The Impact of Norms Generality on MAS Goal

Jhonatan Alves , Jomi Fred Hübner , Jerusa Marchi

[1]Federal University of Santa Catarina (UFSC)
Florianópolis – Brazil

jhonatan.alves@posgrad.ufsc.br,{jomi.hubner,jerusa.marchi}@ufsc.br

Abstract. *In this paper we aim to investigate the impact of levels of generality on norms synthesized on the global goal satisfiability and conflicts avoidance of Normative Multiagent Systems. We found out that exists a level of generality in the scenarios we perform some experiments, which we call* apex level, *where norms tend to be efficient in avoiding conflicts while ensuring goal system reachability.*

1. Introduction

Norms are a topic of interest in many research areas. In Multiagent systems (MAS) they are generally understood as rules which, in certain contexts, impose restrictions on agent's behavior indicating which actions are allowed, prohibited or obligatory to achieve social order [Boella and van der Torre 2004]. Among the topics investigated on the norms subject in the MAS' field, we have the *synthesis* which refers to the process of creating a set of norms to regulate the agents. Basically, norms can be synthesized in two different ways: *manually* - where a designer produces the set of norms; or *automatically* - where a computational mechanism processes such synthesis [Frantz and Pigozz 2018].

Although the automated synthesis may provide certain advantages as time saving and standardization of results, it is a complex problem [Shoham and Tennenholtz 1995] and some important issues must be considered in this process to the resulting norms regulate the agents efficiently balancing control and autonomy. One of those problems is the synthesis of *mildly restrictive norms*, which barely regulate the agents giving them much freedom to act as they wish and, consequently, allowing conflicts during the system execution. Another problem is the synthesis of *strict norms* which, in turn, may excessively regulate the agents giving them low freedom to perform their actions preventing them from executing the necessary actions for the system fulfills its goal.

Given two norms n_1 and n_2, we say that norm n_1 is more generic than norm n_2 if it regulates the behaviors that norm n_2 does, however the opposite does not occur. For example, a norm which prohibits agents to move between two adjacent vertices in a grid is more generic than a norm which prohibits only the agent *B*ob to perform such action. The norm n_1 prohibits all agents to move, thus the behaviors regulated by norm n_2 are regulated by norm n_1 as well. Such norm is a *strict* one and turns the system goal unreachable since the agents get stuck in their depart vertices. On the other hand, norm n_2 is less restraining, since just Bob cannot move, thus the set of behaviors regulated by norm n_2 does not include those regulated by norm n_1. Norm n_2 is *mildly restrictive* and it is not capable to avoid conflicts since when the other agents move collisions may occur.

In this sense, exploring the *generality of norms* is a fundamental key to obtain a set of norms which constrains the agents avoiding conflicts and making the system goal feasible to be reached. Although different mechanisms for the automated norm synthesis have been proposed [Onn and Tennenholtz 1997, Boella and van der Torre 2007, Savarimuthu et al. 2008, Christelis and Rovatsos 2009, Morales et al. 2011], just a few of them [Fitoussi and Tennenholtz 1998, Morales et al. 2013, Morales et al. 2014] consider such generality as part of the synthesis scope to measure how generic norms must be to obtain norms that are not mildly restrictive nor too restraining.

In this paper our aim is to investigate *how the generality of norms impacts on the system's goals reachability*. For this, we propose a model in which the norms are organized in levels of generality so that it is possible to correlate the generalities with the capability to avoid conflicts and keep the objectives of the system attainable. According to our experiments, there exists at least one intermediate level of generality, which we call *apex level*, where some norms are capable to maximize the goal reachability and minimize the conflicts' occurrence. We intend to use this information to posteriorly build a synthesis algorithm which use them as a heuristic to guide the process to obtain norms whose level of generality corresponds to the *apex level*.

The remainder of this paper is structured as follows. In Section 2 we introduce our model for norms generalization. In section 3 we present two algorithms to norm synthesis and norm evaluation. In section 4 we look for an answer to our research question, by presenting a set of tests which explore the generality of norms in order to figure out the existence of a level of generality where norms may regulate the agents efficiently. In Section 5 related works are presented and discussed. We also point out a possible and hypothetical direction in which we may overcome them. Finally, in Section 6 we present our conclusion and future works.

2. A Model for Norms Generalization

Before introducing our model consider the following toy scenario which will be used along the text to exemplify some of its concepts. The scenario is composed by a simple grid domain $m \times n$ where intersections represent vertices and edges represent paths which connect such vertices. There are two agents on the grid, $\texttt{ag}_1$ and $\texttt{ag}_2$ which must depart from an initial vertex to a goal one executing their traversal plans without getting in conflicts. A conflict is characterized as a collision occurring when an agent moves to a local occupied by another agent. Moreover, agents may just move between vertices which are adjacent.

Definition 1 (Language). *Let $\mathcal{L}$ be a restrict form of a first order language, defined as a 5-uple* $\langle \mathrm{Pred}_\mathcal{L}, \mathrm{Var}_\mathcal{L}, \mathrm{Const}_\mathcal{L}, \mathrm{Types}_\mathcal{L}, \mathrm{Conec}_\mathcal{L} \rangle$*, where* $\mathrm{Pred}_\mathcal{L} = \{pred_1, \ldots, pred_m\}$ *is a set of predicate symbols* $\cup \{False\}$*,* $\mathrm{Var}_\mathcal{L} = \{v_1, \ldots, v_n\}$ *is a set of typed variable symbols,* $\mathrm{Const}_\mathcal{L} = \{c_1, \ldots, c_k\}$ *is a set of typed constant symbols,* $\mathrm{Types}_\mathcal{L} = \{t_1, \ldots, t_p\}$ *is a set of types and* $\mathrm{Conec}_\mathcal{L} = \{\wedge\}$ *is a set of logical connectives.*

Each symbol of predicate is associated with a finite set of terms. A term is a variable or a constant symbol associated with a given type of $\mathrm{Types}_\mathcal{L}$, such that, they form the set $\mathrm{Terms} = \mathrm{Var}_\mathcal{L} \cup \mathrm{Const}_\mathcal{L}$. In special, there is one predicate with zero terms: *False* meaning false. The well-formed formulas are given by the formation rules of the first-order logic considering the connectives in $\mathrm{Conec}_\mathcal{L}$. It is established that variables begin with capital letters while constants begin with lowercase ones and terms described with distinct symbols represent distinct objects.

Example 1. *Consider the grid domain previously introduced, the formula* $\texttt{next(L}_1\texttt{,L}_2\texttt{)}$ *represents that an arbitrary vertex* $\texttt{L}_1$ *is adjacent to another arbitrary vertex* $\texttt{L}_2$ *while the formula* $\texttt{at(ag}_1\texttt{,L}_1\texttt{)}$ *represents that the specific agent* $\texttt{ag}_1$ *is at the vertex* $\texttt{L}_1$.

Definition 2 (System State). *Let* $\mathcal{P} = \{p_1,\ldots,p_m\}$ *be a set of all ground atomic formulas with* $\mathcal{P} \subset \mathcal{L}$ *and* $\mathcal{S} = \{s_1,\ldots,s_n\}$ *be a set of system states, where* $\mathcal{S} \subseteq 2^{\mathcal{P}}$. *A state* $s_i = \{p_1,\ldots,p_k\}$ *is a set of ground atomic formulas which holds in a given time i.*

The states are built according to the hypothesis of closed world (predicates that do not make part of a given state are considered false). We may say that the special predicated False is prohibited to take part of any system state. As we will see, it is used as part of norms definition and has impact on the norm generality.

Definition 3 (Action). *Let* $\mathcal{A} = \{a_1,\ldots,a_m\}$ *be a set of unground actions. An action* $a_i = \langle \mathrm{Id}, \mathrm{Par}, \mathrm{Pre}, \mathrm{Add}, \mathrm{Del} \rangle$ *is a 5-uple, such that,* Id *is a identifier,* $\mathrm{Par} = \{par_1,\ldots,par_n\}$ *is a set of parameters with* $\mathrm{Par} \subset \mathrm{Var}$, $\mathrm{Pre} = \{p_1,\ldots,p_k\}$ *is a set of preconditions which must hold to* a_i *be executed,* $\mathrm{Add} = \{p_1,\ldots,p_j\}$ *is a set of effects which start to hold after the action execution and* $\mathrm{Del} = \{p_1,\ldots,p_l\}$ *is a set of effects which does not hold anymore after the action execution, where* Pre, Add, Del $\subset \mathcal{L}$ *are unground atomic formulas. Let* $\mathcal{A}' = \{a_1,\ldots,a_z\}$ *be a set of ground actions, such that, an action* a_i' *is a copy of* a_i *where its parameters are constants of* $\mathrm{Const}_{\mathcal{L}}$ *and its sets are formed by ground atomic formulas.*

From the set $\mathcal{A}$ partial and grounded actions may be obtained which, in turn, are important to synthesise norms with different levels of generality (see algorithm 1 in section 3). However, to act in a environment the agents perform grounded actions of $\mathcal{A}'$.

Example 2. *To* `move` *between vertices, the agents may perform a grounded version of action* $\langle\texttt{move}, \{\texttt{Ag}_1\texttt{,L}_1\texttt{,L}_2\},\{\texttt{at(Ag}_1\texttt{,L}_1\texttt{),next(L}_1\texttt{,L}_2\texttt{)}\},\{\texttt{at(Ag}_1\texttt{,L}_2\texttt{)}\},\{\texttt{at(Ag}_1\texttt{,L}_1\texttt{)}\}\rangle$.

According to the action `move`, to an agent $\texttt{Ag}_1$ move from a vertex $\texttt{L}_1$ to a vertex $\texttt{L}_2$, such agent must be at $\texttt{L}_1$ which, in turn, must be adjacent to $\texttt{L}_2$. After executing `move`, the agent will be at vertex $\texttt{L}_2$ and the agent won't be longer at the former vertex. We will use $Par(a)$, $Pre(a)$, $Add(a)$ and $Del(a)$ along the text to refer that such sets are components of a given action $a \in \mathcal{A} \cup \mathcal{A}'$. However, in the examples we will use the a's signature[1] to refer to it.

Definition 4 (MAS Domain). *Let* $\mathcal{D} = \langle \mathcal{S}, \mathcal{A}', \gamma \rangle$ *be a domain of a MAS, such that,* $\mathcal{S}$ *denotes a set of system states,* $\mathcal{A}'$ *denotes a set of (grounded) agent's actions and* $\gamma : \mathcal{S} \times \mathcal{A}' \to \mathcal{S}$ *denotes a deterministic and discrete transition function between system states, such that,* $\gamma(s,a) = (s - Del(a)) \cup Add(a)$.

Other important concept of our model is conflict. The notion of conflict appears in a variety of areas, and it is defined by reflecting the own characteristics of each one. Conceptually, in this paper we consider a conflict as an event which may cause negative impacts on the system as avoiding agents to accomplish their tasks and compromising the system goal reachability. Formally, we define a conflict in terms of its effects.

Definition 5 (Conflict). *Let* $\mathcal{C} = \{c_1,\ldots,c_m\}$ *be a set of conflicts which may occur during the system execution, a conflict* c_i *is defined as a set of atomic formulas which denotes its effects, such that,* $c_i = \{p_1,\ldots,p_n\}$, *where* $c_i \subset \mathcal{L}$.

[1]A signature is a concatenation between the a action's identifier, an open parentheses, the action's parameters and a close parentheses with no curly braces as $\texttt{move(Ag}_1\texttt{,L}_1\texttt{,L}_2\texttt{)}$, for example.

Example 3. *A conflict which establishes a collision between agents on the grid may be described as* $c_1 = \{\texttt{at}(\texttt{Ag}_1,\texttt{L}_1),\texttt{at}(\texttt{Ag}_2,\texttt{L}_1)\}$. *Thus, according to* c_1*, a collision occurs when two distinct agents are on the same vertex* $\texttt{L}_1$.

Definition 6 (Conflict Instances). *Given a conflict* $c_i \in \mathcal{C}$, $I(c_i) = \{c_i^1, \ldots, c_i^n\}$ *is a set of instances of* c_i*, where an instance* c_i^j *is a copy of* c_i *with ground atomic formulas.*

Example 4. *An instance of conflict* c_1*, from example 3, could be* $c_1^j = \{\texttt{at}(\texttt{ag}_1,\texttt{a}),\texttt{at}(\texttt{ag}_2,\texttt{a})\}$ *denoting that both agents* $\texttt{ag}_1$ *and* $\texttt{ag}_2$ *are on the vertex* $\texttt{a}$.

Definition 7 (Conflict State). *A given state* $s \in \mathcal{S}$ *is said to be a conflict state if* $\exists c_i \in \mathcal{C}$: $\exists c_i^j \in I(c_i)$*, such that,* $c_i^j \subset s$.

A way to avoid conflict states and preserve the system goal reachability is regulating the agents' behaviors through norms.

Definition 8 (Norm). *Let* $\mathcal{N} = \{n_1,\ldots,n_m\}$ *be a set of prohibitive norms, where a norm* n_i *is a rule of the form* $\varphi \underset{p}{\rightarrow} a$*, where* $\varphi = \{p_1, p_2, \ldots, p_n\}$ *is a set of atomic formulas as being a context activation with* $\varphi \subset \mathcal{L}$ *and* $a \in \mathcal{A} \cup \mathcal{A}'$. *Given a state* $s \in S$*, if* φ *holds in s, then the action a is prohibited to be executed in s.*

From now onward, we will use $\varphi(n_i)$ and $a(n_i)$ to refer, respectively, to the context activation and prohibited action of given a norm $n_i \in \mathcal{N}$. In the examples we refer to $a(n_i)$ by its *signature*.

Example 5. *A norm which may regulate the agents' behavior in order to avoid collisions on the grid may be defined as* $n_1 = \{\texttt{at}(\texttt{Ag}_2,\texttt{L}_2)\} \underset{p}{\rightarrow} \texttt{move}(\texttt{Ag}_1,\texttt{L}_1,\texttt{L}_2)$ *which, in natural language, it may be read as "if an agent* $\texttt{Ag}_2$ *is on a vertex* $\texttt{L}_2$*, then an agent* $\texttt{Ag}_1$ *is prohibited to move from a vertex* $\texttt{L}_1$ *to vertex* $\texttt{L}_2$*".*

Definition 9 (Norm Instance). *Given a norm* $n_i \in \mathcal{N}$, $\mathfrak{I}(n_i) = \{n_i^1,\ldots,n_i^m\}$ *is a set of instances of* n_i*, where an instance* $n_i^j = \varphi' \underset{p}{\rightarrow} a'$ *is a copy of* n_i *with* $\varphi' \subset \mathcal{L}$ *being a set of ground atomic formulas and* $Par(a') \subset Const$.

Example 6. *An instance of norm* n_1*, from example 5, is* $n_1^j = \{\texttt{at}(\texttt{ag}_2,\texttt{b})\} \underset{p}{\rightarrow} \texttt{move}(\texttt{ag}_1,\texttt{a},\texttt{b})$ *which prohibits agent* $\texttt{ag}_1$ *to move from the vertex* $\texttt{a}$ *to the vertex* $\texttt{b}$ *if agent* $\texttt{ag}_2$ *is on* $\texttt{b}$.

Definition 10 (Norm Applicability). *Given a state* $s \in S$ *and a norm* $n_i \in \mathcal{N}$*,* n_i *is said to be applicable (or active) in s if* $\exists n_i^j \in \mathfrak{I}(n_i)$*, such that,* $\varphi'(n_i^j), Pre(a'(n_i^j)) \subset s$.

Definition 11 (Norm Generality). *Let* $Sa(n_i)$ *and* $Sa(n_j)$ *be, respectively, the sets of system states in which the norms* $n_i, n_j \in \mathcal{N}$ *are applicable, with* $n_i \neq n_j$. *We say that* n_i *is more generic than* n_j*, denoted by* $n_i \underset{g}{>} n_j$*, if* $Sa(n_j) \subset Sa(n_i)$ *and* $Sa(n_i) \not\subset Sa(n_j)$.

Whether $n_i \underset{g}{>} n_j$, then n_i regulates all the behaviour that norm n_j does, however the opposite does not occur. Syntactically, we may say that $\varphi(n_i)$ subsumes $\varphi(n_j)$ and $a(n_i)$ subsumes $a(n_j)$.

Example 7. *Consider the following norms:*

$n_2 = \{\texttt{False}\} \underset{p}{\rightarrow} \texttt{move}(\texttt{ag}_1,\texttt{a},\texttt{b})$ $\quad$ $n_3 = \{\texttt{at}(\texttt{ag}_2,\texttt{b}),\texttt{next}(\texttt{a},\texttt{b})\} \underset{p}{\rightarrow} \texttt{move}(\texttt{ag}_1,\texttt{a},\texttt{b})$

$n_4 = \{\texttt{at}(\texttt{ag}_2,\texttt{L}_2),\texttt{next}(\texttt{L}_1,\texttt{L}_2)\} \underset{p}{\rightarrow} \texttt{move}(\texttt{ag}_1,\texttt{L}_1,\texttt{L}_2)$ $\quad$ $n_5 = \{\texttt{at}(\texttt{Ag}_2,\texttt{b}),\texttt{next}(\texttt{a},\texttt{b})\} \underset{p}{\rightarrow} \texttt{move}(\texttt{Ag}_1,\texttt{a},\texttt{b})$

$n_6 = \{\texttt{at}(\texttt{Ag}_2,\texttt{L}_2),\texttt{next}(\texttt{L}_1,\texttt{L}_2)\} \underset{p}{\rightarrow} \texttt{move}(\texttt{Ag}_1,\texttt{L}_1,\texttt{L}_2)$ $\quad$ $n_7 = \{\ \} \underset{p}{\rightarrow} \texttt{move}(\texttt{Ag}_1,\texttt{L}_1,\texttt{L}_2)$

The norm n_2 is the less generic norm since it does not apply to any system state. Thus, it is generalized by any other norm. On the other hand, since $\varphi(n_7)$ is empty, norm n_7 is applicable to all system states. Moreover, $a(n_7)$ subsumes all the other norm actions (since any substitution of its parameters to the parameters of the other actions is possible), then norm n_7 is the most generic norm. Considering the norm n_1 from example 5, we have two distinct orders of norm generalities (note that norms n_5 and n_4 are not comparable): i) $n_7 \underset{g}{>} n_1 \underset{g}{>} n_6 \underset{g}{>} n_5 \underset{g}{>} n_3 \underset{g}{>} n_2$; and ii) $n_7 \underset{g}{>} n_1 \underset{g}{>} n_6 \underset{g}{>} n_4 \underset{g}{>} n_3 \underset{g}{>} n_2$.

To each norm we assign a level of generality which is a value indicating how generic a norm is in a given partial order. The more generic a norm is, the bigger is its level of generality. Norms with certain properties in common are assigned with the same level of generality k. Such norms are generalized by those with they are comparable with a level $k+n$, with $n > 0$.

Definition 12 (Level of Generality). *The level of generality of a given norm $n \in N$, NG(n) is given by Equation 1*

$$NG(n) = \begin{cases} 0, & \text{if } \varphi(n) = \{False\} \\ \dfrac{1}{pred(n) + \frac{1}{1+var(n)}}, & \text{if } \varphi(n) = \{p_1, \dots, p_n\} \\ \dfrac{1}{1+const(n)} + 1, & \text{if } \varphi = \{\ \} \end{cases} \tag{1}$$

In Equation 1, pred(n), var(n) and const(n) refer, respectively, to the number of atomic formulas, variables and constants of a norm n. According to such equation, the levels of generality are distributed along the interval [0,2]. The levels increase from the norms whose activation context are False (NG = 0), going towards norms whose activation context are formed by a set of atomic formulas different from False (NG $\in$]0,1]) up to norms whose activation contexts are empty sets (NG $\in$]1,2] - norms that are always true). The more atomic formulas and constants a norm has, the more specific (and mildly restrictive) it is. However, as the number of atomic formulas decreases and the number of variables increases, the norms becomes more generic (and restraining). This way, mildly restrictive norms are assigned with levels of generality which put them more directed to the left of the interval [0,2], while the restraining ones are assigned with levels which put them more directed to the right of the given interval.

Example 8. *The Figure 1 illustrates a partial order formed by the norms of example 7. Such ordering is represented as a graph, where nodes correspond to norms and edges to relations of generalities. Whether $n_i \underset{g}{>} n_j$, then the edge depart from n_j to n_i.The values in red correspond to the levels of generality of each norm.*

Definition 13 (System Goal). *Let $\mathcal{G} = \{p_1, \dots, p_n\}$ be a set of ground atomic formulas as a declarative system goal, where $\mathcal{G} \subset \mathcal{L}$.*

Lastly, one can define what a Normative Multiagent System is in our perspective.

Definition 14 (Normative MAS). *Let nMAS = $\{Ag, \mathcal{D}, s_0, \mathcal{G}, \mathcal{N}, \mathcal{C}\}$ be a normative Multiagent System, such that, $Ag = \{ag_1, \dots, ag_m\}$ denotes a set of agents, $\mathcal{D}$ denotes a MAS domain, $s_0 \in S$ denotes a system initial state, $\mathcal{G}$ denotes a system goal, $\mathcal{N} = \{n_1, \dots, n_k\}$ denotes a set of norms and $\mathcal{C} = \{c_1, \dots, c_j\}$ denotes a set of conflicts.*

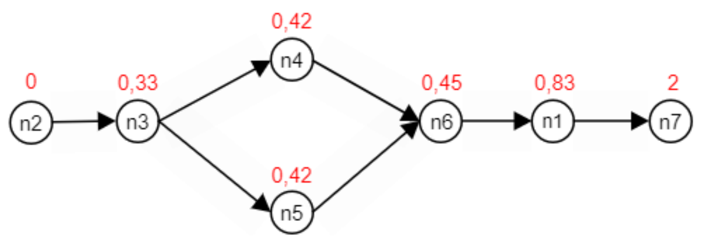

Figure 1. A partial order of norm generalities.

3. Norm Synthesis and System Execution

In this section we introduce the algorithms 1 and 2 which, respectively, synthesizes a set of norms and evaluates the norms' performance in the system execution. Basically, the algorithm 1 synthesises all ungrounded and grounded norms while the algorithm 2 groups the obtained results by level of generality. This way, we can explore the greatest amount of levels of generality and investigate how they impact on the system's goal reachability.

Algorithm 1 Norm Synthesis

```
Require: $\mathcal{M}$, $\mathcal{A}$, Pred $\subset \mathcal{L}$
1  $\mathcal{N}, \mathcal{N}' \leftarrow \emptyset$
2  $\mathcal{X} \leftarrow 2^{\text{Pred}} \times \mathcal{A}$
3  foreach $(\varphi, a) \in \mathcal{X}$ do
4      $\mathcal{N} \leftarrow \varphi \underset{p}{\rightarrow} a$
5  foreach $n \in \mathcal{N}$ do
6      $j \leftarrow$ getNumberOfTerms(n)
7      foreach $p \in [0, j]$ do
8          $\mathcal{N}' \leftarrow \mathcal{N}' \cup$
9          getOtherNorms($n$,$\mathcal{M}$,$p$)
10 return $\mathcal{N}'$
```

The algorithm 1 takes as input a set of atomic formulas Pred and actions $\mathcal{A}$, both ungrounded, and a mapper $\mathcal{M}$ which associates substitutions from variables to constants. Firstly, it synthesises all unground norms combining the possible activation contexts (with lengths from 0 to $|\text{Pred}|$) with the actions. For this, the algorithm calculates the power set of Pred multiplying it by $\mathcal{A}$ (line 2). To each pair (φ, a) of set $\mathcal{X}$, a norm is obtained and saved in the set $\mathcal{N}$ (line 4). Secondly, the algorithm obtains partial and grounded versions of norms of $\mathcal{N}$ through the method getOtherNorms (line 6). For each norm $n \in \mathcal{N}$, such method takes its variables p to p, with $0 \leq p \leq j$ (with j the number of terms of n obtained by the method *getNumberOfTerms*) to be substituted by all constants of same type (according to map $\mathcal{M}$). Each possible substitution generates a new norm which is assigned with a given level of generality (according to the definition 12) and is included in the set $\mathcal{N}'$ (consider the example 7, the norms n_1, n_6 and n_7 were obtained in the first step while n_2, n_3 and n_4 in the second one).

The algorithm 2 takes as input the previous set $\mathcal{N}'$ and map $\mathcal{M}$, the sets of ungrounded atomic formulas Pred$'$ and Pred$''$ (which are ungrounded versions of s_0 and $\mathcal{G}$ from definition 14, respectively) and a MAS. The algorithm works in three steps. Firstly, it obtains all possible initial states and system goals, S_0 and S_G, respectively, through the methods *getSMAInitialStates* (line 1) and *getSMAGoals* (line 2). Such methods instantiate the formulas of Pred$'$ and Pred$''$ by substituting their variables by constants (according to $\mathcal{M}$). In the second step, the MAS is executed multiple times through method *run* (line 6) varying its current *configuration* - a quadruple formed by a norm, an initial state, a goal system and a conflict. To certify the MAS' goal will not be satisfied before its launching, the MAS runs only if the intersection between its initial state and goal is empty (line 5).

As the configurations change, the agents will execute different sequences of actions to satisfy the system goal and, in this dynamic, different situations of conflicts may arise.

Algorithm 2 Norms Evaluation.

```
Require: $\mathcal{N}'$, $\mathcal{M}$, Pred′,Pred″ $\subset \mathcal{L}$, $c \in \mathcal{C}$,
         MAS
1 $S_0 \leftarrow$ getSMAInitialStates(Pred′, Map)
2 $S_\mathcal{G} \leftarrow$ getSMAGoals(Pred″, Map)
3 foreach norm $n \in \mathcal{N}'$ do
4     foreach state $s_0 \in S_0$ and $\mathcal{G} \in S_\mathcal{G}$ do
5         if $s_0 \cap \mathcal{G} = \emptyset$ then
6             sat,tim,conf $\leftarrow$ MAS.run($s_0$, $\mathcal{G}$,
                c, n)
7             storeResults(sat,tim,conf)
8 getLevelsPerformance()
```

Given a norm *n*, its performance in the system may: (i) avoid the conflict and make the system goals feasible to be reached; (ii) avoid the conflict to the detriment of system goals reachability (the system timeout since the effort to the goals to be satisfied becomes even greater or there are no alternative actions to be executed); (iii) not avoid the conflict which cause the system interruption (since we consider that conflicts are impediments to the system goals be satisfied). The method *run* returns a vector of length 3, indicating whether: the goal were satisfied (*sat* - according to case i), the execution finished by timeout (*tim* - according to case ii), or there was a conflict (*conf* - according to case iii). The results obtained at each execution are saved by method *storeResults* (line 7). At the end of the algorithm, the results are summarized by method *getLevelsPerformance* (line 12) which calculates, to each level of generality, the percentage of tests which ended up in goal satisfied, conflict occurrence and timeout. From the summarized results, we intend to identify how the norms' levels of generality influence on the system goals reachability.

4. Tests and Results

We have developed a simulator of MAS in Java, which implements the algorithms 1 and 2, to empirically evaluate the performance of norms in avoiding conflicts and keeping the system goals reachable. We perform experiments in two distinct scenarios: the traffic scenario domain, presented in section 2 and a scenario where agents (called as employees) try to access a set of files available by a given organization. In this scenario, agents may have individual goals which may be incoherent with the system goals (as open certain documents not specified by the system), where conflicts may arise since obtain certain information may compromise the organization's data and privacy. Thus, the norms must be able to regulate the agents in order to keep the data safe and the system goals reachable. We implemented a systematic sampling in methods *getInitialStates* and *getSMAGoals* of algorithm 2 with confidence level of 95% with a error margin of 5% for both scenarios. Moreover, in the graphics of the tests, the *x*-axis represents the levels of generality of norms while the *y*-axis represents rates of satisfiability, conflicts and timeout. Each scenario was tested ten times on a computer using a Intel Core i5 5200U processor running at 2.2GHz, with Ubuntu 16.10 64-bit as operating system and 8GB of memory.

4.1. Grid Scenario Evaluation

We tested the system varying the number of agents on the grid from 2 to 9. They used the *Manhattan distance* to estimate the best plan to reach the system goals. Let |agents| and |Vertices| be, respectively, the number of agents and vertices on the grid, then each agent had $2^{|\text{Vertices}|}$ steps to conclude its plan and the system timeout in $|\text{agents}| * 2^{|\text{locals}|} + 1$ steps.

Figure 2 illustrates the curves of satisfiability rates. Generally speaking, the curves grow smoothly up to the level 0.8, reaching their highest value at the level 0.833 and, then, decreasing abruptly until the level 2. This happens because from the level 0 to 0.8 the norms are very specific which makes them inefficient in avoiding collisions as they hardly ever become active. However, as they become more generic (the number of atomic formulas decreases and the number of variables increases) the satisfiability rate grows somewhat since more agents' behaviors start to be regulated. Furthermore, as the number of agents increases, the rate of satisfiability decreases, since as more agents are on the grid, more collisions are likely to occur (for 2 agents, for example, the rate is initially about 66% while for 3 agents it is about 27%). At the level 0.833 there is uniquely the norm $n_1 = \{\mathtt{at(Ag_2,L_2)}\} \underset{p}{\rightarrow} \mathtt{move(Ag_1,L_1,L_2)}$ (from example 5 of section 2), thus no collision occurs when this norm is active. For 2, 3 and 4 agents on the grid, the satisfiability rate is *high* because it is easier to obtain alternative paths to achieve the system goal. However, for 5 or more agents, such rate is *low* since situations in which the agents are surrounded by other agents and can not move became common.

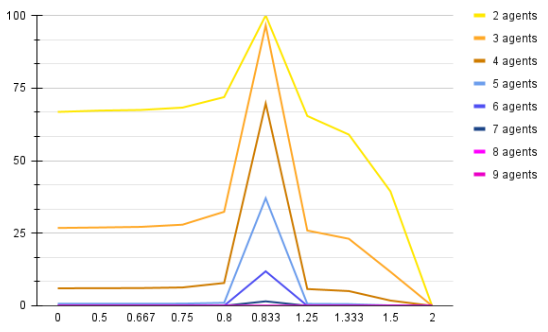

Figure 2. The rate of satisfiability for all executions.

After the level 0.833, the norms are of the form $\{\ \} \underset{p}{\rightarrow} a$. This type of norm are very generic and severely restrains the agents' autonomy since it applies to every system state. Such norms reduce the number of alternative actions to be taken in place of those that have been prohibited. Thus, achieving the system goal become a more complex or unfeasible task and, consequently, the satisfiability rate started to decrease. At level 2, there is uniquely the norm $n_7 = \{\ \} \underset{p}{\rightarrow} \mathtt{move(Ag_1,L_1,L_2)}$ (from example 5 of section 2) which prevents the agents from moving, then no goal can be satisfied and the satisfiability rate is 0. Figure 4 illustrates the curves of conflict rates. As the number of agents increases the rate of conflicts increases as well. For all curves, such rate started in a given positive value and decreased smoothly until the level 0.8 becoming 0 at level 0.833 (due to the norm n_1 as explained previously). After that level, the rate of conflicts started to grow since the norms become more generic and strictly, thus the number of paths to be chosen became reduced and the agents were more susceptible to cross and collide. At level 2, the rate of conflicts became 0 since no agent can move (due to the norm n_7).

Lastly, figure 3 illustrates the curves of timeout rates. For all curves, as the level of generality gets bigger, the rate of timeout, which started as 0, keeps unchangeable until level 0.8. Such results correspond to executions with mildly restrictive norms. Since the agents had enough time to accomplish the system goals none of them got stuck in a given vertex and, consequently, there was no timeout for the interval [0,0.8] of levels of generality. Thus, conflicts occurred for some configurations and for others the system

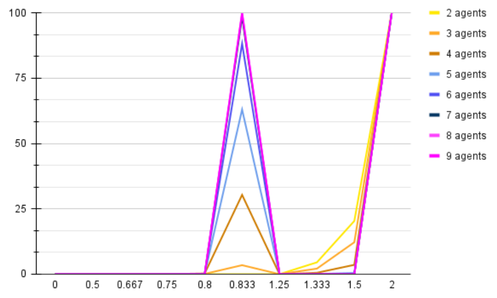

Figure 3. The rate of timeouts for all executions.

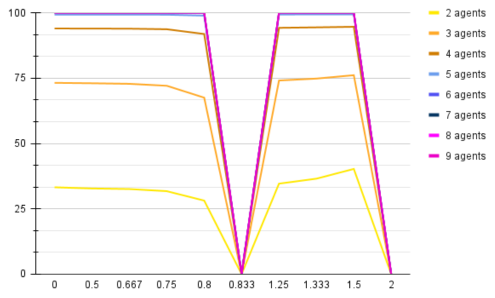

Figure 4. The rate of conflicts for all executions.

goals were satisfied. At level 0.833 the rate of timeout gets bigger as the number of agents increases on the grid. Since such results refers to executions with the norm n_1, the more agents are on the grid, the more vertices are occupied. Then, the agents got stuck in their depart vertices whereas was not possible to move to an adjacent vertex which was occupied. For all curves, the rate of timeout reached the value of 100% at level 2. This occurred because of norm n_7 which prevented the agents from moving to any vertex.

With these results we may think there exists a level of generality where norms tend to be capable of avoiding conflicts and keeping the goals reachable as much as possible. Let us call that level as *apex level*. According to the grid scenario, the *apex level* corresponds to the level 0.833 (around the center of the domain). In order to look further about the apex level, in the next section we provide some tests in another domain.

4.2. Security Domain Evaluation

In this scenario, an organization shares with its employees a set of documents. Such documents are classified as public or confidential while the employees, in turn, as low, medium or high according to the roles which they play in the company. The documents are available in a repository of a local server which is accessible from any computer of the organization's internal network. Then, to open a given document an employee must provide his credentials to access the repository. However, due to data protection and privacy issues the confidential documents must be open only by high employees. Thus, in this context, when a low or medium employee opens a document of that type a conflict situation occurs. Although opening a confidential document by those employees does not make part of the system goal, it can make part of the individual goal of some of them.

We have tested this scenario with 10 agents and 30 documents. Basically, the agents' behaviour consisted of entering the repository, trying to open the documents allocated to it by the system goal or those ones related to its individual goals and, finally, leaving the repository. Each initial state consisted in varying the classifications of employees and documents. Moreover, in all initial states the employees were logged out to the repository. On the other hand, each system goal consisted in varying the documents the employees should open. However, 30% of employees (low and medium) had individual goals which were not coherent with the system goal, that is, they intended to open confidential documents. Moreover, the agents decided with a 50% of chance whether they should accomplish the system goal or their individual ones.

Figure 5 illustrates the curves of satisfiability, conflict and timeout rates. The norm $n_8 = \{\texttt{roleLevel}(\texttt{Ag}_1,\texttt{low}), \texttt{info}(\texttt{Doc},\texttt{confidential})\} \underset{p}{\rightarrow} \texttt{open}(\texttt{Ag}_1,\texttt{Doc})$, which is at level 0.428, is the efficient norm for this scenario being capable of avoiding conflicts and making the system goal reachable. Such level corresponds to the apex level for this scenario and, differently from the grid scenario where there was only the norm n_1 at the corresponding apex level, it is composed by many norms. The most norms of the level 0.428 are inefficient in avoiding conflicts, then its rate of satisfiability is affected by the low system performance when they rule the agents' behaviours. This way, the shape of the current apex level is less evident than that of the grid domain. Nevertheless, the biggest rate of satisfiability is still found in the apex level (around 40%).

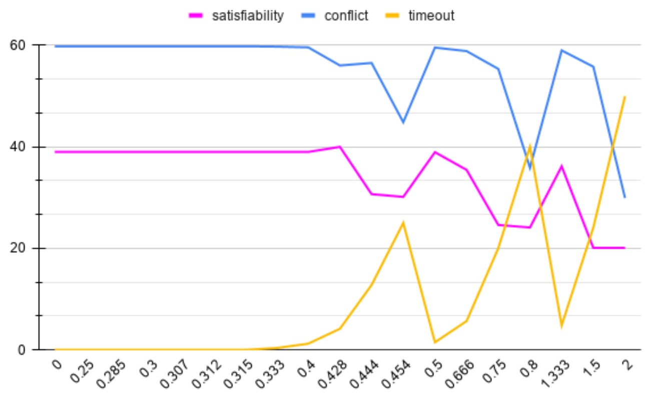

Figure 5. The results for Security Scenario.

As in the grid domain, before the apex level the norms are very specific and they hardly become active. Consequently, the agents have great autonomy to open the documents they wish which makes the rate of conflict for the interval [0,0.4] be high (around 60%) and the timeout be 0 (unless for levels 0.333 and 0.4). After the apex level, the norms start to become more strict having less atomic formulas. Whether a given norm prohibits an agent to open his target document and no alternative action in this scenario exist, the rate of timeout start to increase while the others rates decrease. However, the timeout declines from the level 0.454 to 0.5 and from 0.8 to 1.333 (which makes the others rates increase again). This happens because the levels 0.5 and 1.333 have one less formula, in relation to their previous levels, with all terms being constants. This way, the norms in levels 0.5 and 1.333 are not more genetic than those in the levels 0.454 and 0.8, respectively, thereby more agents became capable again to open their target documents which increases the conflicts and satisfiability rates. At last, the rate of timeout ends up 50% while the satisfiability and conflict ones ends up, respectively, 20% and 30%. Even though the current apex level has a satisfiability rate more discrete than that of the grid scenario, such apex level was found exactly in the center of the domain.

4.3. Discussion

According to the results, the apex level was found in the center of the domain or next to it. Before the apex level, the most norms showed to be mildly restrictive giving to the agents much freedom to act as they wished. Consequently, many conflicts occurred during the system execution. After the apex level, the most norms showed to be very strict giving to the agent low freedom to perform their actions. Consequently, the system did not satisfied its goal and timed out in many tests. On the basis of these data, we may say that norms which do not belong to the apex level impact negatively on the reachability of the system goal. On the other hand, the apex level showed to be an intermediate level of generality which have, at least, one norm capable to maximize the system goal reachability and minimize the occurrence of conflicts balancing control and autonomy as much as possible.

The appearance of the apex level around the center of the domain is resulting from the way which our model organizes the norms (at least to problems which admit one norm as a solution to establish the social order). We suppose we can use this fact as a heuristic to guide a search to find an efficient norm for a given domain just synthesizing the norms from a selected set of levels of generality (those around the center of the domain). For domains where the apex level may not be found around their center, a mechanism to decide by which levels of generality the search must start still need be provided.

5. Related Works

A method to synthesize norms which gives the agents the maximum autonomy to accomplish their goals is proposed in [Fitoussi and Tennenholtz 1998]. The authors employed the concept of *minimality*. A norm n_1 is said to be minimal if it is useful and there is no other norm n_2 which is more specific than n_1. Although a minimal norm gives the agents great flexibly to perform their actions as they wish, a minimal norm is, in the context of our work, the most mildly restrictive norm. Depending on the application domain, the chances of conflicts occur may be high and, according to our tests, synthesizing norms with an intermediate level of generality can make the system performs better in terms of avoiding conflicts and satisfying goals.

In [Morales et al. 2013], a normative network in which norms are organized hierarchically via generalization relationships is proposed. During the system execution, as conflicts occur, norms are synthesized to avoid them in the future. Those norms are inserted in the network which has its structure updated in a way that just when all children of a potential generalization exist, a new norm is created to generalize them (the children become inactive).The network is built and restructured until no conflicts occur. However, until the system get in a stable situation, this is accomplished trough different conflicts occurrence. In many domains, such incidents are not admissible implying in high costs to implement recovery actions and making the system goals unfeasible to be reached.

Posteriorly, in [Morales et al. 2014] the same authors proposed a new method to obtain a normative network where the generalization relationships are inferred through an ontology which structures the domain knowledge as a tree representing a taxonomy of terms. In the same way as before, as conflicts occur, norms are synthesized to avoid them. Whether two or more norms are generalizable, a new norm is inferred to generalize them taking into account the terms in the ontology and selecting alternative more general terms. Such proposal also has the limitation to allow conflicts for a while. However, this new approach does not need that all children to its parent be inferred. Thus, the occurrence of conflicts may decrease along the system execution in relation to their former work.

6. Conclusion

In this paper we have presented a model for exploring the generality of norms. Our aim was to find out a level of generality where norms are capable to avoid conflict and keep the system goals reachable. We found, for two distinct domains, that there exists a level of generality, which we called as *apex level*, where exists a norm which tend to be efficient in avoiding conflicts and ensuring goals reachability as much as possible. As future works we plan to evaluate empirically other scenarios where more than one type of conflict may occur and a set of distinct norms must be necessary to efficiently regulate the agents in order to further characterise the existence of the *apex level*.

References

Boella, G. and van der Torre, L. (2004). Regulative and constitutive norms in normative multiagent systems. In *Proc. of the Ninth Int. Conf. on Principles of Knowledge Representation and Reasoning*, KR'04, pages 255–265. AAAI Press.

Boella, G. and van der Torre, L. (2007). Norm negotiation in multiagent systems. *International Journal of Cooperative Information Systems (IJCIS)*, 16(2).

Christelis, G. and Rovatsos, M. (2009). Automated norm synthesis in an agent-based planning environment. In *Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems*, volume 1 of *AAMAS '09*, pages 161–168, Richland, SC. International Foundation for Autonomous Agents and Multiagent Systems.

Fitoussi, D. and Tennenholtz, M. (1998). Minimal social laws. In *Proc. of the Fifteenth National/Tenth Conf. on Artif. Intell./Innovative App. of Artif. Intell.*, page 26–31.

Frantz, C. K. and Pigozz, G. (2018). Modeling norm dynamics in multi-agent systems.

Morales, J., López-Sánchez, M., and Esteva, M. (2011). Using experience to generate new regulations. In *Proc. of IJCAI 2001*, page 307–312.

Morales, J., López-Sánchez, M., Rodríguez-Aguilar, J., Wooldridge, M., and Vasconcelos, W. (2014). Minimality and simplicity in the on-line automated synthesis of normative systems. In *AAMAS*.

Morales, J., Lopez-Sanchez, M., Rodriguez-Aguilar, J. A., Wooldridge, M., and Vasconcelos, W. (2013). Automated synthesis of normative systems. In *Proc. of AAMAS 2013*, pages 483–490.

Onn, S. and Tennenholtz, M. (1997). Determination of social laws for multi-agent mobilization. *Artificial Intelligence*, 95(1):155 – 167.

Savarimuthu, B. T. R., Cranefield, S., Purvis, M., and Purvis, M. (2008). Role model based mechanism for norm emergence in artificial agent societies. In *Proc. of COIN 2008*, pages 203–217.

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