover, we consider that such perceptions are limited to partial views of the global state of the NA-MAS.

IRON receives as an input (i) a function ($f_{conflict}$) to detect conflicts in the partial views it perceives; (ii) a grammar $\mathcal{G}$ to define norms; (iii) a function to determine whether a norm applies to the agents in a given view (f_{apply}); (iv) evaluation functions to compute the effectiveness (μ_{eff}) and necessity (μ_{nec}) of norms and normative systems; (v) the satisfaction degrees and thresholds(Θ); as well as (vi) the time interval (T) considered when solving the NSP.

Our norm synthesis mechanism is composed of: (i) a *normative network* (NN) to compactly represent the current normative system and to store the norms synthesised (explored) so far; (ii) a control unit in charge of directing the NSP solving. The control unit continuously perceives the NA-MAS to regulate by collecting partial views. After collecting views, the control unit calls the strategy Π described in Section 3.4 to apply a collection of operators and to eventually produce a new normative system that prevents the conflicts observed in the views. The normative system (Ω) is broadcast to the agents in the NA-MAS. Once the new normative system is deployed, the control unit collects new partial views of the NA-MAS to be analysed by the strategy. This cyclic process continues until the control unit receives from the strategy a normative system that is evaluated effective and necessary enough, according to the evaluation functions and satisfaction degrees set as input, during a period of time T. Such normative system will represent a solution to the NSP.

Finally, we notice that, in general, the size of the search space to explore is at most 2^m, where m is the number of norms defined by grammar $\mathcal{G}$.

4. EMPIRICAL EVALUATION

In this section we empirically demonstrate that IRON successfully manages to solve the NSP. Moreover, we also show that IRON can solve the NSP despite a high non-compliant behaviour in the agent population.

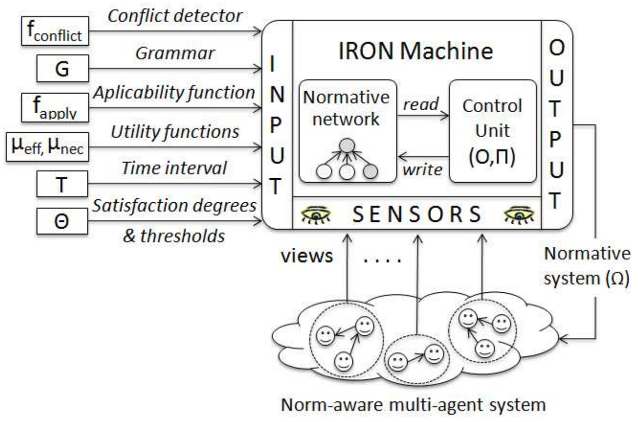

Figure 2: IRON's architecture.

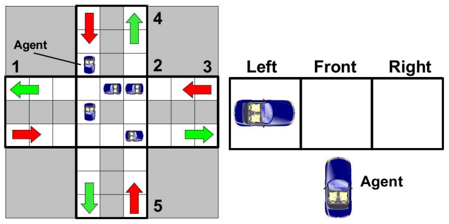

Figure 3: Left: Junction. Right: Agent context.

4.1 Empirical settings

Our experiments simulate a traffic junction composed of two orthogonal roads represented by a 19×19 grid. Each road has two 19-cell lanes (one per direction). Figure 3 shows the centre of the junction. Each agent is a car that travels along the grid at one cell per tick by following a random trajectory (i.e., random entry and exit points). In order to favour a high frequency of collisions, we cause a high traffic density (from 41% to 48% of occupied cells) by having three cars entering the scenario every tick. At each tick, each car decides whether to apply or violate norms. The probability of violating norms is fixed at the beginning of each simulation and is the same for all cars.

Each experiment consists of a set of simulations until either IRON converges to a stable normative system, solving the NSP, or the simulation reaches 50,000 ticks. IRON starts each simulation with an empty normative system. As the simulation goes on, collisions among cars occur, and IRON synthesises a new normative system to avoid future collisions. We consider that IRON has converged to a normative system if during a 10,000-tick period: *(i)* no collisions occur; and *(ii)* the normative system remains unchanged.

Each norm is represented as 3 cells (see Figure 3), which represent the context of a reference car. Cells may have 6 values each: either a car with its heading (>, <, ∨, ∧); a "-", representing the value of "nothing"; or the empty value (standing for a generalisation). Therefore, the grammar that we employ can synthesise $6^3(216)$ different norms and the number of normative systems to consider amounts to 2^{216} (larger than 10^{65}). We set IRON's parameters as follows: *(i)* low deactivation thresholds ($\alpha_{eff}^{deact} = \alpha_{nec}^{deact} = 0.2$) to only deactivate norms performing very poorly; *(ii)* high generalisation thresholds ($\alpha_{eff}^{gen} = \alpha_{nec}^{gen} = 0.6$) to only generalise norms when performing very well; *(iii)* $w_{A_C} = 5$ and $w_{A_{\bar{C}}} = 1$ to ensure that norm applications leading to collisions (ineffective norms)

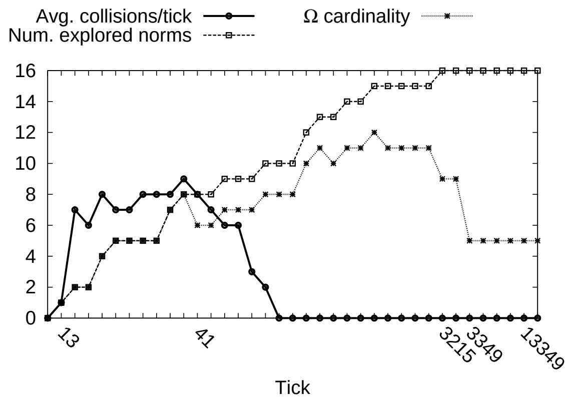

Figure 4: Norm synthesis along time.

are much more penalised than those avoiding collisions (effective norm); *(iv)* $w_{V_C} = 2$ and $w_{V_{\bar{C}}} = 1$ to ensure that violations leading to collisions (necessary norm) obtain a much higher reward than those leading to no collisions (unnecessary norm).

4.2 Experiment 1: Norm Synthesis

Next, we show that IRON manages to successfully synthesise norms which are effective and necessary within an acceptable range of values. First, we performed 100 simulations with a 30% norm violation rate, namely 30% of agent's decisions do not comply with the norms output by IRON. As Figure 5 shows, IRON successfully converged for all simulations to a normative system that avoids collisions as long as cars comply with norms. Regarding the quality of the resulting normative systems, the average effectiveness was high (89.73%), and the necessity as well (73.15%). One may think that, once IRON converges, the effectiveness should be 100%. This is not the case because of the way we evaluate norms when collisions occur. Notice that when several norms are involved in a collision (because of their application or violation), we consider that all of them led to the collision because we cannot distinguish which ones actually did. Therefore, we take a very conservative evaluation approach in this case.

Figure 4 shows a prototypical simulation (out of the 100 performed), with 30% violations, to analyse IRON's synthesis process. At tick 13, the first collision arises and IRON synthesises the first norm. From that tick onwards, IRON keeps generating norms when needed. At tick 41, IRON performs the first norm generalisation, reducing the cardinality of the normative system from 9 to 8 norms. At tick 3215, IRON synthesises the last norm. By using the resulting normative system, cars do not collide any longer provided that cars comply with norms. IRON performs the last generalisation at tick 3349, reducing the cardinality of the normative system to 4 norms. From tick 3349 onwards, the normative system remains stable. After 10000 further ticks, IRON reaches convergence (tick 13349). At the end of the simulation, IRON explored 16 different norms (out of 125 possible ones), which were generalised into 4 norms, and 20 different normative systems (out of 2^{125} possible ones) to find a 4-norm normative system that successfully prevents collisions as long as cars comply with norms.

The 4-norm normative system that IRON converged to is depicted in table 2. Norm n_1 is a left-hand side priority specifying that a car must stop when it perceives a car to its left heading to its right, and no matter what it perceives in front or to its right. It has very high values of effectiveness (0.86) and necessity (0.90), what makes the norm to be *essential*.

Norm n_4 forces a car to stop when it perceives a car in front

Norm	Pre-condition (θ)	Modality	$\bar{\mu}_{eff}$	$\bar{\mu}_{nec}$
n_1	*left*(>)	$obl(stop)$	0.86	0.90
n_2	*left*(<)&*front*(<)	$obl(stop)$	0.87	0.73
n_3	*front*(>)&*right*(>)	$obl(stop)$	0.86	0.81
n_4	*front*($\wedge$)	$obl(stop)$	0.83	0.33

Table 2: A normative system upon convergence.

heading forward, which is a situation that exceptionally leads to collisions. Therefore, when violated, this norm is exceptionally evaluated as necessary, whereas most times it is evaluated as unnecessary. As a result, its necessity continuously oscillates, with a low average value ($\bar{\mu}_{nec}$). We say this norm is *preventive*, since agents should comply with it "just in case" if we want to totally remove collisions.

To conclude, we showed that IRON can successfully synthesise a normative system with high effectiveness and reduced cardinality. Moreover, we observe that the synthesised norms in the resulting normative systems are either *essential* (high effectiveness, high necessity) or *preventive* (high effectiveness, low necessity).

4.2.1 IRON's regulation versus traffic lights

Now we compare IRON with an alternative way of regulating traffic (traffic lights) as done in [3]. We simulated a traffic junction regulated by 4 traffic lights, one per lane coming from the 4 cardinal points. There are 4 green light turns. When a light changes to green, allowing cars to pass, the other three lights turn red, forcing cars to stop. Thus, traffic lights avoid collisions by giving pass to the cars of one unique lane at the same time. Since we observed that both IRON and traffic lights are 100% effective to prevent collisions, we compare them in terms of traffic fluidity. Table 3 compares traffic fluidity between traffic lights and the normative system found by IRON in the previous example.

	Expected time	Average time	Delay
IRON	19	27.073	42.48%
Traffic lights	19	46.094	142.6%

Table 3: IRON's synthesised norms versus traffic lights.

Cars are expected to reach their destinations in 19 ticks average. However, both IRON and traffic lights eventually require stops to avoid collisions. Thus, cars following IRON norms invest 27.073 ticks on average to reach their destinations, while cars regulated by traffic lights invest 46.094 ticks on average. Therefore, cars following IRON's norms are delayed three times less than they do when following the traffic lights (42.48% average vs. 142.6% average). Thus, IRON's synthesised norms are as effective as traffic lights, but outperform them in terms of traffic fluidity.

4.3 Experiment 2: Robustness analysis

Next we explore the limits of IRON by testing its synthesis capabilities under different cars' violation rates. Violation rates (i.e., the probability of each car violating a norm) ranged from 10% to 90%. We performed 100 simulations per violation rate. Fig. 5 shows averaged results for the effectiveness and necessity of the synthesised normative systems[2]. Moreover, the *convergence* series

[2]For the sake of clarity, we do not plot standard deviations. However, it is worth mentioning that the standard deviations for the effectiveness and necessity for each violation rate are within $[0.006, 0.011]$ and $[0.080, 0.0137]$ respectively.

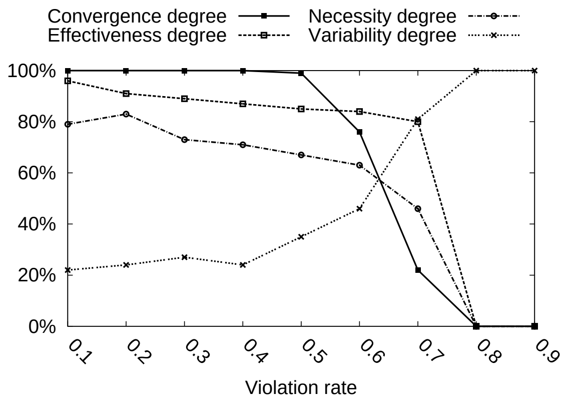

Figure 5: Robustness analysis depending on violations.

shows the number of runs that converged to a stable normative system. With violation rates up to 50% IRON successfully converged 100% of the times to highly effective and necessary normative systems. Between 50% and 80% of violation rate, the convergence decreases (due to oscillations in the normative systems) until IRON cannot find a normative system beyond 80%.

Figure 5 shows the variability of IRON's synthesis, namely whether it yields different normative systems. Below the 50% violation rate, the variability remains near 20 (i.e., 100 executions converged to 20 different normative systems). Since preventive norms become unstable (activated and deactivated back and forth) with high violation rates, IRON takes longer to synthesise stable norms. This leads to new, different normative systems, which were not required to be explored with lower violation rates.

Overall, IRON proved to be highly resilient to non-compliant behaviours during the synthesis process. IRON managed to successfully synthesise norms despite up to a 50% violation rate of agents.

5. CONCLUSIONS AND FUTURE WORK

In this paper we have described progress towards the automated synthesis of normative systems. Firstly, we formally introduced the norm synthesis problem. Secondly, we described IRON, a novel mechanism for the on-line synthesis of norms. IRON produces norms that guarantee conflict-free coordination, while avoiding over-regulation. For this purpose, IRON employs norm effectiveness and necessity as the measures that characterise the quality of a normative system. Furthermore, we endow IRON with the capability of generalising norms. By keeping effective norms, generalising norms, and discarding unnecessary norms, we allow IRON to yield *effective* and *concise* normative systems.

The machinery of IRON has been designed to operate: on-line (while observing the NA-MAS to regulate); conflict-driven (conflicts trigger the evolution of the normative system); and without requiring any prior normative knowledge. Moreover, IRON uses a data structure (the normative network) that represents the generated norms and their relationships. Finally, we empirically show that IRON successfully synthesises norms even in the presence of non-compliance behaviours in a MAS, being resilient to a very high percentage of violations (up to 50%).

As future work, we plan to: (i) empirically study the sensitivity of IRON to its parameters; (ii) investigate further relationships between norms in the normative network; and (iii) extend the norm synthesis process to detect the cause of conflicts within a sequence of previous states instead of a single state as we do now.

6. ACKNOWLEDGEMENTS

This work was funded by AT (CONSOLIDER CSD2007-0022), EVE (TIN2009-14702-C02-01,TIN2009-14702-C02-02), COR (TIN2012-38876-C02-01, TIN2012-38876-C02-02), MECER (201250E053) and the Generalitat of Catalunya (2009-SGR-1434). Michael Wooldridge was supported by the ERC under Advanced Grant 291528 (“RACE").

7. REFERENCES

[1] T. Agotnes and M. Wooldridge. Optimal Social Laws. In *Proceedings of the International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, pages 667–674, 2010.

[2] G. Christelis and M. Rovatsos. Automated norm synthesis in an agent-based planning enviroment. In *Proceedings of the 8th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 161–168, 2009.

[3] K. Dresner and P. Stone. A multiagent approach to autonomous intersection management. *Journal of Artificial Intelligence Research (JAIR)*, 31:591–656, March 2008.

[4] D. Fitoussi and M. Tennenholtz. Minimal social laws. In *Proceedings of the National Conference on Artificial Intelligence*, pages 26–31. John Wiley & Sons LTD, 1998.

[5] A. García-Camino, J. A. Rodríguez-Aguilar, C. Sierra, and W. Vasconcelos. Constraint rule-based programming of norms for electronic institutions. *JAAMAS*, 18(1):186–217, 2009.

[6] N. Griffiths and M. Luck. Norm Emergence in Tag-Based Cooperation. In *9th International Workshop on Coordination, Organization, Institutions and Norms in Multi-Agent Systems. 79-86*, 2010.

[7] J. Morales, M. López-Sánchez, and M. Esteva. Using Experience to Generate New Regulations. In *International Joint Conference in Artificial Intelligence*, pages 307–312. AAAI Press, USA, 2011.

[8] N. Salazar, J. A. Rodriguez-Aguilar, and J. L. Arcos. Robust coordination in large convention spaces. *AI Commun.*, 23(4):357–372, Dec. 2010.

[9] B. Savarimuthu, S. Cranefield, M. Purvis, and M. Purvis. Role model based mechanism for norm emergence in artificial agent societies. *Lecture Notes in Computer Science*, 4870:203–217, 2008.

[10] O. Sen and S. Sen. Effects of social network topology and options on norm emergence. In *Proceedings of the 5th international conference on Coordination, organizations, institutions, and norms in agent systems*, COIN'09, pages 211–222, Berlin, Heidelberg, 2010. Springer-Verlag.

[11] S. Sen and S. Airiau. Emergence of norms through social learning. In *Proceedings of the 20th international joint conference on Artifical intelligence*, IJCAI'07, pages 1507–1512, San Francisco, CA, USA, 2007. Morgan Kaufmann Publishers Inc.

[12] Y. Shoham and M. Tennenholtz. On social laws for artificial agent societies: off-line design. *Journal of Artificial Intelligence*, 73(1-2):231–252, February 1995.

[13] D. Villatoro, J. Sabater-Mir, and S. Sen. Social instruments for robust convention emergence. In T. Walsh, editor, *Proceedings of the 22nd International Joint Conference on Artificial Intelligence*, pages 420–425. IJCAI/AAAI, 2011.

[14] M. Wooldridge. *An Introduction to MultiAgent Systems*. John Wiley & Sons, 1st edition, June 2002.