INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DO TRIÂNGULO MINEIRO
PRÓ-REITORIA DE PESQUISA, PÓS-GRADUAÇÃO E INOVAÇÃO
htttp://editora.iftm.edu.br/index.php/inova

ISSN 2447-4924 (Impressa)
ISSN 2447-598X (Digital)

CIÊNCIA DA COMPUTAÇÃO

A PLATFORM FOR DETECTION AND RESOLUTION OF CONFLICTS AMONG MULTIPLE NORMS IN MULTI-AGENT SYSTEMS

Eduardo Augusto Silvestre[1], Viviane Torres da Silva[2], Oscar da Silva[3], Luccas Felippe Oliveira[4], Wendler Souza Ramos[5]

RESUMO: In open Multi-Agent Systems, norms are being used to regulate the behavior of the autonomous, heterogeneous and independently designed agents. Norms describe the behavior that can be performed, must be performed, and cannot be performed in the system. One of the main challenges on developing normative systems is that norms may conflict with each other. Norms are in conflict when the fulfillment of one norm violates the other and vice-versa. In previous works, the conflict checkers consider that conflicts can be detected by simply analyzing pairs of norms. However, there may be conflicts that can only be detected when we analyze several norms together. This work presents a complete approach for conflict detection and conflict resolution among pairs and multiple norms. The strategy for conflict detection is divided into three steps in order to smooth the computational cost since the problem is intrinsically exponential. The strategy for conflict resolution applies famous strategies found in literature to rewrite or remove conflicting norms. The paper describes a tool, named Multiple Norms Conflict Checker Tool (MuNoCC), that integrates different screens for conflict detection and resolution. This approach works as a basis to assist MAS literature, defining norms formally, checking conflicts among pairs and multiple norms and resolving the conflicts found.

Palavras-chave: software agent, multi-agent systems, normative system, conflict detection, conflict resolution.

1 Doutor em Computação. Instituto Federal do Triângulo Mineiro (IFTM), *Campus* Avançado Uberaba Parque Tecnológico, Uberaba, MG, Brasil. eduardosilvestre@iftm.edu.br
2 Doutora em Computação. IBM Research, Rio de Janeiro, RJ, Brasil. vivianet@br.ibm.com
3 Graduando em Análise e Desenvolvimento de Sistemas. Instituto Federal do Triângulo Mineiro (IFTM), *Campus* Avançado Uberaba Parque Tecnológico, Uberaba, MG, Brasil. oscardasilva52@gmail.com
4 Graduando em Engenharia da Computação. Instituto Federal do Triângulo Mineiro, *Campus* Avançado Uberaba Parque Tecnológico, Uberaba, MG, Brasil. lfelippeoliveira@gmail.com
5 Graduando em Engenharia da Computação. Instituto Federal do Triângulo Mineiro, *Campus* Avançado Uberaba Parque Tecnológico, Uberaba, MG, Brasil. wendler.ramos@etec.sp.gov.br

INTRODUCTION

Multi-Agent Systems (MAS) have been gaining great importance in the development of various applications. MAS are autonomous, and heterogeneous societies that can work to achieve common or different goals (WOOLDRIDGE, 2009). MAS has been applied in different areas, such as, power engineering applications (MCARTHUR et al., 2007), robotic (TANG et al., 2016) and health-care (IQBAL et al., 2016).

In order to deal with the autonomy and diversity of interests among different members, the behavior of agents is governed by a set of norms specified to regulate their actions (DA SILVA, 2008). The norms govern the behavior of agents by defining obligations (stating the actions that the agents must perform), prohibitions (stating the actions that the agents must not perform) or permissions (stating the actions that the agents can perform). In a MAS with many agents with different goals, an important issue to tackle is that norms can conflict with each other. A conflict occurs when norms regulating the same behavior have been activated and are inconsistent (VASCONCELOS; KOLLINGBAUM; NORMAN, 2009). In such cases, the agent is unable to fulfill all the activated norms. The detection and resolution of conflicts are two of the most challenges of the area.

Although there are several works that deal with normative conflicts such as (CHOLVY; CUPPENS, 1995; DA SILVA; ZAHN, 2013a; ELHAG; BREUKER; BROUWER, 1999; KOLLINGBAUM et al., 2007; VASCONCELOS; KOLLINGBAUM; NORMAN, 2009), to the best of our knowledge, all those approaches check for conflicts by analyzing the norms in pairs. However, there are conflicts that can only be detected when we consider several norms together.

For instance, let's consider a conflict that can only be detected if norms N1, N2 and N3 are analyzed together. N1 obliges agent A to dress a red shirt. N2 forbids agent A to dress red pants. N3 obliges agent A to dress pants and shirt of the same color. There are no conflicts between the pairs N1-N2, N2-N3 and N1-N3, but when the three norms are analyzed together, we can figure out the conflict.

In this paper, we present an approach to detect and resolve direct conflicts among multiple norms in MAS. The analysis of all possible combinations of norms is a NP-complete problem (MORALES et al., 2014; SHOHAM; TENNENHOLTZ, 1995). A strategy to address this problem was proposed. This paper doesn't deal with indirect conflicts.

The remainder of this paper is organized as follows. Section 2 presents the background material about the definition of norms. Section 3 describes the conflict detection mechanism. Section 4 describes the conflict resolution approach. Section 5 analyzes some of the main works related to this research. Section 6 presents details about the implementations performed in this research. Finally, Section 7 states some conclusions and future work.

NORMS

Norms have been vastly used in open MAS to cope with the heterogeneity, autonomy and diversity of interests among the different members. Norms describe the behavior that can be performed, that must be performed, and that cannot be performed.

Our norm definition is based on (DA SILVA FIGUEIREDO; DA SILVA; DE OLIVEIRA BRAGA, 2011), where the authors analyze the key strategies found in the literature for describing a norm. According to the authors, a norm prohibits, permits or obliges an entity to execute an action in a given context during a certain period of time. The difference between our norm definition and the one presented by them is the representation of the action being regulated. They represent the action by a single constant (e.g. pursue, reach). Our representation is more expressive. Furthermore, we consider only action for the behavior; we are not considering states. From now, it will only be used the term action.

For our definition of norm, consider the following definitions for sets: ***Nrm*** is the set of all norms, ***C*** is the set of all contexts, ***E*** is the set of all entities, ***A*** is the set of all actions, ***Cd*** is the set of all activations and deactivations condition, ***Org*** is the set of all organizations, ***Env*** is the set of all environments, ***Ag*** is the set of all agents and ***R*** is the set of all roles.

A norm $n \in Nrm$ is a tuple of the form:

$$(deoC, c, e, a, ac, dc)$$

where ***deoC*** is a deontic concept from the set {***O, F, P***}, respectively, obliged, forbidden and permitted; $c \in C$ is the context where the norm is defined; $e \in E$ is the entity whose action is being regulated; $a \in A$ is the action being regulated; $ac \in Cd$ indicates the condition that activates the norm and $dc \in Cd$ is the condition that deactivates the norm.

Every norm is defined in the scope of a context. The entity, whose action is being regulated, must fulfill the norm when executing in the context where the norm is being defined. In this paper, we consider that a norm can be defined in the context of an organization $o \in Org$ or of an environment $env \in Env$. The set of possible contexts are defined as $C = Org \cup Env$. A norm regulates the action of an agent $a \in Ag$, an organization (or group of agents) $o \in Org$ or a role $r \in R$. Agents, organizations and roles are entities of the set $E = Ag \cup R \cup Org$.

The activation and deactivation conditions, $ac \in Cd$ and $dc \in Cd$, can state an event that can be a date, the execution of an action, the fulfillment of a norm, etc. In this paper, we will focus on the specification of dates since it is easier to figure out which event has occurred first. Thus, we use simple mathematic symbols such as ≤ and ≥ to indicate that an event occurs before or after another ($\forall n \in N$, $ac \leq dc$).

An action is defined by the name of the action and, optionally, an object where the action will be executed and

a list of attributes (with their values). Thus, in this paper we define four different ways to represent the action:

(i) action;
(ii) action object;
(iii) action (attribute1 = {value1}, attribute2 = {value2}, ...});
(iv) action object (attribute1 = {value1}, attribute2 = {value2}, ...}).

The designer of a MAS can set any of the types of norms to represent his/her domain. These different ways of defining a norm represent a great flexibility in creating a MAS. In order to exemplify these four ways to describe an action, let's consider the following four prohibition norms:

(i) Na forbids agent A to get dressed;
(ii) Nb forbids agent A to dress pants;
(iii) Nc forbids agent A to dress red;
(iv) Nd forbids agent A to dress red pants.

The actions described in the norms are represented as:

(i) Na: dress;
(ii) Nb: dress pants;
(iii) Nc: dress (color={red});
(iv) Nd: dress pants (color={red}).

With the purpose of formally describe norms, the authors (OLIVEIRA; SILVESTRE; SILVA, 2017) developed a grammar in the language Backus–Naur form (BNF) (MCCRACKEN; REILLY, 2003).

CONFLICT DETECTION

The main goal of this section is to present the approach that is able to detect direct conflicts among multiple norms. The conflict checker algorithm is divided in three steps. The first step is responsible for filtering the norms by including them into sets of similar norms. In order to do so, such step uses three filters. The first filter separates into sets the norms that apply in the same context, the second filter separates into subsets the norms that govern the same entity and the third filter separates into subsets the ones that regulate the same action.

After applying all filters, only the norms stored in the same set are the ones that may be in conflict. Norms stored in different sets apply in different contexts, govern different entities and regulate different actions, thus, they do not conflict.

The second step of the algorithm is done to solve the problem of analyzing several norms with different deontic concepts at the same time. Our strategy to overcome such problem is to use a single deontic concept to analyze the norms. A copy of the original norms (that are not permissions) with all its deontic concepts transformed to permissions is created. Note that we do not change the original norms, but the copies of such norms.

Several approaches studied the deontic transformations. Some approaches use the O operator as primitive (MCNAMARA, 2014) while others use the *P* operator (VON WRIGHT, 1951). In this work, we use the *P* operator as primitive and apply the following abbreviations to transform an obligation to a permission (case (1)) and a prohibition to a permission (case (2)):

1. Op ↔ ***¬P¬p*** ***2. Fp*** ↔ ***¬Pp***

a) From obligation to permission

Von Wright (1951) proposed the weak axiom ***Op→Pp*** that indicates that when p is obliged, p is permitted. Following such axiom and assuming that the designer wants to enable agent A to execute p, we consider that if there is a norm obliging an agent A to execute an action p, such norm can be used as a permission to execute action p. Thus, in this step of the algorithm, all obligations are used as permissions.

b) From prohibition to permission

In this paper we are using the Closure Principle which says that what is not explicitly forbidden is permitted (CZELAKOWSKI, 2015; TRYPUZ, 2013). Therefore, if there is not a prohibition addressed to an agent to execute an action over an object, the agent is permitted to execute such action over the object.

Following this principal, we consider that if there is a norm prohibiting an agent *A* to execute an action *p*, such norm states that *A* is not permitted to execute *p*. We assume that it is not necessary to create permissions related to everything that is not said in the prohibition. Thus, in this step of the algorithm, all prohibitions are used as negations of permissions.

The checking for conflicts is executed in the third step of the algorithm. The algorithm checks if the norms in each set are in conflict. Since all norms are permissions, the analysis made by the conflict checker is very simple; it checks if the norms "intersect". Two or more norms intersect if there is at least one possible situation where all the permissions are activated and can be fulfilled. In such case, the norms are not in conflict because there is a situation where the agent is able to fulfill all the original norms.

The conflict checker starts by checking the norms in a set by pairs of norms and then consider all possible set of k-norms until k be equal to the number of norms in the set. At the end, the algorithm has checked for conflicts among all the norms of the set at the same time.

In order to exemplify our approach, let's consider the three norms described in Section 1. We have augmented these norms by including the context where the norms are executed and the periods during while the norms are activated.

N1: Obliges agent A in orgO to dress a red shirt in 03/01/2015.

N2: Forbids agent A in orgO to dress red pants from 01/01/2015 until 12/31/2015.

N3: Obliges agent A in orgO to dress pants and shirt of the same color after 02/01/2015.

Norm N3 applies to two objects. It is natural to imagine (without loss of information) that this norm can be divided into two norms: one on pants and one on shirt (strategy to facilitate visualization and checking for conflicts). As the color of the pants and the shirt are indifferent, but must be the same color, a variable is used (for convenience, X) and this variable must have the same value for pants and shirt. We get:

N1:(O, orgO, agentA, dress shirt (color = {red}), 03/01/2015, 03/01/2015)

N2:(F, orgO, agentA, dress pants (color = {red}), 01/01/2015, 12/31/2015)

N3a:(O, orgO, agentA, dress pants (color = {X}), 02/01/2015, _)

N3b:(O, orgO, agent A, dress shirt (color = {X}), 02/01/2015, _)

In the first step, the algorithm groups all norms in the same set since they are applied in the same context (*orgO*), govern action of the same entity (*agentA*) and regulate the same action (*to dress*).

In the second step, the algorithm transforms the norms to permissions. Remembering that this transformation is not made in the original norms. Let's take a look to norm N2 that prohibits agent A to dress red pants. N2 does not say anything about agentA to dress shirts (of any color), to dress white or black pants or to execute the action of writing papers. In short, the prohibition is just about to dress red pants.

In order to transform a prohibition into a permission, we assume that it is not necessary to create permissions related to everything that is not said in the prohibition (since it is already done by applying the Closure Principle). In addition, any permission that talks about actions and objects that are not the ones refereed in the prohibition are not relevant to the checking of conflicts between the prohibition and any other norm. Therefore, a prohibition like N2 is transformed to a permission that only talks about the agent, action, object and attributes described in the norm. N2 is transformed to a norm that permits agent A to dress pants NOT red.

Returning to our example, the transformation of norms N1, N3a and N3b into permissions is very simple since they are obligations, as follows:

N1: (P, orgO, agentA, dress shirt (color = {red}), 03/01/2015, 03/01/2015)

N3a: (P, orgO, agentA, dress pants (color = {X}), 02/01/2015, _)

N3b: (P, orgO, agent A, dress shirt (color = {X}), 02/01/2015, _)

The transformation of N2, which is a prohibition, to a permission is done by negating the color of the pants, that is, the color of the pants is changed to its complement.

N2: (P, orgO, agentA, dress pants (color = {NOT red}), 01/01/2015, 12/31/2015)

In the third step, the checking for conflicts is executed. In our example of N1, N2 and N3, it is easy to see that any group of two norms is not in conflict. Therefore, it is possible to find out situations where all norms can be fulfilled. The conflict only takes place when we consider the three norms together. In 03/01/2015, when the three norms are activated, agentA executing in orgO is permitted to dress a red shirt, a not red pants and a pants and a shirt of the same color.

CONFLICT RESOLUTION

This section presents the strategies that have been developed to resolve the conflicts detected by the conflict checker. Section 4.1 presents the main strategies found in the literature used for conflict resolution among pairs of norms. Section 4.2 describes the strategies that have been created that can be used to resolve conflicts among multiple norms. Section 4.3 details the implementation of conflict resolution.

Famous Strategies for Conflict Resolution

Another important point in the study of normative conflicts is the resolution of the conflicts found. After the detection of the conflicts by the conflict checker a strategy of conflict resolution is necessary. To resolve the conflict two basic operations can be performed:

- Removal of one or more existing norms for conflict elimination
- Rewriting of one or more existing norms for conflict elimination

In summary, some of the main strategies found in the literature are described in subsections.

Removal of one or more existing norms for the elimination of conflicts.

Lex Posterior, Lex Superior and Lex Specialis (KOLLINGBAUM; NORMAN, 2004; LEITE; ALFERES; PEREIRA, 2001; LOPES CARDOSO; OLIVEIRA, 2008). In the case of Lex posterior, the newer norm takes precedence over older norms. In the case of the Lex Superior, the norm imposed by the greater power has priority. In the case of Lex Specialis, the more specific norm takes precedence.

Precedence of modality (KAGAL; FININ, 2007; MOFFETT; SLOMAN, 1994). If the positive modality is chosen then, in cases of conflicts, permission and obligation will overwrite the prohibition. If negative modality is chosen then, in cases of conflict, the prohibition will overwrite the permission and the obligation.

Priority among the norms (GAERTNER et al., 2007; GOVERNATORI; ROTOLO, 2011). At design time, a priority

is chosen for the norms arbitrarily. Norms with the lowest priority are excluded to resolve the conflict. In (GARCÍA-CAMINO; NORIEGA; RODRÍGUEZ-AGUILAR, 2007) authors define priority as the salience of the norm.

Rewritten one or more existing norms for the elimination of conflict.

Direct Conflict with Restrictions (VASCONCELOS; KOLLINGBAUM; NORMAN, 2009). In this case, besides the normal definition of the norm, the representation of authors also has restrictions. The conflict is solved by manipulating these constraints; by manipulating the constraints, the conflicting intersections are eliminated.

Indirect Conflicts (VASCONCELOS; KOLLINGBAUM; NORMAN, 2009). The strategy for resolving indirect conflicts is the same applied to direct conflicts described previously.

Strategies for Resolving Conflicts Developed

As the conflict among multiple norms was first addressed in the literature recently (SILVESTRE, 2017; SILVESTRE; DA SILVA, 2015, 2016a, 2016b; SILVESTRE; SILVA, 2018), no approach was also found for conflict resolution among multiple norms. Using the strategies presented previously as inspiration, the authors developed and implemented some conflict checking strategies among multiple norms. Strategies can be based on the removal of conflicting norms (Strategy 1, Strategy 2 and Strategy 3) or strategies can be based on changing conflicting norms (Strategy 4, Strategy 5 and Strategy 6).

Strategy 1. This strategy of conflict resolution is divided into two steps: removal of all norms with variables at a single time and application of the policy of conflict resolution among pairs of norms.

The first step is to remove all norms that have variables at a single time. Because conflicts among multiple norms only occurs if there is a variable, if these norms are excluded, there will automatically be no conflict among multiple norms. In this strategy, all norms are deleted at one time.

The second step of the strategy is to resolve existing conflicts among pairs of norms. The strategy is applied by changing the activation period of the norms, so that the norms do not have more conflict. The norm that participates most in conflicts is chosen, its activation period is modified so that it does not intercept with any other norm and is checked again if a new conflict still exists. This step is done repeatedly until there are no more conflicts among the input norms. The second step is the same for all the six strategies, so it will only be described here.

Strategy 2. This strategy of conflict resolution is divided into two steps: removal of all norms with variables (one norm at a time) and application of the policy of conflict resolution among pairs of norms.

The first step is to remove all norms that have variables until conflicts among multiple norms are no longer found. Unlike Strategy 1, norms with variables are deleted in a process repeatedly, one at a time. A norm is chosen with the variable that participates in most conflicts among multiple norms and this norm is removed. The conflict checker runs again until there is no more conflict among multiple norms. In this approach, it is not mandatory that all norms with variables be removed.

Strategy 3. This strategy of conflict resolution is divided into two steps: removal of all norms with greater participation in conflicts and application of the policy of conflict resolution among pairs of norms.

The first step is to remove norms that participate most in conflicts until there is no conflict among multiple norms. The norm that participates in the largest number of conflict combinations is chosen and is removed from the set. This process is repeated until there is no more conflict among multiple norms. The norm chosen may have a variable or not.

Strategy 4. This strategy of conflict resolution is divided into two steps: change of all norms with variables (one norm at a time) and application of the policy of conflict resolution among pairs of norms.

The first step is to change the activation period for all norms that have variables at one time. Because the conflict among multiple norms only occurs if there is a variable, if these norms are changed, and no longer intercept, there will be no conflict automatically. In this strategy, all norms are changed at once.

Strategy 5. This strategy of conflict resolution is divided into two steps: change of all norms with variables (one norm at a time) and application of the policy of conflict resolution among pairs of norms.

The first step is to change the activation period of norms that have variables until no conflicts between multiple norms are found anymore. Unlike Strategy 4, norms with variables are changed in a process repeatedly, one at a time. It is chosen a norm with variable that participates of more conflicts and this norm has its period of activation changed. The conflict checker runs again until there is no more conflict among multiple norms. In this approach, it is not mandatory that all norms with variables be changed.

Strategy 6. This strategy of conflict resolution is divided into two steps: change of all norms with greater participation in conflicts and application of the policy of conflict resolution among pairs of norms.

The first step is to change the activation period of the norms that participate most in the conflicts until there is no conflict between multiple norms. The norm that participates in the largest number of conflict combinations is chosen and has its activation period changed. This process is repeated until there is no more conflict between multiple norms. The norm chosen may have a variable or not.

Implementation of Conflict Resolution.

In order to implement previously defined conflict resolution strategies, an input norms scenario was used. In this scenario the authors have manually defined 40 norms in a given domain (9 norms of TYPE (i), 9 norms of TYPE (ii), 9 norms of TYPE (iii) and 13 norms of type (iv)). The conflict checker detected 625 conflicts (6 conflicts among norms of TYPE (i), 6 conflicts among norms of TYPE (ii), 10 conflicts among norms of TYPE (iii), 510 conflicts among norms of TYPE (iv), 12 conflicts between norms of TYPE (i) and TYPE (ii), 12 conflicts between norms of TYPE (i) and TYPE (iii), 17 conflicts between norms of TYPE (i) and TYPE (iv), 0 conflict between norms of TYPE (ii) and TYPE (iii) (as expected), 17 conflicts between norms of TYPE (ii) and TYPE (iv) and 35 conflicts among norms of TYPE (iii) and TYPE (iv)). The program was able to detect these conflicts without identifying false positives and forgetting false negative conflicts.

The Table 1 presents a summary of the application of conflict resolution strategies implemented in Java. It is worth mentioning that the execution time of conflict resolution strategies was similar in all cases. In strategies that remove norms the total number of norms at the end is less than the number of norms at the beginning. Some strategies require a greater number of executions of conflict checker, in larger applications this requirement can be an impediment in terms of computational cost. The conflict resolution strategy to be used will depend on the need to maintain the number of original norms or not.

Table 1. Summary of implementation of conflict resolution strategies.

	Strategy 1	Strategy 2	Strategy 3	Strategy 4	Strategy 5	Strategy 6
Norms Initially	40	40	40	40	40	40
Norms TYPE(i)	9	9	9	9	9	9
Norms TYPE(ii)	9	9	9	9	9	9
Norms TYPE(iii)	9	9	9	9	9	9
Norms TYPE(iv)	13	13	13	13	13	13
Conflicts Initially Detected	625	625	625	625	625	625
Conflicts Detected at the end	0	0	0	0	0	0
Time for Resolution	5978ms	6124ms	6174ms	5279ms	6724ms	6250ms
Norms at the end	33	33	35	40	40	40
Executions of conflict checker	13	16	15	13	16	15

RELATED WORK

There are many approaches to detect and solve normative conflicts. These approaches were nicely summarized in (SANTOS et al., 2017).

Some approaches focus on the identification of direct and indirect conflicts (i.e., conflicts that occurs when the elements of norms being analyzed are not the same, but are somehow related), such as (BEIRLAEN; STRAßER; MEHEUS, 2013; CHOLVY; CUPPENS, 1995; DA SILVA; ZAHN, 2013b; ELHAG; BREUKER; BROUWER, 1999; GAERTNER et al., 2007; KAGAL; FININ, 2007; KOLLINGBAUM et al., 2007; OREN et al., 2008; VASCONCELOS; KOLLINGBAUM; NORMAN, 2009). All of these approaches analyze the norms in pairs when checking for conflicts.

Some approaches focus on the resolution of conflicts. Some were specified to be used at *design time* while others at *runtime*. In addition, some of them are able to solve only *direct* normative conflicts while others are also able to solve indirect conflicts.

The strategies used by the analyzed proposals can be divided into two kinds: norm prioritization (one norm overrides another in particular circumstances) and norm update (one of the norms in conflict is updated).

As an example of norm prioritization, (CHOLVY; CUPPENS, 1995) present two norms: (N1) a Christian ought not kill his neighbor; and (N2) if a soldier is ordered to kill an enemy, then he ought to kill him; and assume that there is an individual that is a Christian soldier who received the order to kill. Then, norms N1 and N2 are addressed to this individual and there is a normative conflict between N1 and N2. The strategy applied to resolve the conflict consists in deciding which norm is more relevant based on the role. In this example, the norms addressed to the role soldier are stated to be more relevant than norms addressed to the role Christian. Thus, norm N2 takes precedence on norm N1.

A situation of norm update is found in (VASCONCELOS; KOLLINGBAUM; NORMAN, 2009), where norms regulate parameterized actions. The authors present two conflicting norms: (N1) agent1 is forbidden to deploy (X), where $5 \leq X < 10$; and (N2) agent1 is obliged to deploy (X), where $8 < X < 10$. The mechanism used for resolving the conflict updates a norm by modifying the values of its constraints in order to reduce the scope of the norm and eliminate the conflict

The majority of proposals establish an order of prioritization between norms to specify which norm is more relevant. In this sense, there are three classic principles found in the literature (VASCONCELOS; KOLLINGBAUM; NORMAN, 2009) that have been used to solve deontic conflicts: lex posterior (it prioritizes the most recent norm), lex specialis (it prioritizes the most specific norm), and lex superior (it prioritizes the norm imposed by the most important issuing authority). Other approaches reduce the scope of influence of the conflicting norms in order to eliminate the overlap between them. They do so by manipulating the components of one of the norms.

IMPLEMENTATIONS
Overview

This section provides details about the implementations performed in this research. The code was developed using the Java language and its frameworks. Details are available at https://goo.gl/PqKHZ4. The implementations were divided into three main parts: (i) conflict verification and resolution, (ii) formal verification, and (iii) the integrator tool named MuNoCC (Multiple Norms Conflict Checker).

(i) Conflict verification and resolution: These projects contain all the code for verification and resolution of conflicts.

(ii) Formal verification: This project contains the code used to formally prove the algorithms used. Design-by-contract technique was used through JML (Java Modeling Language) (GARY T. LEAVENS ERIK POLL; DIETL, 2013).

(iii) MuNoCC integrator tool: The MuNoCC is a complete tool for checking and resolving conflicts. The main functionalities are: registering the components of the norms, creating norms and verifying conflicts (Figure 1); generate norms randomly and verify the program's ability to capture conflicts (Figure 2); import norms from a grammar (using a developed compiler) and verify conflicts (OLIVEIRA; SILVESTRE; SILVA, 2017); generate norms randomly and verify the program's ability to capture conflicts (Figure 3, Figure 4 and Figure 5); insert norms, detect and resolve conflicts and; import norms from an ontology and check their conflicts (Figure 6 and Figure 7).

The next two subsections present details of the parser developed and the verification of conflicts from an ontology.

Figure 1. Screen where you can manually create norms.

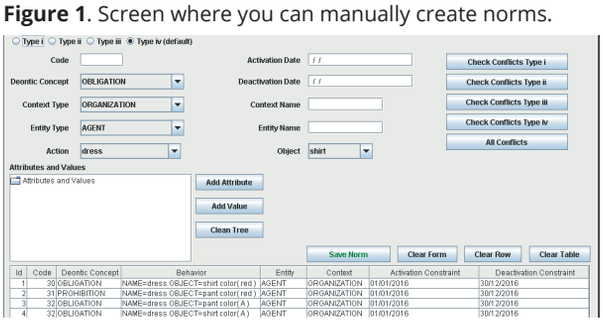
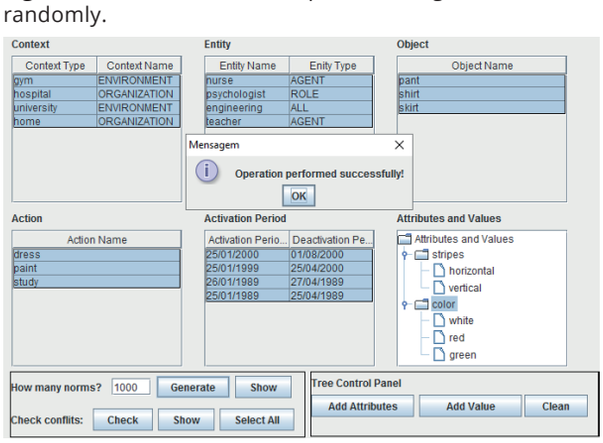


Figure 2. Screen where it is possible to generate norms randomly.



Conflicts from BNF

These subsections introduce the compiler developed. For further details check (OLIVEIRA; SILVESTRE; SILVA, 2017). Figure 3 shows the screen where the user can import norms and check their conflicts. It shows the norms described in Section 1 and the conflict that the algorithm found. The authors created a compiler to make the process of translating a norm in semi-structured language to Java objects.

The Lexical Analysis is the first phase of the compiler. It takes the user input written in the form of sentences and break the sentences into a series of tokens.

The Syntax Analysis is the second phase of the compiler. It takes the input from a lexical analyzer in the form of token streams. The parser analyzes the token stream against the production rules to detect any errors in the code. To verify the order of the tokens is correct a state machine has been implemented (Figure 4).

The state machine of Figure 4 shows the sequence of states that pass tokens for the sentence to be accepted by the compiler. Figure 5 illustrates the "Action" state, which is a composite of several states. Notice the state machine accepts sentences according to the norm defined in Section 1.

Figure 3. Import norms, parse them and detect conflicts

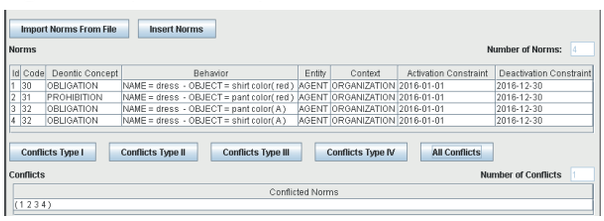


Figure 4. State machine of the norm.

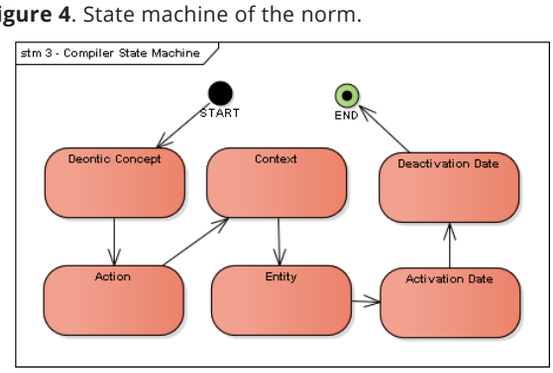
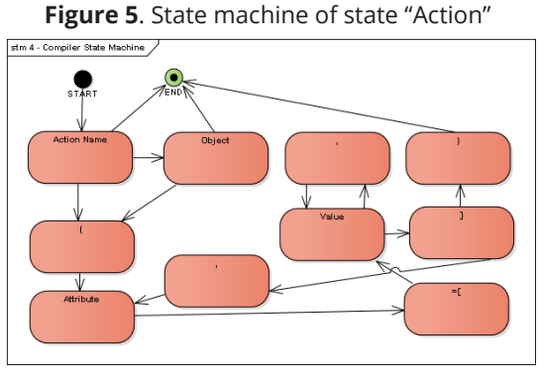


Figure 5. State machine of state "Action"



The Code Generation is the final phase of compilation. It takes the result of syntax analysis and produce java objects instances of class Norm.java. Each line as input produces one instance.

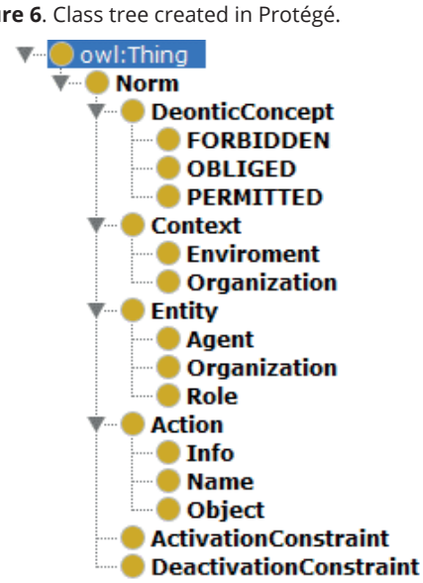
Conflicts from Ontology

Another strategy for conflict verification was the use of ontologies. According to (BORST; BORST, 1997), an ontology is defined as a formal and explicit specification of a shared conceptualization, where formal specification means something that is readable for computers, explicit refers to concepts, properties, relations, functions, constraints, and explicitly defined axioms, conceptualization represents an abstract model of some real and shared world phenomena means consensual knowledge.

It was identified that it is possible to represent, through an ontology, a normative system as a MAS, since the ontology offers several resources that can be used to represent the characteristics of a MAS and its norms and relationships. In addition, Protégè (MUSEN, 2015) was chosen as a computational tool to develop this project, since it has a simple and interactive visual interface, a high market acceptance and a vast amount of related contents in bibliographies.

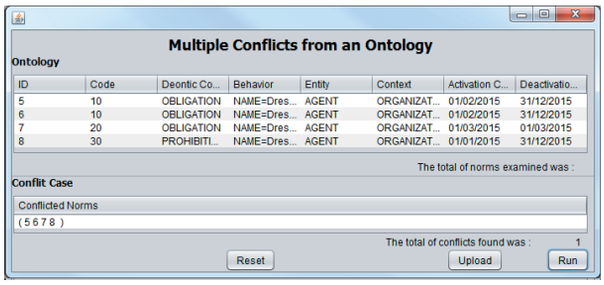
The definition of norm represented in Section 1 was represented by classes in Protégè (Figure 6). These classes are used to instantiate the norms of Section 1.

Figure 6. Class tree created in Protégé.



An ontology in Protégè is defined by an OWL (Ontology Web Language) file. This file is used in the tool as input for conflict checking. The software reads the OWL file and transforms the content into Java objects. To perform this process, the HermiT library (HERMIT, 2016), which verifies if an OWL file is consistent and its relationships, is used. From the Java objects it is possible to verify conflicts. Figure 7 shows the MuNoCC screen responsible for this task. The norms of Section 1 are used as input.

Figure 7. Conflict found from the norms defined in OWL.



The use of Protégè to define an ontology and the HermiT library to reason about it provides an alternative vision in the verification of normative conflicts, using techniques known in the literature in the area of normative MAS.

CONCLUSIONS AND FUTURE WORK

After an extensive literature search, several articles were found on the identification and resolution of normative conflicts. Such works focus on the identification and resolution of normative conflicts by considering pairs of norms. However, there are conflicts, as the ones exemplified in the text, which can only be detected and resolved when checking for conflicts among multiple norms.

The paper presents an approach able to check for conflicts among multiple norms that uses filters and transformations to reduce the computational cost of the algorithm.

The paper also discusses and implements several strategies for conflict resolution. Two groups of strategies were applied. Strategies 1, 2 and 3 remove norms and apply the policy of conflict resolution among pairs of norms. Strategies 4, 5 and 6 change norms and apply the policy of conflict resolution among pairs of norms. After applying the strategies, the initial set of norms will be free of conflicts. The strategies used appear as a first alternative for further research on conflict resolution among multiple norms.

Lastly, the paper describes a tool, named MuNoCC that integrate different screens for conflict detection and resolution.

In this work, we have not considered indirect conflicts (DA SILVA; ZAHN, 2013a). The algorithm presented in this paper does only check for direct conflicts, i.e., conflicts among norms that have the same entities, contexts and actions. An important and necessary extension of our work is the identification of indirect conflict among multiple norms.

REFERENCES

BEIRLAEN, M.; STRAßER, C.; MEHEUS, J. An Inconsistency-Adaptive Deontic Logic for Normative Conflicts. **Journal of Philosophical Logic**, v. 42, n. 2, p. 285–315, 2013.

BORST, W. N.; BORST, W. N. **Construction of Engineering Ontologies for Knowledge Sharing and Reuse**. Netherlands: University of Twente, 1997.

CHOLVY, L.; CUPPENS, F. **Solving Normative Conflicts by Merging Roles**. Proceedings of the 5th International Conference on Artificial Intelligence and Law. Anais...: ICAIL '95.New York, NY, USA: ACM, 1995

CZELAKOWSKI, J. **Freedom and Enforcement in Action: A Study in Formal Action Theory**. [s.l.] Springer Publishing Company, Incorporated, 2015.

DA SILVA FIGUEIREDO, K.; DA SILVA, V. T.; DE OLIVEIRA BRAGA, C. **Modeling Norms in Multi-agent Systems with NormML**. Proceedings of the 6th International Conference on Coordination, Organizations, Institutions, and Norms in Agent Systems. **Anais**...: COIN@AAMAS'10.Berlin, Heidelberg: Springer-Verlag, 2011

DA SILVA, V. T. From the specification to the implementation of norms: an automatic approach to generate rules from norms to govern the behavior of agents. **Autonomous Agents and Multi-Agent Systems**, v. 17, n. 1, p. 113–155, 2008.

DA SILVA, V. T.; ZAHN, J. **Normative Conflicts that Depend on the Domain**. Coordination, Organizations, Institutions, and Norms in Agent Systems {IX} - {COIN} 2013 International Workshops, COIN@AAMAS, St. Paul, MN, USA, May 6, 2013, COIN@PRIMA, Dunedin, New Zealand, December 3, 2013, Revised Selected Papers. **Anais**...2013a

DA SILVA, V. T.; ZAHN, J. **Normative Conflicts that Depend on the Domain**. Coordination, Organizations, Institutions, and Norms in Agent Systems {IX} - {COIN} 2013 International Workshops, COIN@AAMAS, St. Paul, MN, USA, May 6, 2013, COIN@PRIMA, Dunedin, New Zealand, December 3, 2013, Revised Selected Papers. **Anais**...2013bDisponível em: <http://dx.doi.org/10.1007/978-3-319-07314-9_17>

ELHAG, A. A. O.; BREUKER, J. A.; BROUWER, B. W. On the Formal Analysis of Normative Conflicts. (H. J. van den Herik et al., Ed.)JURIX 1999: The Twelfth Annual Conference. **Anais**...: Frontiers in Artificial Intelligence and Applications.Nijmegen: GNI, 1999

GAERTNER, D. et al. **Distributed Norm Management in Regulated Multiagent Systems**. Proceedings of the 6th International Joint Conference on Autonomous Agents and Multiagent Systems. Anais...: AAMAS '07.New York, NY, USA: ACM, 2007Disponível em: <http://doi.acm.org/10.1145/1329125.1329235>

GARCÍA-CAMINO, A.; NORIEGA, P.; RODRÍGUEZ-AGUILAR, J.-A. **An Algorithm for Conflict Resolution in Regulated Compound Activities**. Proceedings of the 7th International Conference on Engineering Societies in the Agents World VII. Anais...: ESAW'06.Berlin, Heidelberg: Springer-Verlag, 2007

GARY T. LEAVENS ERIK POLL, C. C. Y. C. C. R. D. C. P. M. J. K. P. C. D. M. Z.; DIETL, W. **JML Reference Manual**. [s.l: s.n.].

GOVERNATORI, G.; ROTOLO, A. Justice Delayed Is Justice Denied: Logics for a Temporal Account of Reparations and Legal Compliance. In: LEITE, J. et al. (Eds.). . **Computational Logic in Multi-Agent Systems**. Lecture Notes in Computer Science. [s.l.] Springer Berlin Heidelberg, 2011. v. 6814p. 364–382.

HERMIT, O. HermiT Reasoner. **URL: http://www. hermit-reasoner. com**, 2016.

IQBAL, S. et al. Application of Intelligent Agents in Healthcare: Review. **Artif. Intell. Rev**., v. 46, n. 1, p. 83–112, jun. 2016.

KAGAL, L.; FININ, T. Modeling conversation policies using permissions and obligations. **Autonomous Agents and Multi-Agent Systems**, v. 14, n. 2, p. 187–206, 2007.

KOLLINGBAUM, M. et al. Norm Conflicts and Inconsistencies in Virtual Organisations. In: NORIEGA, P. et al. (Eds.). . **Coordination, Organizations, Institutions, and Norms in Agent Systems II**. Lecture Notes in Computer Science. [s.l.] Springer Berlin Heidelberg, 2007. v. 4386p. 245–258.

KOLLINGBAUM, M.; NORMAN, T. Strategies for resolving norm conflict in practical reasoning. **ECAI Workshop Coordination in Emergent Agent Societies 2004**, 2004.

LEITE, J. A.; ALFERES, J. J.; PEREIRA, L. M. **Multi-dimensional Dynamic Knowledge Representation**. Proceedings of the 6th International Conference on Logic Programming and Nonmonotonic Reasoning. **Anais**...: LPNMR '01.London, UK, UK: Springer-Verlag, 2001

LOPES CARDOSO, H.; OLIVEIRA, E. **Norm Defeasibility in an Institutional Normative Framework**. Proceedings of the 2008 Conference on ECAI 2008: 18th European Conference on Artificial Intelligence. **Anais**...Amsterdam, The Netherlands, The Netherlands: IOS Press, 2008

MCARTHUR, S. D. J. et al. Multi-Agent Systems for Power Engineering Applications—Part I: Concepts, Approaches, and Technical Challenges. **IEEE Transactions on Power Systems**, v. 22, n. 4, p. 1743–1752, nov. 2007.

MCCRACKEN, D. D.; REILLY, E. D. Backus-naur form (bnf). 2003.

MCNAMARA, P. Deontic Logic. In: ZALTA, E. N. (Ed.). . **The Stanford Encyclopedia of Philosophy**. Winter 201 ed. [s.l: s.n.].

MOFFETT, J. D.; SLOMAN, M. S. Policy conflict analysis in distributed system management. **Journal of Organizational Computing and Electronic Commerce**, v. 4, n. 1, p. 1–22, 1994.

MORALES, J. et al. **Minimality and Simplicity in the On-line Automated Synthesis of Normative Systems**. : AAMAS '14. Paris, France: IFAAMAS, 2014

MUSEN, M. A. The ProtÉGÉ Project: A Look Back and a Look Forward. **AI Matters**, v. 1, n. 4, p. 4–12, jun. 2015.

OLIVEIRA, L. F.; SILVESTRE, E. A.; SILVA, V. T. DA. Um compilador para definição e geração de normas em sistemas multiagentes. **Revista Inova Ciência & Tecnologia**, 2017.

OREN, N. et al. An argumentation inspired heuristic for resolving normative conflict. 2008.

SANTOS, J. S. et al. Detection and resolution of normative conflicts in multi-agent systems: a literature survey. **Autonomous Agents and Multi-Agent Systems**, p. 1–47, 2017.

SHOHAM, Y.; TENNENHOLTZ, M. On social laws for artificial agent societies: off-line design. **Artificial Intelligence**, v. 73, n. 1–2, p. 231–252, 1995.

SILVESTRE, E. A. **Verificação de Conflitos entre Múltiplas Normas em Sistemas Multiagentes**. [s.l.] Universidade Federal Fluminense, 2017.

SILVESTRE, E. A.; DA SILVA, V. T. **Verifying Conflicts Between Multiple Norms in Multi-agent Systems**. Proceedings of the 2015 International Conference on Autonomous Agents

and Multiagent Systems, {AAMAS} 2015, Istanbul, Turkey, May 4-8, 2015. Anais...2015Disponível em: <http://dl.acm.org/citation.cfm?id=2773552>

SILVESTRE, E. A.; DA SILVA, V. T. **Verifying Conflicts Among Multiple Norms in Multi-agent Systems: (Extended Abstract)**. Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems. **Anais**...: AAMAS '16.Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems, 2016aDisponível em: <http://dl.acm.org/citation.cfm?id=2936924.2937171>

SILVESTRE, E. A.; DA SILVA, V. T. **An Approach to Verify Conflicts among Multiple Norms in Multi-agent Systems.** 2016 {IEEE/WIC/ACM} International Conference on Web Intelligence, {WI} 2016, Omaha, NE, USA, October 13-16, 2016. **Anais**...2016bDisponível em: <http://dx.doi.org/10.1109/WI.2016.0058>

SILVESTRE, E. A.; SILVA, V. T. DA. Conflict Detection among Multiple Norms in Multi-Agent Systems. **Applied Artificial Intelligence**, v. 0, n. 0, p. 1–31, 2018.

TANG, Y. et al. Tracking Control of Networked Multi-Agent Systems Under New Characterizations of Impulses and Its Applications in Robotic Systems. **IEEE Transactions on Industrial Electronics**, v. 63, n. 2, p. 1299–1307, fev. 2016.

TRYPUZ, R. **Krister Segerberg on Logic of Actions**. [s.l.] Springer Publishing Company, Incorporated, 2013.

VASCONCELOS, W. W.; KOLLINGBAUM, M. J.; NORMAN, T. J. Normative conflict resolution in multi-agent systems. **Autonomous Agents and Multi-Agent Systems**, v. 19, n. 2, p. 124–152, 2009.

VON WRIGHT, G. H. Deontic Logic. **Mind**, v. 60, n. 237, p. 1–15, 1951.

WOOLDRIDGE, M. **An Introduction to MultiAgent Systems**. 2nd. ed. [s.l.] Wiley Publishing, 2009.