Noname manuscript No.
(will be inserted by the editor)

Detection and Resolution of Normative Conflicts in Multi-agent Systems: a Literature Survey

Jéssica S. Santos[*] · **Jean O. Zahn** ·
Eduardo A. Silvestre · **Viviane T. Silva** ·
Wamberto W. Vasconcelos

Received: date / Accepted: date

Abstract Multi-agents systems are composed of autonomous and possibly heterogeneous software agents that act according to their own interests. Some coordination mechanism must be adopted to ensure a proper functioning of the whole system. Norms can be viewed as a powerful means to regulate and influence the behaviour of the agents by specifying, for instance, obligations, permissions, or prohibitions in a given context. A critical issue that must be considered in a system governed by multiple norms is the possible existence of normative conflicts. A conflict between norms is a situation in which the fulfilment of a norm causes a violation of another one. In this paper, we present several techniques that have been proposed to detect and resolve normative conflicts in multi-agent systems. Our aim is to organize the literature, present a classification of the techniques found, and provide a means to compare alternative approaches dealing with normative conflicts.

Keywords Norms · Conflict Detection · Conflict Resolution · Multi-agent Systems

1 Introduction

Interest in multi-agent systems has increased in the last decade since such systems present attractive means to model and design complex, concurrent and distributed systems. In [Russell and Norvig, 2009], a software agent is defined as an entity capable of perceiving its environment through sensors and acting upon that environment through actuators. Intelligent software agents can be classified according to the way they collect information and

[*]Corresponding author.

J. S. Santos · J. O. Zahn · E. A. Silvestre
Computer Science Department, Institute of Computing, Universidade Federal Fluminense (UFF), Av. Gal. Milton Tavares de Souza, s/nº, Niterói, Rio de Janeiro 24210-346, Brazil E-mail: {jsoares, jzahn, esilvestre}@ic.uff.br

V. T. Silva
IBM Research (on leave from UFF), Av. Pasteur, 138, Rio de Janeiro, Rio de Janeiro 22290-240, Brazil E-mail: vivianet@br.ibm.com

W. W. Vasconcelos
Department of Computing Science, University of Aberdeen, Meston Walk, Aberdeen AB24 3UT, United Kingdom E-mail: w.w.vasconcelos@abdn.ac.uk

act on the environment. In the case of multiple agents cooperating or competing with each other, operating within the same (virtual) environment and sharing information, we call this a multi-agent system (MAS) [Russell and Norvig, 2009]. MASs are autonomous and heterogeneous societies that work to achieve common and/or individual goals [Wooldridge, 2009].

In open MASs, agents are independently designed and act according to their own interests. A form of regulation is often adopted in order to restrict the behaviour of heterogeneous and autonomous agents and achieve the overall goal of the multi-agent system [García-Camino et al., 2005]. Norms can be applied to regulate such systems influencing and restricting the behaviour of their agents but not directly interfering with their autonomy. Norm-governed agents are able to reason about norms and choose their actions following or not following obligations, permissions, and prohibitions [Kollingbaum, 2005].

However, there is the possibility of conflicts arising among norms. A normative conflict arises when the fulfilment of one norm causes the violation of another. The most common cases of conflict occur (i) between a norm that obliges and another one that prohibits the same behaviour; or (ii) between a norm that prohibits and another one that permits the same behaviour. When there is a normative conflict, whatever the agents do or refrain from doing may lead to a state that is not norm-compliant [Kollingbaum et al., 2008b; Vasconcelos et al., 2009].

This paper presents the techniques found in the literature for the detection and resolution of conflicts between norms in multi-agent systems and points out the limitations and shortcomings of these approaches. Our presentation targets practitioners (*i.e.*, designers and developers of multi-agent systems, knowledge engineers, requirement analysts, and so on) as well as those studying normative multi-agent systems from a logic-theoretic, philosophical or legal perspective (*i.e.*, logicians, philosophers, policy-makers, lawyers, and so on).

We classify the lines of research studied as (i) approaches that deal with normative conflicts at *design time*, that is, the detection method is performed before the execution of the MAS in order to identify potential conflicts and avoid their occurrence; (ii) approaches that deal with normative conflicts at *runtime*, that is, the mechanism proposed is performed during the execution of the MAS; (iii) approaches that can identify *direct* conflicts, which are normative conflicts that can be detected through the analysis of the norm components; and (iv) approaches that can also identify *indirect* conflicts, which are conflicts that can only be detected when the conflict detection takes into account the characteristics of the application domain. Moreover, we also compare the norm expressiveness of the different approaches, i.e., the components adopted in their norm definition.

This paper is organized as follows. Section 2 presents a definition of normative multi-agent systems and introduces the concept of norm. Section 3 explores approaches that deal with detection of normative conflicts. Section 4 presents approaches that support resolution of normative conflicts. Some approaches are mentioned both in Section 3 and in Section 4 because they are able to detect and resolve normative conflicts. Section 5 exhibits a comparative analysis of detection and resolution techniques. Finally, Section 6 presents our conclusions and possible lines for further research.

2 Normative MAS

Norms are social concepts that determine behavioural patterns and are used to guide and control the performance of actions within a specific context [Kollingbaum, 2005]. A norm can explicitly establish a prohibition, an obligation or a permission over the behaviour of an entity in a given context. It specifies what is being regulated (that is, the performance of an action or the achievement of a state), when the regulation occurs (that is, the time, event or valid state), and who (namely, the agent, role, or group) is the subject of this regulation. Thus, norms consist of a way to obtain a desirable system behaviour [Grossi et al., 2010].

A normative MAS is a multi-agent system that explicitly represents norms in order to govern the behaviour of its software agents and maintain social order [Boella et al., 2006]. Such systems have norm enforcement mechanisms and are able to represent, interpret, police, manage and update norms.

Deontic logic [von Wright, 1951] is a modal logic with operators for permission, obligation, and prohibition. This logic was developed for reasoning about ideal behaviour. The concepts of deontic logic have traditionally been used for the analysis and reasoning about normative law and are widely used in normative MASs to formally describe norms and their modalities [Meyer and Wieringa, 1993; Meyer and Wieringa, 1994]. In deontic logic, the modality of a norm is called a deontic concept. It represents the nature of the regulation established by the norm, namely, a prohibition, a permission or an obligation.

There is not a consensus in the literature about which elements (e.g. role, action, agent) a norm should include in its representation. Each work puts forward a norm representation with different components. However, some components are common to several approaches. A norm is always associated with a deontic concept (obligation, permission or prohibition), regulates a given behaviour and is addressed to an entity. A norm can regulate the performance of an atomic action, a complex action (parameterized action or composed one) or the achievement of a state of affairs. Additionally, a norm is always associated with an entity that may be an agent, an organization (or group of agents) or a role. When the norm regulates the behaviour of an organization, only agents playing roles within it are subject to the regulation. When the norm is applied to a role, only agents playing such a role are subject to the regulation.

Some approaches determine that a norm can also be associated with a context. When the context is formally represented as a norm element, it indicates the circumstances or situations in which the norm operates. In such a case, the context may be an organization, an environment, or an interaction, for example. When the context is represented as part of the norm, the norm must be fulfilled only when the entities which the norm targets are in such a context. Outside this context, the norm is not applicable. If the context is not explicitly represented as part of the norm, the norm is applicable in all contexts of the MAS. Other approaches consider that norms may be associated with activation and deactivation conditions. In this case, the norm becomes active and must be fulfilled when the activation condition is fired and it becomes inactive when the deactivation condition is triggered. The activation and deactivation conditions can be states or events, such as a date, the execution of an action, the fulfilment of a norm, etc. If activation and deactivation conditions are not represented, the norm is always applicable. For instance, the work described in [da Silva et al., 2015] presents norms in a stock exchange scenario, where shares are bought and sold. One of the norms determines that an Intervening (i.e., a third party that is the intermediary between the parties involved in sale of shares) is obliged to agree to the terms applied in the contract, upon signature. This norm is accepted throughout the USA, activated on 11/09/2005 at 12:00:00 and deactivated on 11/09/2005 at 23:59:59. Following the representation of norms proposed by the authors, the context where the norm is applied is "USA", the entity whose behaviour is being regulated is "Intervening", the deontic concept is "obligation" and is applied over the behaviour of "agree the terms of the contract", the period of activation is between 11/09/2005 (activation condition) and 11/09/2005 (deactivation condition).

In order to enforce the fulfilment of the norms, some norms specify the sanctions that may be applied to punish agents when they violate a norm. Alternatively, rewards may be represented as incentives to agents when they fulfil a norm [Boella et al., 2006; Meneguzzi and Luck, 2009]. Some approaches allow variables in some/all of their components, conferring generality and compactness on the representation, as variables parameterize a norm specification. Some approaches associate variables with constraints [Aphale et al., 2013; Şensoy et al., 2012; Vasconcelos et al., 2009; Vasconcelos and Norman, 2009], restricting their value.

Table 1 and Table 2 list the components of norms from different approaches. These tables convey the following information. Each row shows a proposal (with its bibliographic reference) and the columns are individual norm components. The approaches mentioned will be presented in the next sections. The aim of Tables 1 and 2 is to illustrate the number of approaches using the same components to represent a norm. We have chosen to list those norm components which appear in more than one paper by different authors, leaving out, for instance, the declaration time (presented in [Vasconcelos et al., 2009] and which defines the time when the norm was introduced in the system), state of the norm (described in [da Silva and Zahn, 2014; Zahn and da Silva, 2014] and determines whether the norm has been fulfilled or violated), social context (proposed in [Oren et al., 2008] as the social entity that imposes the norm), and regulation (the research in [Cholvy and Cuppens, 1998] establishes that norms are addressed to regulations), as these are particular to one proposal only. In Table 2 we only list the kinds of activation condition that appear in more than one approach (time, state, action, context), but an activation condition can also be represented as a conjunctive semantic formula [Aphale et al., 2013; Şensoy et al., 2012], a set of literals in conjunctive or disjunctive form [Boella et al., 2012] or a formula of propositional logic [Broersen et al., 2001a; Broersen et al., 2001b].

Note that some approaches are mentioned in Table 1 but are not mentioned in Table 2. This is because those approaches do not consider any of the norm elements listed in Table 2 (activation and deactivation conditions, rewards, sanctions, context, constraints).

Table 1 Norm components of different approaches 1-2

		Regulated entity			Regulated behaviour		Norm language		
	Deontic concept	Agent	Role	Organization	Action	State	Parameters 1st order	Propositional	Logical operators
[Aphale et al., 2013]	✓	✓	✓		✓	✓	✓		✓
[Boella et al., 2012]	✓				✓	✓		✓	✓
[Broersen et al., 2001a]	✓	✓			✓	✓		✓	✓
[Broersen et al., 2001b]	✓	✓			✓	✓		✓	✓
[Cholvy and Cuppens, 1995]	✓		✓		✓			✓	✓
[Cholvy and Cuppens, 1998]	✓				✓			✓	✓
[Criado et al., 2010c]	✓				✓	✓		✓	
[da Silva et al., 2015]	✓	✓	✓	✓	✓			✓	
[da Silva and Zahn, 2014]	✓	✓	✓	✓	✓			✓	
[dos Santos Neto et al., 2012]	✓	✓				✓		✓	
[dos Santos Neto et al., 2013]	✓	✓	✓			✓		✓	
[Fenech et al., 2009]	✓				✓			✓	✓
[Fenech et al., 2008]	✓				✓			✓	✓
[Gaertner et al., 2007]	✓	✓	✓		✓		✓		
[García-Camino et al., 2007]	✓	✓			✓			✓	
[Garcia et al., 2013]	✓	✓	✓	✓	✓		✓		✓
[Giannikis and Daskalopulu, 2009]	✓	✓	✓		✓			✓	
[Giannikis and Daskalopulu, 2011]	✓	✓	✓		✓			✓	✓
[Günay and Yolum, 2013b]			✓			✓		✓	✓
[Kagal and Finin, 2005]	✓	✓			✓		✓		
[Kagal and Finin, 2007]	✓	✓			✓		✓		
[Kollingbaum and Norman, 2004]	✓		✓		✓	✓	✓		
[Kollingbaum and Norman, 2006]	✓		✓		✓	✓	✓		
[Kollingbaum et al., 2006]	✓		✓		✓	✓	✓		
[Kollingbaum et al., 2007]	✓		✓		✓	✓	✓		✓
[Li, 2014]	✓		✓		✓			✓	✓
[Oren et al., 2008]	✓	✓				✓		✓	
[Şensoy et al., 2012]	✓	✓	✓		✓	✓	✓		✓
[Vasconcelos et al., 2012]	✓	✓	✓		✓		✓		
[Vasconcelos et al., 2009]	✓	✓	✓		✓		✓		
[Vasconcelos and Norman, 2009]	✓	✓	✓		✓		✓		
[Zahn, 2015]	✓	✓	✓	✓	✓	✓		✓	
[Zahn and da Silva, 2014]	✓	✓	✓	✓	✓			✓	

Table 2 Norm components of different approaches 2-2

	Activation condition					Deactivation Condition				Rewards		Sanctions		Context		
	Time	State	Event	Action	Context	Time	State	Event	Context	Action	State	Action	State	Organiz.	Environ.	Constrs.
[Aphale et al., 2013]																✓
[Criado et al., 2010c]						✓				✓		✓				
[da Silva et al., 2015]	✓					✓								✓	✓	
[da Silva and Zahn, 2014]	✓					✓								✓	✓	
[dos Santos Neto et al., 2012]					✓				✓		✓		✓			
[dos Santos Neto et al., 2013]		✓					✓			✓		✓				
[Fenech et al., 2009]			✓	✓												
[Fenech et al., 2008]			✓	✓												
[Gaertner et al., 2007]						✓										
[Kagal and Finin, 2005]						✓										
[Kagal and Finin, 2007]						✓										
[Kollingbaum and Norman, 2004]		✓					✓									
[Li, 2014]		✓	✓					✓								
[Şensoy et al., 2012]																✓
[Vasconcelos et al., 2012]						✓										
[Vasconcelos et al., 2009]	✓					✓										✓
[Vasconcelos and Norman, 2009]																✓
[Zahn, 2015]	✓					✓								✓	✓	
[Zahn and da Silva, 2014]	✓					✓								✓	✓	

3 Conflict Detection

In this section, we present the approaches that focus on the identification of conflicts among norms within a MAS. We discuss those approaches in Sections 5 and 6. The first step towards the resolution of normative conflicts is their detection. This process is essential to the proper functioning of normative MASs.

The detection of conflicts may be done either at *design time* to avoid the occurrence of conflicts during the execution of the MAS, or at *runtime* wherein the MAS or the agents must be able to solve conflicts dynamically. The efficacy of tools used to detect normative conflicts depends on some factors, such as the norm expressiveness supported by the detection method and the ability of the proposed mechanism to reason about the relationships of the application domain.

Some approaches distinguish deontic conflicts and deontic inconsistencies. According to [Elhag et al., 2000], the former occur when a norm prohibits a certain behaviour that is obliged by another norm at the same time. In such cases, norm-compliant agents are unable to comply with the two norms in a consistent way and a violation will always occur, that is, if the agent chooses to comply with the obligation it will violate the prohibition and vice versa. According to [Elhag et al., 2000], deontic inconsistencies occur when there is a norm permitting a certain behaviour that is prohibited by another norm simultaneously. In this case, the agent may choose to comply with the prohibition and not perform the behaviour established in the permission, avoiding violations. It is possible because a permissive norm does not force its addressee to perform an action, i.e., the permission may not be acted upon. This point of view is based on the impossibility-of-joint-compliance test for conflict (IJC-test) [Hill, 1987], which determines that compliance statements represent the fulfilment of the content of a norm. For instance, the norm "obligatory p" corresponds to the compliance statement "p is done". Permissions are not susceptible for constructing compliance statements and for this reason a conflict analysis involving permissive norms may only indicate a possibility of conflict.

Conflicts between norms can also be classified as *direct* or *indirect* conflicts. A *direct* conflict arises between norms that regulate the behaviour of one agent and have opposite or contradictory deontic modalities, i.e., obligation versus prohibition, or, permission versus obligation. For instance, suppose that a norm is of the form: Da, where D is a deontic concept that represents an obligation (O) or a prohibition (F); and a represents the action being regulated. If an agent is associated with the norms Op and Fp, a direct comparison between the norm components can detect a conflict. On the other hand, an *indirect* conflict addresses conflicts between norms in which the component elements are not the same, but are related. Thus, in order to detect *indirect* conflicts, the conflict checker must consider the characteristics of the application domain [da Silva and Zahn, 2014]. For instance, if an agent is associated with the norms Oq and Fp and $q \rightarrow p$, there is an *indirect* conflict. The actions q and p are not the same but the action q implies the action p.

Conflicts may also arise between norms that have the same deontic concept. It occurs, for instance, when two activated norms oblige the same agent to execute actions that cannot be performed at the same time.

Table 3 shows the papers that check for conflicts at *design time* and those that check for conflicts at *runtime*, and the papers that only consider *direct* normative conflicts and those able to deal with some kind of *indirect* normative detection. The strategies to detect conflicts are summarized in Table 4. All detection methods presented in this section are only able to detect conflicts involving two norms. The approaches mentioned on Table 3 and Table 4 are briefly presented in the next subsections.

Table 3 Classification of approaches to detect normative conflicts

	When the approach is used		Kind of conflict detected	
	Design time	Runtime	Direct conflicts	Indirect conflicts
[Aphale et al., 2013]	✓		✓	✓
[Cholvy and Cuppens, 1995]				
[Cholvy and Cuppens, 1998]				
[da Silva and Zahn, 2014]				
[da Silva et al., 2015]				
[Fenech et al., 2009]				
[Fenech et al., 2008]				
[Şensoy et al., 2012]				
[Zahn, 2015]				
[Zahn and da Silva, 2014]				
[Garcia et al., 2013]	✓		✓	
[Li, 2013]				
[Li et al., 2014]				
[Li, 2014]				
[Vasconcelos and Norman, 2009]				
[Giannikis and Daskalopulu, 2009]		✓	✓	✓
[Giannikis and Daskalopulu, 2011]				
[Kollingbaum and Norman, 2004]				
[Kollingbaum and Norman, 2006]				
[Kollingbaum et al., 2006]				
[Kollingbaum et al., 2007]				
[Oren et al., 2008]				
[Vasconcelos et al., 2009]				
[dos Santos Neto et al., 2012]		✓	✓	
[dos Santos Neto et al., 2013]				
[Gaertner et al., 2007]				
[Günay and Yolum, 2013b]				
[Vasconcelos et al., 2012]				

Table 4 Different Strategies for Normative Conflict Detection

	Normaliz.	Ontology		Unific.	Constr. Satisf.	Subst.	Context Anticip.	Trace Analys.	Comp. analys.	Plan investig.	Model Check.
		Analysis	Consist. check								
[da Silva and Zahn, 2014] [da Silva et al., 2015] [Zahn, 2015] [Zahn and da Silva, 2014]	✓	✓							✓		
[Vasconcelos et al., 2009]				✓	✓	✓			✓		
[Vasconcelos and Norman, 2009]				✓	✓	✓			✓		
[Aphale et al., 2013] [Şensoy et al., 2012]		✓	✓			✓	✓		✓		
[Cholvy and Cuppens, 1995]									✓		
[Cholvy and Cuppens, 1998]									✓		
[Fenech et al., 2009] [Fenech et al., 2008]								✓			
[Li, 2013] [Li et al., 2014]								✓			
[Gaertner et al., 2007] [Vasconcelos et al., 2012]				✓		✓					
[Kollingbaum and Norman, 2004] [Kollingbaum and Norman, 2006] [Kollingbaum et al., 2006] [Kollingbaum et al., 2007]									✓	✓	
[Giannikis and Daskalopulu, 2009] [Giannikis and Daskalopulu, 2011]							✓		✓		
[dos Santos Neto et al., 2012]									✓		
[dos Santos Neto et al., 2013]									✓		
[Günay and Yolum, 2013b]					✓				✓		
[Oren et al., 2008]									✓		
[Garcia et al., 2013]											✓

3.1 Conflict Checker

The work in [da Silva and Zahn, 2014] presents the first version of the Conflict Checker[1], a Java program capable of detecting *direct* and *indirect* conflicts at *design time*. In this work, normative conflicts can be detected taking into account the system ontology that describes a set of norms and relationships of the application domain. The algorithm used by the Conflict Checker program is based on locally rewriting norms in order to transform these into an alternative format (normalization) and finding out if the norms overlap each other. This approach is similar to the unification method presented elsewhere [Fitting, 1990; Gaertner et al., 2007; Vasconcelos et al., 2009; Vasconcelos et al., 2012; Vasconcelos and Norman, 2009].

Five kinds of relationships among entities are supported, as follows: the relationship *inhabit* relates an entity to an environment, i.e., when a norm is applied in an environment, it is applied to all entities that inhabits that environment; the relationship *play* relates an entity to the roles that it can play, i.e., when a norm is applied to a role, it means that the norm is applied to all entities that are playing that role; the relationship *playIn* relates an entity to the organization where it is playing a role, i.e., when a norm is applied to an organization, it applies to all entities playing roles in the organization; the relationship *ownership* describes the roles within an organization, i.e., when a norm is applied to an organization, it applies to all roles being played in the organization; and the relationship *hierarchy*, when a norm is applied to a super-element, it is applied to all sub-elements of the super-element. This approach also supports two kinds of relationships among actions: *refinement* (actions that are specializations of other actions); and *composition* (actions composed of other actions). The work described in [Zahn and da Silva, 2014] extends the work in [da Silva and Zahn, 2014] by including two new kinds of relationships among actions: *orthogonality* (actions that cannot be executed at the same time by the same entity or related entities) and *dependency* (actions that are preconditions of other actions).

In [da Silva and Zahn, 2014] and [Zahn and da Silva, 2014] a norm is a tuple of the form $\langle Deoc, c, e, a, ac, dc, s\rangle$, where *Deoc* is the deontic concept; *c* is the context where the norm exists (organization or environment); *e* is the entity whose behaviour is being regulated; *a* is an atomic action being regulated; *ac* is the condition that activates the norm (a date); *dc* is the condition that deactivates the norm (a date); *s* is the state of the norm from the set $\{fulfilled, violated, none\}$. The state *none* indicates that the norm has not been fulfilled or violated yet. The norm components are propositions. The difference between these two approaches, in relation to the norm specification, is that in [da Silva and Zahn, 2014] the deontic concept can be either a prohibition or an obligation and in [Zahn and da Silva, 2014] the deontic concept can also be a permission.

The work in [da Silva et al., 2015] introduces OnCheckIn, which is an algorithm that can detect *direct* and *indirect* conflicts among norms. The idea of the authors is to adapt the Conflict Checker program in order to include propagation algorithms, which will propagate the regulations from a context (or entity) to a related context (or entity). After that, it analyzes relationships among the behaviours of the propagated norms. In [da Silva et al., 2015] the state of the norm is not represented as part of the norm.

For instance, consider an ontology describing a university domain. The University is an environment and UFF is an organization that inhabits that environment (relationship *inhabit*). The Computing Institute (CI) is a sub-organization of UFF (relationship *hierarchy*). Professor is a role of CI (relationship *ownership*). John is Professor at CI (relationship *play*).

[1] http://goo.gl/Qkutmo

Considering the relationships aforementioned, when the norm n1 = $\langle$*Permission*, *University*, *CI*, *CallforPapers*, _, _$\rangle$ is applied in the domain, a set of norms is generated through the propagation as follows:

n1.1 = $\langle Permission, UFF, CallforPapers, \_, \_\rangle$

n1.1.1 = $\langle Permission, CI, CallforPapers, \_, \_\rangle$

n1.1.1.1 = $\langle Permission, Professor, CallforPapers, \_, \_\rangle$

n1.1.1.1 = $\langle Permission, John, CallforPapers, \_, \_\rangle$

Due to the domain relationships, n1 is applied to the “University” and is propagated to be applied to “UFF” (norm n1.1). Since norm n1.1 is applied to “UFF” and “CI” is a sub-organization of “UFF”, n1.1 creates norm n1.1.1. Since “Professor” is a role played in “CI”, n1.1.1 creates norm n1.1.1.1. Finally, norm n1.1.1.1 is created because “John” is a “Professor”.

The work described in [Zahn, 2015] extends the Conflict Checker program to capture states in the description of norms. A state may be either a precondition for performing an action (condition state) or the result of the performance of an action (effect state). Thus, the performance of actions may affect and change states. This work considers the same relationships described in [Zahn and da Silva, 2014] and the same relationships among actions, which are extended in order to support states. Additionally, the verification of relationships among behaviours may occur: (i) between states; (ii) between an action and a state; and (iii) between actions. This occurs because the cycle of an action (condition state, action, effect state) must be considered in order to detect normative conflicts.

3.2 Unification and constraints [Vasconcelos et al., 2009]

The work presented in [Vasconcelos et al., 2009] integrates and extends the research presented in [Vasconcelos et al., 2007b; Kollingbaum et al., 2008a; Kollingbaum et al., 2008b; Vasconcelos et al., 2007a]. The definition of norm used was introduced in [Vasconcelos et al., 2007a], in which norms are associated with constraints. The work in [Vasconcelos et al., 2009] presents mechanisms to detect *direct* and *indirect* normative conflicts at *run-time* between prohibitions and obligations or permissions.

The authors use first-order unification to check if the variables of a prohibition overlap with the variables of an obligation or permission. If a substitution can unify the variable of both norms, the algorithm verifies if the conjunction of constraints can be satisfied and if the activation periods of the norms overlap. Thus the algorithm detects potential conflicts before they arise. This work is able to detect *indirect* conflicts that occur due to the relationships among actions (composition relationship, side effects, causal effects) and due to the delegation of tasks or norms among agents. The authors determine that the technique applied to unify is an algorithm which always terminates (possibly failing, if a unifier cannot be found), is correct, and has a linear computational complexity. The action/role relationships are declared by using a set of domain/delegation axioms.

A norm is represented as a tuple $\langle v, t_d, t_a, t_e\rangle$ where v is of the form $N_{\alpha:\rho}\varphi \circ \Gamma$; N is a deontic concept from the set $\{obligation, prohibition, permission\}$; α is an agent identifier; ρ is a role identifier, φ is a first-order atomic formula that represents an action with parameters; Γ is a set of constraints on those variables occurring in the atomic formula; t_d is the time when v was declared; t_a is the time when v becomes active; t_e is the time when v expires. The association between the first-order atomic formula φ and the set of constraints Γ is represented as $\varphi \circ \Gamma$. A norm can also be associated with an issuing authority.

The approach in [Vasconcelos et al., 2009] is applied in a rescue operation scenario: a simplified non-combatant evacuation scenario in which software agents help humans to coordinate their activities and to share information. In this scenario there are two coalition partners, viz., team A and team B, operating within the same area. Members of a non-governmental organization are stranded in a hazardous location and team A and team B must work together to evacuate these people to a safe location. The authors present two norms over action *deploy*(S,X,Y), establishing that sensor S is to be deployed on grid position (X,Y). The norms are as follows:

$$O_{A_1:R_1} deploy(s_1,X_1,Y_1) \circ \{10 \leq X_1 \leq 50, 5 \leq Y_1 \leq 45\}$$

$$F_{A_2:R_2} deploy(s_1,X_2,Y_2) \circ \{5 \leq X_2 \leq 60, 15 \leq Y_2 \leq 40\}$$

In the example above, the conflict occurs between a prohibition (F) and an obligation (O). The obligation has constraints $10 \leq X_1 \leq 50, 5 \leq Y_1 \leq 45$ and the prohibition has constraints $5 \leq X_2 \leq 60, 15 \leq Y_2 \leq 40$. By using the substitutions, we can "merge" the constraints as $10 \leq X_2 \leq 50, 5 \leq X_2 \leq 60, 5 \leq Y_2 \leq 45, 15 \leq Y_2 \leq 40$. The overlap of the merged constraints is $10 \leq X_2 \leq 60$ and $15 \leq Y_2 \leq 40$ and they represent respectively ranges of values for variables X_1, X_2 and Y_1, Y_2 where a conflict will occur.

3.3 Pre-emptive approach [Vasconcelos and Norman, 2009]

The work presented in [Vasconcelos and Norman, 2009] proposes a mechanism to analyze a contract and detect *direct* conflicts among norms at *runtime*. A contract is a set of rules that describe which norms must be added or removed according to the agents' actions. The left-hand side of the rule establishes the conditions that must hold for a norm to be added or removed. The right-hand side of the norm depicts which norms must be added or removed under such conditions. This approach is pre-emptive because the rules of a contract are analyzed beforehand for their potential conflicts. Thus, normative conflicts are considered before the contract is enacted.

The computational model presented in [Vasconcelos and Norman, 2009] stores all the actions performed and all the norms that currently hold in order to maintain a global history of the enactment of the society of agents. Thus, the computational model has a set of global enactment states $\triangle$, each one reflecting the state of the system in a given moment.

The mechanism used to detect conflicts is similar to the method presented in [Vasconcelos et al., 2009] since it also uses unification and constraint satisfaction to detect potential conflicts among norms. However, as stated before, [Vasconcelos et al., 2009] not only deals with *direct* conflicts but also copes with *indirect* conflicts. Two rules are in conflict if: (i) they add conflicting norms to the enactment state; and (ii) they can be triggered simultaneously.

Vasconcelos and Norman (2009) state that the unification algorithm is correct, always terminates (possibly failing, if there is no unifier found) and has linear complexity. However, such an algorithm was not presented in [Vasconcelos and Norman, 2009], hence we could not confirm this. The definition of norm is similar to that adopted in [Vasconcelos et al., 2009]. However, the norm representation of [Vasconcelos and Norman, 2009] does not include activation and deactivation conditions.

3.4 OWL-POLAR

The approaches introduced in [Aphale et al., 2013; Şensoy et al., 2012] are related to OWL-POLAR [Şensoy et al., 2010]. In [Aphale et al., 2013], the authors present the POLAR Agent application that is able to detect *direct* and *indirect* conflicts at *design time*. The conflict detection mechanism uses substitution and takes into account relationships of entity specialization. In addition, side-effects of actions and preconditions of actions are also considered to detect *indirect* conflicts. Since the operations to conflict detection are very expensive, this approach identifies the activities that are most relevant according to the goals of the organization/agent. The weight of each activity is calculated based on the frequency of occurrence of the activity in all the plans to achieve a given goal, the average cost of achieving the goal from the activity and the probability of successful execution of the activity.

The work in [Şensoy et al., 2012] extends OWL-POLAR [Şensoy et al., 2010] by providing mechanisms to deal with conflicts among OWL-POLAR policies. The language for semantic representation of norms is based on OWL-DL[2]. Idle policies, i.e., norms that are never activated or their expiration condition is satisfied whenever the norm is activated, can be detected. The detection algorithm identifies conflicting norms by anticipating contexts in which conflicts may arise. The problem of anticipating conflicts between two norms is transformed into an ontology consistency checking problem. The Pellet [Parsia and Sirin, 2004] reasoner checks the consistency of the ontology.

In [Aphale et al., 2013; Şensoy et al., 2012] norms are expressed using conjunctive semantic formulas (conjunction of atomic assertions). A conjunctive formula over an ontology associates the variables used in the assertions (concepts or relations from the ontology) with a set of constraints that restrict the values that they can assume. A norm is represented as $\alpha \to N\chi : \rho(\lambda : \varphi)/e$, where α is the condition that activates the norm; N is a deontic concept from the set $\{obligation, permission, prohibition\}$; χ is the policy addressee (χ may directly refer to a specific individual in the ontology or a variable); ρ is a role; $\lambda : \varphi$ is the regulated action or state; e is the condition that deactivates the norm. The activation and expiration conditions are conjunctive semantic formulas.

According to [Şensoy et al., 2012], the worst-case complexity of consistency checking OWL-DL is NEXPTIME-complete. The methods for anticipating conflicts and reasoning about idle policies are decidable but, due to the complexity, they are not tractable in OWL-DL in the worst-case.

3.5 Deontic Logic approach [Cholvy and Cuppens, 1995]

The work reported in [Cholvy and Cuppens, 1995] introduces a deontic logic for reasoning about norms, which is based on the concept of role. However, only the axiomatic part of this logic is described. The authors consider that a role is associated with a conflict-free set of norms. In this approach, the norms are associated directly with roles and agents inherit a set of norms when playing a given role. Thus, normative conflicts can only exist when an agent plays different roles and there is a conflict between the norms associated with these roles. Consequently, this work can relate an agent to the roles it plays and can detect *direct* and *indirect* conflicts at *design time*.

Norms are represented in the form Nrp, where N is a deontic concept from the set $\{obligation, prohibition, permission\}$; r is the role; p is the action being regulated. The norm

[2] http://www.w3.org/TR/owl2-overview/

components are propositions. This work distinguishes three kinds of normative conflicts among norms associated with different roles as follows:

1. Conflict between two obligations, one regulating an action and the other regulation the negation of this action ($Or_ip \wedge Or_j\neg p$);
2. Conflict between an obligation and a prohibition regulating the same action ($Or_ip \wedge Fr_jp$);
3. Conflict between one norm obliging an action and other permitting its negation ($Or_ip \wedge Pr_j\neg p$).

As an example of normative conflict, the authors consider the following norms:

(N1) A Christian ought not kill his neighbour.

(N2) If a Soldier is ordered to kill an enemy, then he ought kill him.

To represent these norms in this logic the authors consider that N1 applies only to a first role, namely r_1 = Christian and the second norm applies to another role, namely r_2 = Ordered_Soldier. This leads to the following specification $Or_1\neg Kill \wedge Or_2Kill$, and corresponds to the first case of normative conflict because the individual is obliged to kill and obliged not to kill.

3.6 FUSION Logic [Cholvy and Cuppens, 1998]

The work described in [Cholvy and Cuppens, 1998] is able to detect *direct* and *indirect* normative conflicts at *design time*. It states that an organization may be subject to several regulations and conflicts may arise among these regulations. For this reason, the authors present the FUSION logic for reasoning about norms when several possibly conflicting regulations are merged together. The paper assumes that each regulation is conflict-free.

FUSION logic can be viewed as a revision of the logic presented in [Cholvy and Cuppens, 1995], which did not make a clear distinction between deontic reasoning and the management of contradictions (i.e., non-monotonic reasoning). Norms are represented in the form Nrp, where N is a deontic concept from the set $\{obligation, prohibition, permission\}$; r is the associated regulation of the norm; p is the action being regulated. The norm components are propositions. The norm Orp means that within regulation r p is obligatory.

The authors introduce a new modality Br for representing the notion "within the regulation r" (or equivalently, "regulation r says that"). They define modalities Or and Pr by combining modalities Br (with formula Brp to read "regulation r says that p is true") with modalities Obligation (O) and Permission (P): $Orp \equiv BrOp$ and $Prp \equiv BrPp$. Orp and Prp are respectively to be read: "regulation r says p is obligatory (resp, permitted)". There are then two types of axioms: axioms which only concern the deontic modalities (O, P, F) and FUSION axioms which only concern Br.

FUSION logic is provided with a set of axioms and the associated semantics. Furthermore, the language is equipped with a set of inference rules. The axioms and the inference rules are used to prove if the norms are conflicting. One of the examples presented in [Cholvy and Cuppens, 1998] shows two regulations r_1 and r_2, where r_1 says that a is obligatory or b is obligatory (i.e. $Oa \vee Ob$) and r_2 says that a is not obligatory and b is not obligatory (i.e. $\neg Oa \wedge \neg Ob$). Applying the axioms and inference rules, we would derive $B_{r_1>r_2}Oa \vee Ob$ and $B_{r_1>r_2}\neg Oa \wedge B_{r_1>r_2}\neg Ob$ which are contradictory. The main drawback of this approach is that it is only able to merge sets of deontic literals (propositions or negations of propositions), i.e., it cannot be applied to disjunctive norms. Additionally, the authors show how conditional norms could be modeled in their approach. Although the authors cite as a possible future work to extend the FUSION logic in order to support domain constraints (to represent contradictory actions), we did not find this extension in the literature.

3.7 $\mathcal{CL}$ Contracts [Fenech et al., 2008]

The proposal of [Fenech et al., 2008] describes the use of multiple contracts within an organization. Each contract and each conjunction of contracts must be conflict-free. In this context, the authors developed a decision procedure to detect *direct* and *indirect* conflicts at *design time* in contracts written in $\mathcal{CL}$ [Prisacariu and Schneider, 2007], a formal language to specify deontic electronic contracts. Contract clauses in $\mathcal{CL}$ can represent either an obligation, or a prohibition or a permission. In order to detect conflicts, this work extends the trace semantic presented in [Kyas et al., 2008], which enables checking whether a trace satisfies a contract or not.

The analysis of conflicts can capture four different kinds of conflicts: (i) prohibitions and obligations regulating the same action; (ii) prohibitions and permissions regulating the same action; (iii) two obligations regulating mutually exclusive actions; (iv) obligations and permissions regulating mutually exclusive actions. The authors consider that each contract is conflict-free and the detection algorithm identifies potential normative conflicts between contracts. The detection method is complete and always terminates. This is explained in detail in [Fenech et al., 2009]. The algorithm generates an automaton that accepts all the traces that satisfy the contract. A normative conflict situation may result in reaching a state where the only option is to violate the contract. For this reason, any infinite trace that leads to a conflicting state will not be accepted by the semantics.

The contract clause is represented as follows:

$C := C_O|C_P|C_F|C \wedge C|[\beta]C|\top|\bot$

$C_O := O_C(\alpha)|C_O \oplus C_O$

$C_P := P(\alpha)|C_P \oplus C_P$

$C_F := F_C(\delta)|C_F \vee [\alpha]C_F$

$\alpha := 0|1|a|\alpha\&\alpha|\alpha \cdot \alpha|\alpha + \alpha$

$\beta := 0|1|a|\beta\&\beta|\beta \cdot \beta|\beta + \beta|\beta*$

A contract clause regulates an action and can be either an obligation (C_O), or a permission (C_P) or a prohibition (C_F) clause, or a conjunction of two clauses or a conditional clause. The action can be either an atomic action, or a compound action or concurrent actions. Atomic actions are represented by lower case letters, compound actions are represented by Greek letters and concurrent actions are represented by Greek letters with a subscript &. Compound actions are composed of atomic actions by using the operators: & (for actions occurring concurrently); $\cdot$ (for actions occurring in sequence); + (for a choice between actions); and $*$ (the Kleene star). 1 represents an action expression matching any action, while 0 represents the impossible action. When a contract clause is violated, then a reparation contract C must be executed; $[\beta]$C is interpreted as: if action β is performed then the contract C must be executed; if β is not performed, the contract is trivially satisfied. The activation condition is the occurrence of an event or the performance of an action. Additionally, mutually exclusive actions are represented as: a # b.

3.8 Institutional facts [Li, 2014]

In [Li, 2014; Li, 2013; Li et al., 2014], the authors emphasize that institutions can facilitate the development and regulation of open MASs. Institutions are normative frameworks that provide a means to govern agents in open distributed systems. Each institution has a set of norms to regulate specific situations and different institutions may cooperate to govern the same agents at the same time. However, in this context, conflicts may arise among norms of different institutions. Usually, this occurs when the agent is playing different roles in different institutions. These papers present means to detect *direct* normative conflicts among institutions at *design time*. The strategy for detecting conflicts focuses on composite institutions, which are institutions that have a common governance but are treated independently.

In [Li et al., 2014], a norm is formalised as a tuple of the form $\langle$*role*, *deontic*, *action*, *condition*, *deadline*$\rangle$, where *role* is a role identifier (all agents associated with this role are subject to the norm); *deontic* is a deontic concept from the set $\{obligation, permission, prohibition\}$; *action* is the institution action being regulated; *condition* is represented as $\langle \Sigma, E \rangle$, where Σ is a set of states under which the norm is applicable and E is a sequence of events; *deadline* is an event that establishes the deactivation condition of the norm. Additionally, obligations and prohibitions may be associated with sanctions.

Normative conflicts are classified as: (i) weak conflicts (between a permission and a prohibition); and (ii) strong conflicts (between an obligation and a prohibition). Weak conflicts may be avoided taking into account which actions are prohibited. Additionally, this approach is based on the formal declarative language Inst*AL* [Cliffe et al., 2007], which models the changes of the institutional states based on the interpretation of external events. The resultant model is converted into a logic program. Besides, the authors translated institutions into computational models using the declarative logic programming language of Answer Set Programming (ASP) [Gelfond and Lifschitz, 1991]. ASP programs are written in the AnsProlog language, in which the elements are atoms assigned with truth values, and which allows the negation of atoms by means of negation as failure.

A conflict is detected when an institutional fact is true in one institution and false in another one at the same time. Institutional facts are results of the events and the norms of the institutions. The detection mechanism generates all the possible event traces in order to verify which traces lead to conflicts.

3.9 The Normative Structure [Gaertner et al., 2007]

The work described in [Gaertner et al., 2007], proposes a normative model called Normative Structure (NS) in which normative positions (i.e., obligations, prohibitions or permissions) are propagated as consequences of the agents' actions. This approach considers that actions are carried out in a MAS by means of illocutionary speech acts [Searle, 1969] exchanged among agents of the system. It also presents a means to detect *direct* conflicts among norms at *runtime*.

Normative positions regulate illocutions, which are represented in the form $p(ag, r, ag', r', \delta, t)$, where p is an illocutionary particle (e.g. inform, request, offer); ag and ag' are agent identifiers; r and r' are role identifiers of the agents ag and ag', respectively; δ is an arbitrary ground term that represents an action with parameters; t is a time stamp. Illocutions may be partially instantiated, that is, they may contain variables in their representation. Normative positions are represented as $D(p(ag, r, ag', r', \delta, t))$, where D is a deontic concept from the set $\{obligation, permission, prohibition\}$. This normative position is interpreted as: agent ag that plays the role r is obliged/prohibited/permitted to send the message δ of the type p to the agent *ag*' that plays the role r', at time t.

The authors map the NSs into Coloured Petri Nets (CPNs) and use well-known theoretical results from work on CPNs in order to prove that ensuring conflict-freedom of a NS at *design time* is computationally intractable. Additionally, in order to reflect the nature of a MAS, this work proposes a means to deal with normative conflicts in a distributed manner. Normative Structures model the coordination level (agents' interactions) separately from the normative level (propagation of normative positions).

This proposal makes a distinction between conflicts and inconsistencies, adopting the same view as [Elhag et al., 2000]. The detection mechanism identifies only conflicts between prohibitions and obligations or permissions. To detect conflicts, this approach uses the standard notion of unification [Fitting, 1990], that is, two normative positions are in conflict if, and only if, they have contradictory deontic modalities and there is a substitution that unifies their illocutions. The authors do not present the algorithm but assume that the unification method is an algorithm: (i) that always terminates (failing if there is no substitution that unifies the norms); (ii) is correct; and (iii) has linear computational complexity.

The work in [Vasconcelos et al., 2012] extends the work in [Gaertner et al., 2007]. It provides a novel semantics for normative positions and formalises the notion of norm violation. The representation of norms and the method for detecting normative conflicts are the same as in [Gaertner et al., 2007].

3.10 The NoA Architecture

The work reported in [Kollingbaum and Norman, 2004; Kollingbaum et al., 2006; Kollingbaum and Norman, 2006; Kollingbaum et al., 2007] presents the NoA architecture. NoA can capture *direct* and *indirect* normative conflicts between prohibitions and permissions or obligations at *runtime*. In [Kollingbaum and Norman, 2004] the authors introduce this architecture, which is inspired by the Belief-Desire-Intention (BDI) model [Wooldridge, 2009] and was designed to support the implementation of norm-governed practical reasoning agents. NoA is a reactive planning architecture that provides a means to preserve the consistency of the set of norms associated with an agent. The authors consider that during the norm adoption process, normative conflicts may arise. For this reason, before adopting a new norm, the agent must check if this norm is consistent with the set of norms currently held. In this work, the norm addressee is a role and norms regulate activities. An activity may be either the performance of an action or the achievement of a state of affairs.

In [Kollingbaum et al., 2007] the authors state that in open Virtual Organizations (VOs) agents may sign many contracts when they adopt more than one role, for example. Due to the dynamic nature of coalitions, normative conflicts may arise in these systems and agents must have mechanisms to detect them. An agent is able to fulfil its obligations in a consistent manner only if there are no conflicts among its set of norms. In this approach, conflicts and inconsistencies are distinguished. The former occurs when an agent wants to perform an action that is simultaneously permitted and forbidden. On the other hand, the latter is the situation in which an agent wants to perform an action that is simultaneously obliged and forbidden. This distinction between the concepts of conflicts and inconsistencies is opposite to the distinction presented in [Elhag et al., 2000].

In [Kollingbaum and Norman, 2006], the authors state that NoA is capable of informing agents about conflicts and inconsistencies. In this architecture, normative statements may refer to partially instantiated actions or states using variables within norm specifications.

Normative conflicts, in general, can be detected by investigating the intersection of sets of actions or states that are addressed by partially or fully instantiated activity statements within norm specifications. In addition, the detection of *indirect* normative conflicts is performed by investigating all possible plans that can be chosen to achieve a state of affairs and by analyzing the side-effects of these plans. Thus, in order to avoid normative conflicts, agents must analyze the consequences of the selection and execution of a specific plan. In addition, NoA creates permissive norms (implicit permissions) that allow agents to perform actions (or achievement of states), which are not associated with prohibition norms.

The authors establish some special cases of conflict: (i) conflict between an implicit permission and an obligation; (ii) conflict between an explicit permission and an obligation; (iii) *indirect* conflicts/inconsistencies; (iv) containment: the scope of a norm is contained within the scope of another one, a relationship of specialization is regarded between the two norms and this kind of conflict is also called parent-child conflict; and (v) intersection: the scope of one norm intercepts the scope of another one and the activities in the intersection of the scopes inherit both norms. This kind of conflict is also called multiple-inheritance problem.

In addition, NoA agents have labelling mechanisms to classify the options for actions as consistent or inconsistent, according to their current set of norms. These mechanisms are described in [Kollingbaum and Norman, 2006] and are related to the concept of "informed" deliberation. The agents use labels to reason whether an action is norm-compliant or not. A plan instantiation is a consistent candidate if: (i) it is not a currently forbidden action; (ii) none of its effects is a forbidden state of affairs; and (iii) none of its effects counteracts any active obligation.

The norm is represented as $N(r, a, ac, ec)$, where N is a deontic concept from the set $\{obligation, permission, prohibition\}$; r is a role (norm addressee); a is the activity being regulated, which is the execution of an action with parameters or the achievement of a state of affairs; ac is the condition that activates the norm; ec is the condition that deactivates the norm. The activation and expiration condition are states of affairs. The norm components allow parameterization.

3.11 Reiter's Default Logic approach [Giannikis and Daskalopulu, 2011]

The research described in [Giannikis and Daskalopulu, 2009; Giannikis and Daskalopulu, 2011] presents strategies to detect, at *runtime*, *direct* and *indirect* normative conflicts that arise for agents engaging in electronic contracting.

The authors argue that developing formal models of electronic exchanges between entities engaging in electronic exchanges is a key issue within the electronic commerce community. Software agents can conduct such exchanges in electronic marketplaces or virtual communities. A contract creates a mutual legal relation between the involved parties and determine what actions are obligatory, permitted, forbidden and empowered.

The term *empowered* refers to the situation in which agents are authorized by an institution to create or modify facts that have a conventional significance within that institution. For instance, a priest has legal power (i.e., he was empowered by the church) to perform a wedding. In other words, an action performed by an agent that is empowered to perform it, counts as establishing a certain institutional fact.

This research emphasizes that, in order to avoid conflicts, it is important to evaluate the consequences of committing to a contract. The authors also argue that representing contractual norms as default rules facilitates conflict detection.

The norms of an agreement are represented as default rules using Reiter's Default Logic [Reiter, 1980], which allows the detection of normative conflicts by analyzing extensions. The idea consists of deriving extensions to identify primitive patterns of normative conflicts. The computation of extensions is like a preview of the possible worlds in which the agent can find itself and is performed in order to detect potential conflicts.

Norms are represented as $NN(a1,r1,c,a2,r2)$, where NN is either a deontic concept from the set $\{obligation, permission, prohibition\}$ or legal power; $a1,a2$ are agent identifiers; $r1,r2$ are the roles associated with the agents $a1$ and $a2$, respectively; and c is the action being regulated. The meaning of the norm is: agent $a1$ that acts as $r1$ is in legal relation NN toward agent $a2$ that acts as $r2$. The six primitive patterns are as follows:

1. Conflict between a deontic concept and its negation, where the norms refer to the same agent and regulate the same action: This situation occurs when an agent has contradictory knowledge, namely, $NN(a1,r1,c,a2,r2) \times \neg NN(a1,r11,c,a2,r22)$.
2. Conflict between the prohibition to perform an action and the simultaneous obligation or permission to perform the same action: $prohibition(a1,r1,c,a2,r2) \times permission(a1, r11,c,a2,r22)$ or $prohibition(a1,r1,c,a2,r2) \times obligation(a1,r11,c,a2,r22)$.
3. Conflict between an obligation to perform an action and the simultaneous obligation or permission to perform the negation of this action: $obligation(a1,r1,c,a2,r2) \times obligation(a1,r11,\neg c,a2,r22)$;
4. Conflict between the institutional power to perform an action and the prohibition to perform the same action: $power(a1,r1,c,a2,r2) \times prohibition(a1,r11,c,a2,r22)$.
5. Conflicts between two obligatory actions that cannot be performed at the same time: $obligation(a1,r1,c1,a2,r2) \times obligation(a1,r11,c2,a22,r22)$.
6. Conflict between an obligation and the negation of the agent's legal power or permission to perform it: $obligation(a1,r1,c,a2,r2) \times permission(a1,r11,c,a2,r22)$ or $obligation(a1, r1,c,a2,r2) \times power(a1,r11,c,a2,r22)$.

3.12 ANA

In [dos Santos Neto et al., 2012], the authors introduce an architecture that extends the BDI model by including a means to enable agents to reason with and about norms. This proposal includes a mechanism to detect *direct* normative conflicts at *runtime* between prohibitions and obligations. The use of Autonomous Normative Agents (ANA), which are BDI agents capable of adopting norms and reasoning about them in an autonomous manner has been encouraged to cope with this issue. The normative reasoning process of ANA agents is significantly more challenging than the reasoning process of traditional BDI agents. The proposed architecture was specified with the language Z [Diller, 1990]. In this architecture, the agents are able to check for incoming perceptions and norms, adopt new norms, check the activation and deactivation condition of the norms, and detect the fulfilment and violation of the norms.

An ANA agent is an entity that has a name, is playing a role, has a set of beliefs and a set of intentions, is associated with a set of norms, can achieve a set of goals (environment state), and has a set of motivations to achieve each goal. These motivations represent preferences and are either positive or negative (numerical) values. Positive values indicate that the agent is interested in achieving the given state, and negative values indicate that the agent does not want to achieve it. In addition, ANA agents store information about the current normative situation of a norm (i.e., if the norm is adopted, activated, deactivated, fulfilled or violated) and if a norm was selected to be fulfilled or violated.

The norm is specified as a tuple of the form $\langle a, Deoc, ac, ec, s, r, p\rangle$, where a represents the norm addressee; $Deoc$ is a deontic concept from the set $\{obligation, prohibition\}$; ac is the condition that activates the norm; ec is the condition that deactivates the norm; s is the state being regulated; r is a set of rewards; p is a set of punishments. The activation and deactivation conditions are contexts. The rewards and punishments describe the achievement of environment states. The norm components are described as propositions.

To detect if there is a conflict between two norms, the algorithm verifies if the two norms, one being a prohibition and the other being an obligation, are both activated, have the same addressee and regulate the same state.

3.13 The NBDI Architecture

The work described in [dos Santos Neto et al., 2013] presents the Norm-Belief-Desire-Intention (NBDI) architecture, which is an abstract architecture that provides a means to develop goal-oriented normative agents and extends the BDI architecture [Weiss, 1999] to help agents reason about the norms. This architecture has a component responsible for detecting *direct* normative conflicts at *runtime*. Such a component verifies the existence of two different norms (one being an obligation and the other one a prohibition) that specify the same state. After that, the mechanism verifies if the agent intends to fulfil both norms or violate both norms; only in these two cases will the norms be in conflict.

The NBDI architecture performs the following steps: An agent perceives information about the world by using its sensors and the agent's desires are updated, if it is the case. In the following, the activation conditions of the norms are evaluated and the norms that the agent has the intention to fulfill are selected. Additionally, the agent can identify and solve the conflicts among the selected norms. Next, desires that will become intentions are selected by taking into account the norms the agent wants to fulfil – dropping any intention that does not bring benefits to the agent. Finally, the actions established by the agent's intentions are executed.

The representation of norm is as described in [dos Santos Neto et al., 2010]. It is a tuple of the form $\langle a, ac, ex, r, p, Deoc, s\rangle$, where a is an agent or a role responsible for fulfilling the norm; ac is the condition that activates the norm; ex is the condition that deactivates the norm; r are rewards to be given to the agent for fulfilling a norm; p are the punishments to be given to the agent for violating a norm; $Deoc$ is a deontic concept from the set $\{obligation, prohibition\}$; s is a set of states being regulated by the norm. The activation and expiration conditions of the norm are states. The norm components are represented as logical propositions.

3.14 Commitment conflicts [Günay and Yolum, 2013b]

The work presented in [Günay and Yolum, 2013b] emphasizes that the explicit representation of organizations is a promising way to design a MAS. Usually organizations are specified via roles to establish the normative positions of the agents. Therefore to guarantee the proper functioning of an organization it is essential to check if the roles have been designed correctly. The authors assume that an agent may be associated with more than one role. For this reason, this approach checks for possible normative conflicts among the roles of an organization and within a specific role. The obligations and prohibitions associated with the roles are represented through commitments, as explained below. Situations that would lead to *direct* conflicts among commitments are identified. In addition, the authors use the feasibility concept of commitments to identify unfeasible situations at *runtime*.

C (x, y, q, p) represents a commitment from the debtor agent x to the creditor agent y to bring about the consequent p, if the antecedent q holds. The consequent of a commitment is a single proposition, and the antecedent can be a conjunction of propositions. The antecedent is the activation condition to the commitment. When the consequent starts to hold, the commitment is considered to be fulfilled. Consider an example in a conference organization domain. A member of conference's program committee (PCMember) is committed to review a paper (PaperReviewed), if the program chair (Chair) assigns the paper to the program committee (PaperAssigned). Moreover, an agent cannot review a colleague's paper. The obligation is represented as C(PCMember, Chair, PaperAssigned, PaperReviewed) and the prohibition is represented as C (PCMember, Chair, AuthorIsColleague, $\overline{\neg PaperReviewed}$).

Given two commitments c$_i$ = C(x, y, w, u) and c$_j$ = C(x, y', w', u'), c$_i$ and c$_j$ conflict with each other, denoted by ci $\otimes$ cj, if: it is not the case that w $\rightarrow$ q and w' $\rightarrow$ $\neg$q; and it is the case that u $\rightarrow$ p and u' $\rightarrow$ $\neg$p. The first condition states that two commitments may conflict only if their antecedents are not inconsistent, i.e., it is possible that the commitments become active at the same time. The second condition specifies that two commitments may be in conflict only if their consequences are inconsistent, and it occurs when the two commitments are contradictory.

The authors do not discuss in this approach how to compute the feasibility of the commitments. However, in other work [Günay and Yolum, 2013a], constraint satisfaction methods were applied to deal with it.

3.15 Normative Conflict Graph [Oren et al., 2008]

In the research described in [Oren et al., 2008] the authors examine how an agent may reason about norms and normative conflicts. This approach can detect *direct* and *indirect* conflicts among norms at *runtime*. The authors assume that when an agent adopts a norm it will attempt to comply with it. Additionally, this work does not consider conditional norms and temporal elements in the norm representation. Norms are always addressed to a single agent.

Norms are represented with logical propositions and are of the form $N_c(g)$, where N is a deontic concept from the set $\{obligation, permission, prohibition\}$; c is a social context, which corresponds to the social entity that imposes the norm (this parameter may be used to determine the importance of the norm). The social entity can be an agent, a group of agents or other social structures; and g is the normative goal, which is a state of affairs.

In order to illustrate the idea of social context, consider the following example: Alice is obliged to write a paper for her boss. In this case, the social context is the boss of Alice. The authors argue that the social context of a norm is an important element because it can influence the way the agent deals with the norm.

This approach can detect *indirect* conflicts taking into account mutually exclusive relations, which specify states that cannot be achieved at the same time. Each agent is able to construct a normative conflict graph from its set of norms. The nodes of the normative graph represent norms and each edge represents a conflict between nodes/norms. Then, a graph without edges is a graph free of conflicts.

In order to construct the normative conflict graph, each agent searches for three kinds of conflicts in its set of norms: (i) two obligations regulating mutually exclusive normative goals; (ii) a prohibition and an obligation regulating the same normative goal; and (iii) a prohibition and a permission regulating the same normative goal. The authors present the algorithm for constructing the normative conflict graph and detecting conflicts.

3.16 ROMAS CASE tool

The research reported in [Garcia et al., 2013] presents the Regulated Open Multi-Agent Systems (ROMAS) CASE tool, which is a tool for designing systems in which heterogeneous software agents may need to coexist in a complex social and legal framework. The ROMAS CASE tool can detect *direct* conflicts among organizational norms and agent norms (derived from the norms of its signed contracts) at *design time*.

In this approach, organizations establish norms and contracts to impose limits on the actions that can be performed. Additionally, agents can offer services (functionalities) to other agents. The ROMAS CASE tool presents a graphical interface to model ROMAS systems, provides a means to verify the model created by using model-checking techniques, and also generates the executable code from the model for agent platforms, such as Electronic Institutions [Sierra et al., 2004].

The ROMAS CASE tool has a mechanism to check the coherence, the completeness and the correctness of the designed models. The process to detect conflicts translates the modeled system into a language that can be verified using model-checking: the ROMAS model is translated into PROMELA code and Linear Temporal Logic (LTL) formulas which are languages that can be verified by the SPIN model-checker [Holzmann, 2004], which is a plug-in integrated to the CASE tool. After that, SPIN verifies the coherence of the legal context, i.e., it is able to detect conflicts among norms of the system.

A norm is represented as $\langle ID, Activation, Expiration, Target, Deontic, Action, Sanction, Reward\rangle$, where *id* is the norm identifier; *activation* and *expiration* are conditions that respectively activate and deactivate the norm. Commonly, activation conditions are specified after conditional particles (e.g. if, when, etc.), and deactivation conditions are specified after temporal conditional and exception particles (e.g. unless, until, except when, etc.); *target* can be an agent, a role, or an organization; *deontic* is a deontic concept from the set $\{obligation, permission, prohibition\}$; *action* is the action being regulated by the norm; and *sanction* and *reward* are the sanctions and rewards that can be associated with the norm. The authors do not explain how sanctions and rewards are represented.

The authors consider that there are four important kinds of conflicts: (i) conflict between an obligation and a prohibition to perform the same action; (ii) conflict between a permission and a prohibition to perform the same action; (iii) conflict between two obligations to perform contradictory actions; and (iv) conflict between a permission and an obligation to perform contradictory actions. However, the program can only detect *direct* conflicts (i) and (ii), i.e., conflicts between a prohibition and an obligation or permission to perform the same action. When a conflict is detected, the system designer can revise his model, change it by using the graphical interface of the ROMAS CASE tool and rerun the detection process.

4 Conflict Resolution

As stated in Section 3, there are several approaches that propose different mechanisms for the detection of normative conflicts. Besides detecting the conflicts, it is necessary to solve such conflicts in order to avoid unintentional violations, i.e., violation of norms that cannot be avoided by the agent. When two norms are in conflict, by stating for instance a prohibition and an obligation (or permission), whether the agent executes the behavior being regulated or not, the agent will violate one of the norms. Therefore, several approaches have been proposed for the resolution of conflicts. Some of these 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. Table 5 summarises our analysis of the literature along these dimensions. Our aim in this section is only to present the resolution approaches. A discussion of those approaches is provided in Sections 5 and 6.

Table 5 Classification of approaches to resolve normative conflicts

	When approach used		Kind of conflict solved	
	Design time	Runtime	Direct	Indirect
[Aphale et al., 2013] [Cholvy and Cuppens, 1995] [Cholvy and Cuppens, 1998] [García-Camino et al., 2007] [Şensoy et al., 2012]	✓		✓	✓
[Li, 2013] [Li, 2014]	✓		✓	
[Giannikis and Daskalopulu, 2009] [Giannikis and Daskalopulu, 2011] [Kollingbaum and Norman, 2004] [Kollingbaum et al., 2006] [Kollingbaum et al., 2007] [Oren et al., 2008] [Vasconcelos et al., 2009] [Vasconcelos and Norman, 2009]		✓	✓	✓
[Boella et al., 2012] [Broersen et al., 2001a] [Broersen et al., 2001b] [Criado et al., 2010c] [dos Santos Neto et al., 2012] [dos Santos Neto et al., 2013] [Gaertner et al., 2007] [Günay and Yolum, 2013b] [Kagal and Finin, 2005] [Kagal and Finin, 2007] [Vasconcelos et al., 2012]		✓	✓	

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 and Cuppens, 1995] present two norms: (N1) a Christian ought not kill his neighbour; 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 et al., 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 et al., 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. Table 6 shows the strategies used by the approaches that are described in the next subsections.

Table 6 Different Strategies for Normative Conflict Resolution

	Norm Prioritization				Norm Update	
	Lex posterior	Lex specialis	Lex superior	Other	Extending Scope	Reducing Scope
[Aphale et al., 2013] [Şensoy et al., 2012]	✓	✓	✓	✓		
[Cholvy and Cuppens, 1995]				✓		
[Cholvy and Cuppens, 1998]				✓		
[Oren et al., 2008]				✓		
[García-Camino et al., 2007]	✓	✓		✓		
[Li, 2013] [Li, 2014]				✓		
[Kagal and Finin, 2005] [Kagal and Finin, 2007]				✓		
[Broersen et al., 2001b] [Broersen et al., 2001a]				✓		
[Kollingbaum and Norman, 2004] [Kollingbaum et al., 2006] [Kollingbaum et al., 2007]	✓			✓	✓	✓
[Vasconcelos et al., 2009]		✓	✓			✓
[Vasconcelos and Norman, 2009]						✓
[Gaertner et al., 2007] [Vasconcelos et al., 2012]						✓
[Criado et al., 2010c]				✓		
[Giannikis and Daskalopulu, 2009] [Giannikis and Daskalopulu, 2011]	✓	✓	✓	✓		
[Günay and Yolum, 2013b]				✓		✓
[Boella et al., 2012]				✓		
[dos Santos Neto et al., 2012]				✓		
[dos Santos Neto et al., 2013]				✓		

4.1 OWL-POLAR

The work described in [Aphale et al., 2013] (introduced in Section 3.4) can resolve *direct* and *indirect* normative conflicts at *design time*. The strategies to solve conflicts use a policy refinement method that consists of determining an order of norm overruling after determining norm precedence based on standard techniques.

The work in [Şensoy et al., 2012] states that strategies, such as *lex superior*, *lex posterior* and *lex specialis* may be adopted to resolve conflicts among OWL-POLAR policies. The *lex posterior* strategy can only be adopted when the conflicting norms have been issued by the same authority (because temporal relationships between different authorities may be misleading). The *lex specialis* strategy can only be applied when a more specific norm can be considered an exception of another and the conflicting norms belong to the same organization. The authors present a subsumption reasoning algorithm that verifies if one norm is a specialization of another.

In addition, the paper presents another strategy to resolve conflicts among norms that refines the expiration conditions of the norms. In this case, the norm addressee may use an automated planner to find a plan whose actions imply a state of the world in which the expiration condition of one of the conflicting norms holds. In order to determine which plan to choose, plans can be ranked based on their cost for the norm addressee. The algorithm to find plans to expire norms is also presented.

The research in [Şensoy et al., 2012] presents the complexity of the algorithms proposed to resolve conflicts. The worst-case complexity of the planning method for resolving conflicts is PSPACE-complete.

4.2 Deontic Logic approach [Cholvy and Cuppens, 1995]

The work described in [Cholvy and Cuppens, 1995] (introduced in Section 3.5) is able to resolve *direct* and *indirect* normative conflicts at *design time* among norms associated with different roles.

When there is a normative conflict between two roles and it is possible that an agent plays these roles simultaneously, a judgment of priorities between them is needed to decide which norms to adopt in a given situation. It determines a total order between the roles. For instance, if there is a conflict between roles r_1 and r_2 and the judgment of priorities states that $r_1 > r_2$, then all norms associated with r_1 take precedence over the norms associated with r_2.

The judgment of priorities may depend on the individual (especially in moral dilemmas) or may be derived from the hierarchy of roles (sub-roles, sub-ideal-roles). The authors also consider that an action is not obligatory if there is no norm that explicitly obligates it, and an action is not permitted if there is no norm that explicitly permits/obliges it.

4.3 FUSION Logic [Cholvy and Cuppens, 1998]

The work in [Cholvy and Cuppens, 1998] (introduced in Section 3.6) presents a resolution method to deal with *direct* and *indirect* normative conflicts among regulations at *design time*.

The resolution strategy consists of establishing an order of prioritization between the regulations. This prioritization determines which norm has precedence and may be based on many grounds, such as, specificity. For instance, let us suppose that there is a regulation r_1

which characterizes the behaviour of people who are eating. This regulation states that one should not eat with his fingers and should put his napkin on his lap:

$$r_1 = \{O\neg \mathit{fingers}; \mathit{Onapkin}\}$$

Suppose there is a regulation r_2 which states that a person eating asparagus is permitted to eat with his fingers:

$$r_2 = \{\mathit{Pfingers}\}$$

There is a conflict between regulations r_1 and r_2. If regulation r_2 has priority over regulation r_1 ($r_2 > r_1$), then the individual is permitted to eat with his fingers. On the other hand, if regulation r_1 has priority over regulation r_2 ($r_1 > r_2$), then the individual should not eat with his fingers and he is obliged to put his napkin on his lap.

The authors have also developed a propositional modal logic called majority fusion (MF), described in [Cholvy and Garion, 2004], developed to deal with conflicts that may arise when several information sources are merged. The authors argue that merging information from different sources may lead to an inconsistent knowledge base due to contradictory information. When this occurs, some information can be discarded in order to maintain the consistency of the knowledge base. When the data to be merged is provided by different software agents, the authors argue that if the reliability of the agents is known, then it must be considered during the merging process (as described, for instance, in [Cholvy, 2006]). However, if the reliability of the sources is not known, the merging process is done according to a majority approach. For instance, suppose that there are three information sources db_1, db_2 and db_3 to be merged, db_1 and db_2 say that “John is a student” and db_3 says that “John is not a student”. Then, the merged source will conclude that “John is a student” (since two of the sources agree with it). The merged source can only conclude something when the majority of the sources agree with a given information.

4.4 Normative Conflict Graph [Oren et al., 2008]

The work presented in [Oren et al., 2008] (introduced in Section 3.15) is able to resolve *direct* and *indirect* normative conflicts at *runtime*. This research presents a model of normative agents in which agents have a set of norms and a set with a partial ordering over the social contexts. This determines the importance of complying with norms imposed by different social contexts so that a norm is prioritized.

The authors argue that when two norms are in conflict, the agent should determine which norm to violate in a way that maximizes its compliance with the remaining set of norms. In order to resolve conflicts, agents may prune some edges of the normative conflict graph based on their social context preferences. Agents can also perform the norm pruning taking into account the norm modality (obligation, permission or prohibition). For instance, an agent can prefer to comply with obligations rather than prohibitions. Then, a normative agent can also have a set with a partial ordering over norm modalities.

After the pruning, conflicts may still exist. In this case, normative conflicts are resolved by dropping conflicting norms. This work also presents three heuristics to solve conflicts. One is a random strategy and the others are argumentation-based. In the random strategy an edge (n, m) is selected randomly and the lower priority node of the conflicting pair (n, m) is dropped. This step is made until no edges remain in the graph. Since there are many similarities between the normative conflict graph and the argument systems presented in [Dung, 1995], the authors transform the normative conflict graph to an argument system, in which the nodes are arguments and the edges are attacks between arguments. The second heuristic selects the norms belonging to a maximal (with respect to number of norms) conflict-free set and drop all other norms. The idea of the third heuristic is based on the concept that a preferred extension of an argumentation system is a maximal (with respect to set size) admissible set of arguments, and drop all other norms. An admissible set of arguments is a set in which for each argument A that attacks B, there is an argument C that attacks A.

4.5 Compound activities [García-Camino et al., 2007]

In [García-Camino et al., 2007; Gaertner et al., 2007; Vasconcelos et al., 2012], obligations, prohibitions and permissions associated with an agent are called normative positions. Such approaches are based on the propagation of normative positions. The work in [García-Camino et al., 2007] focuses on the resolution of *direct* and *indirect* normative conflicts at *design time* but the methods for detecting conflicts are not presented. It deals with compound activities, i.e., activities that have subordinate activities. Each activity is a set of actions. In this context, the authors consider that conflicts may occur between normative positions of activities and sub-activities. The algorithm presented avoids conflicts by ordering the normative positions according to three criteria: chronological (*lex posterior*), speciality (*lex specialis*), and salience (relevance of the norm). Only after the resolution of normative conflicts, the resulting conflict-free set of normative positions is propagated to the subordinate activities. In some cases, more than one criterion must be applied in sequence to avoid the occurrence of conflicts and all the criteria involved must be totally ordered. In this approach, the specialization criterion corresponds to the hierarchical dependence between an activity and its subordinate activities.

The authors consider that two normative positions are in conflict if one is a prohibition and the other is an obligation or permission regulating the same action. Thus, the enabling time and the salience of the norms are not considered. Deontic conflicts may occur in compound activities. Two normative positions from different activities states are in conflict if one is a prohibition, the other is a permission or an obligation regulating the same action, and one of them is associated to a sub-state of the other.

The norm is represented as $\delta(a_e, s, t)$, where δ is a deontic concept from the set $\{$*obligation*,*permission*,*prohibition*$\}$; a is the action being regulated; e is an agent identifier; s is a constant (salience criterion) establishing the relevance of the norm; t is a timestamp that enables the norm (activation condition). The norm components are propositions.

4.6 Institutional facts [Li, 2014]

The work described in [Li, 2013; Li, 2014] (introduced in Section 3.8) deals with normative conflicts between independent institutions that are under the same governance scope. It can resolve *direct* normative conflicts at *design time*.

Conflicts may occur, for example, when something is permitted in one institution and it is not permitted in another one at the same time. The strategy adopted to resolve conflicts is based on inductive learning. The method consists of revising a logic program that represents a formal model containing the rules of a specific normative system. In this approach, a precedence order is established among the institutions. The revision method is applied to the norms of the less important institution of the conflicting pair of institutions. The approach is very similar to *lex superior*, however, in this work the organization with lower precedence is not overridden but changed to be consistent with the organization with higher precedence.

For instance, suppose that there is a conflict between an institution X and an institution Y and the institution X has higher precedence than institution Y. Here, the resolution method will revise institution Y to be consistent with institution X, and then compute all the possible changes that can be applied to institution Y.

The revisions are computed by using inductive logic programming (ILP), a symbolic machine learning technique which is capable of generating revisions to an existing theory in order to satisfy some properties, which are established according to the domain of the learning task. Conflicting traces are undesirable properties that are encoded as negative examples. The learning objective is to remove undesirable properties. ILP is able to generate several revisions which can resolve the conflict among the institutions. The resolution mechanism chooses the revision that requires the minimum of changes in relation to the original institution. This mechanism is correct (the generated solutions guarantee the removal of the conflicts) and complete (all possible solutions are generated).

4.7 Conversation policies [Kagal and Finin, 2007]

The work presented in [Kagal and Finin, 2005; Kagal and Finin, 2007] argues that agent communication may be facilitated by using conversation specifications and policies. The work in [Kagal and Finin, 2007] details the work presented in [Kagal and Finin, 2005] to resolve *direct* conflicts at *runtime*. The detection process is not described.

Conversation specifications establish the order in which speech acts [Searle, 1969] can be performed in a conversation. Conversation policies apply restrictions over different aspects of a conversation, such as the content of the messages, the attributes of the sender or the recipient, or any other context during a conversation. The authors present a policy framework that facilitates the communication among agents that is independent of the syntax and semantics of the communication language. Since the use of ontologies establishes the speech acts, high level policies may be represented regardless of the communication language being used.

A norm is represented as a tuple $D(e, a, ac, ec)$ or $D(e, a, ac)$, where D is either a deontic concept from the set $\{obligation, permission, prohibition\}$ or a dispensation, which means the waiver from an obligation; a is either a communicative act or a combination of speech acts including AND (logical conjunction), OR (logical disjunction) and XOR (exclusive disjunction). The speech acts allow parameterization; ac is the condition that activates a norm; ec is the condition that deactivates the norm. The fields ac and ec are constraints about the conditions in which this norm is applicable and can refer to either a time validity or a state. Obligations and dispensations have an extra field called "*obligedTo*", which represents to whom the agent is obliged. In addition, obligations and prohibitions may be associated with a field that describes sanctions to the violation of the norm. When the representation of the norm omits the field ec, it is assumed that there is no expiration condition to the norm.

Permissions and prohibitions are used to describe positive/negative authorizations whereas obligations and dispensations describe positive/negative responsibilities. Normative conflicts can occur between prohibitions and obligations, prohibitions and permissions, obligations and dispensations. In order to resolve possible conflicts, each norm is associated with meta-policies. In a situation of conflict between specifications and policies, conversation policies always override the specifications. The resolution of deontic conflicts among conversation policies makes use of meta-policies which states priority over either certain policies (based on the issuing authority) or the preferred modality, i.e., positive policies override negative policies or vice versa.

4.8 The BOID Architecture

The authors in [Broersen et al., 2001b; Broersen et al., 2001a] present the BOID logic and the BOID architecture, respectively. The main idea of these works is to build a model that extends the BDI architecture with obligations and can resolve *direct* conflicts at *runtime*. In this model conflicts may occur between four attitudes: Beliefs, Obligations, Intentions and Desires. The conflict resolution strategy is an order of overruling according to the agent type (realistic, simple-minded, selfish, social). A realistic agent always prioritizes its beliefs, i.e., beliefs override obligations, intentions or desires; in a selfish agent desires override obligations; in a social agent obligations override desires; in a single-minded or stable agent, obligations and desires override intentions. For other kinds of conflicts extensions are constructed and one is selected. Thus, the BOID logic is parametrized and the input parameter establishes the order of the derivation steps from the different types of attitudes according to the agent's type.

Conflicts may occur among informational and motivational attitudes. The authors distinguish internal conflicts and external conflicts. The former refer to conflicts involving the same attitude: (i) Conflicts among Beliefs; (ii) Conflicts among Obligations; (iii) Conflicts among Intentions; and (iv) Conflicts among Desires. External conflicts refer to conflicts that occur between different attitudes: (i) Belief and Obligation; (ii) Belief and Intention; (iii) Belief and Desire; (iv) Obligation and Intention; (v) Obligation and Desire; (vi) Intention and Desire; (vii) Belief and Obligation and Intention; (viii) Belief and Obligation and Desire; (ix) Belief and Intention and Desire; (x) Obligation and Intention and Desire; (xi) Belief and Obligation and Intention and Desire. In addition, while other approaches define conflicts as minimal sets, this approach states that the whole set of beliefs, obligations, intentions and desires must be considered in order to resolve a normative conflict. The agents should analyze the effects of the performance of an action before performing it. For this reason, the BOID architecture constructs complete extensions before selecting one extension. However, a limitation of this proposal is that only obligations can be represented. In addition, this work considers that the norms are provided to agents at *design time*.

In [Broersen et al., 2001a] a norm is specified as $x(Index, A \rightarrow W)$, where x is an attitude from the set $\{$belief, obligation, intention, desire$\}$; *Index* is a variable that indicates the particular example or situation that is modeled; A is a formula of propositional logic (for unconditional attitudes, A has the value "true"); and W is a formula of propositional logic, which represents the behaviour being regulated.

4.9 The NoA Architecture

In the NoA architecture [Kollingbaum and Norman, 2004] (introduced in Section 3.10) the resolution of *direct* and *indirect* normative conflicts is performed at *runtime*. The authors present an instantiation graph, which represents an action/state declaration as a hierarchy of all possible forms of partial instantiation. This graph is a useful tool to verify the possible values that the variables of the norm can assume. The leaf nodes of the instantiation graph are fully instantiated and represent the actual actions. When a new norm is adopted and can be applied to a node of the graph, this norm is propagated to all child nodes, which inherit this norm. In addition, the authors assume the performance of an action is permitted if there is no active norm prohibiting it.

The authors present different means to deal with conflicts according to their types and define some special cases of conflicts. Conflicts may occur between an implicit permission and an explicit prohibition. Any norm explicitly adopted overrides the implicit permission until the explicit norm expires.

Conflicts between explicit prohibitions and explicit permissions that regulate the same activity can be solved through a ranking of norms. This prioritization is based on the time of the adoption of the norms, the most recent norm overrides the other (according to the classic concept *lex posterior*). The overriding mechanism is only valid if the expiration condition of the prioritized norm does not hold. The same strategy is adopted to resolve *indirect* conflicts.

Another kind of conflict may occur between prohibitions and permissions that refer to different partially instantiated actions/states. One case of this kind of conflict is called the parent-child conflict. It occurs when two norms are applied to different nodes of the instantiation graph and one node is within the scope of the other node. In this case, a leaf node of the graph would inherit the two different norms. The strategy to solve this kind of conflict consists of choosing the norm at the child node to overrule the norm at the parent node.

The last kind of conflict described in this paper is called the multiple-inheritance problem. In this case, the two norms regulate partial instantiated actions/states that have intersecting scopes. To resolve this conflict, the authors determine that the most recent norm overrides the other, according to the *lex posterior* concept. The authors comment that this kind of conflict could be resolved by taking into account the social position and power of the norm issuer, but this strategy was not explored.

Additionally, this work defines three types of consistency: (i) strong consistency: an obligation has strong consistency if it is consistent for all plans; (ii) weak consistency: an obligation has weak consistency if it is consistent for at least one possible plan; and (iii) strong inconsistency: an obligation is inconsistent if it is inconsistent for all plans.

The approaches in [Kollingbaum et al., 2006; Kollingbaum et al., 2007] state that normative conflicts are very common in open virtual organizations (VOs) and show how the NoA model can inform a renegotiation of contracts and norms. This work describes a new approach to solve normative conflicts. When a conflict is detected, the interested parties renegotiate their contracts, changing the conflicting norms. In this context, prohibitions and obligations can be "relaxed" in order to provide a set of consistent norms. The main goal of this renegotiation is to create new possible options for actions for a contracting agent or to extend the existing options. Thus, the NoA architecture informs the agent about norm consistency and provides strategies to solve the normative conflicts.

In [Kollingbaum et al., 2006], three possible options of renegotiation are described: (i) Extending the scope of influence: changing an obligation so that it does not conflict with any prohibitions; (ii) Reducing the scope of influence: changing a prohibition so that it does not conflict with the obligations; and (iii) Overriding prohibitions: introducing new permissions that override temporarily the prohibitions in order to "allow" the performance of additional actions for the fulfilment of obligations. The authority also may revoke the prohibiting norms in order to solve conflicts. Likewise, for eliminating conflicts, the agents may reduce the scope of influence of one of the conflicting norms in order to become either more general or more specialized. The idea is to achieve, at least, the weak consistency level.

4.10 Unification and constraints [Vasconcelos et al., 2009]

The work reported in [Vasconcelos et al., 2009] (introduced in Section 3.2) presents a means to resolve *direct* and *indirect* normative conflicts at *runtime*.

Constraints are used to refine the scope of influence of norms on action. The resolution strategy consists in a curtailment of the conflicting norms by adding constraints to their scope of influence. Given two conflicting norms, policies determine which norm to curtail and the classic concept of *lex superior* can be adopted.

The authors present algorithms for the adoption and removal of norms in order to maintain conflict freedom. When a norm is added to a conflict-free set of norms, an algorithm analyzes whether a curtailment is needed. When this norm is removed, the curtailments it caused are undone. The algorithm to curtail a norm is also presented. The algorithms to adopt/remove a norm preserving conflict-freedom have exponential complexity in the worst case, and have been argued as being correct and always terminating. A limitation of [Vasconcelos et al., 2009] is that the norms can only regulate actions and that this work does not cope with conditional norms.

Therefore, the conflict resolution mechanism manipulates the constraints of norms to avoid overlapping values of variables, i.e., applying the called curtailment of variables (norms).

4.11 Pre-emptive approach [Vasconcelos and Norman, 2009]

The work described in [Vasconcelos and Norman, 2009] (introduced in Section 3.3) is capable of resolving *direct* normative conflicts in contracts at *runtime*. The resolution method deals with these conflicts by adding constraints to the conflicting norms thus reducing or decreasing the scope of the norm. The authors state that they have not adopted the same strategy of [Vasconcelos et al., 2009] for resolving conflicts due to the computational complexity required for that method.

The authors suggest a preemptive approach to deal with the resolution of normative conflicts, as follows: the rules of a contract create and remove norms. Rules are analyzed before their adoption, and when potential conflicting norms are found, the curtailment mechanism is invoked. The norm curtailment avoids future conflicts.

Thus, the authors suggest “amendments” to the clauses of the contract in order to resolve conflicts. An algorithm for preemptive contract formation is presented. This approach can be said to be preemptive, due to the fact that the normative conflicts are considered before the enactment of the contract and hence before any normative conflict actually arises.

4.12 The Normative Structure [Gaertner et al., 2007]

The NS model [Gaertner et al., 2007] extended in [Vasconcelos et al., 2012] (introduced in Section 3.9) is a distributed algorithm for online conflict resolution between normative positions. Once a *direct* conflict is detected, the resolution method annotates the conflicting normative positions and the unifier. These annotations will determine which values the norm cannot assume in order to avoid the conflict, curtailing the influence of the norms. The authors curtail prohibitions but state that the same mechanism can be applied to curtail obligations. The decision to curtail a norm that has a specific deontic concept is dependent on the requirements of the system addressed. This method is applied when a new norm is adopted, and an algorithm to adopt a new norm preserving conflict-freedom is presented.

The authors state that the resolution mechanism is correct and always terminates; for a given normative position and a normative state, it provides a new normative state in which all norms have annotations with the values that cannot be assumed by the norms in order to avoid the occurrence conflicts. In addition, the algorithm proposed has linear complexity. The resolution method presented in [Vasconcelos et al., 2012] is more compact than the method presented in [Gaertner et al., 2007]. While the method presented in [Gaertner et al., 2007] updates the input set of normative positions, the algorithm described in [Vasconcelos et al., 2012] creates commands to add or remove the normative positions which, when applied to the input set of normative positions, will ensure conflict-freedom.

4.13 n-BDI

In [Criado et al., 2010c], the normative multi-context Graded BDI architecture (n-BDI) [Criado et al., 2010a; Criado et al., 2010b] is refined by including a more elaborate notion of norm and norm reasoning. An agent is represented by a set of interconnected contexts. This approach can resolve *direct* conflicts at *runtime* and employs coherence theory as a criterion for removing conflicting propositions (both mental and normative). Thus, in each context, a subset of maximal coherence is chosen in order to resolve the conflicts. The resolution algorithm proposed considers the agents' goals and beliefs and the kind of conflict, namely, between a permission and a prohibition or obligation, and between a prohibition and an obligation.

The authors specify an abstract norm (*Na*) as a tuple $\langle D, A, E, C, S, R \rangle$, where D is a deontic concept from the set $\{obligation, permission, prohibition\}$; A is the activation condition of the norm; E is the expiration condition of the norm; C is a logic formula that represents the state or action being regulated; S is a set of sanctions associated with the norm; and R is a set of rewards associated with the norm.

The norm instance (*Ni*) is formalised as $Ni = \langle D, C' \rangle$, where D is the deontic concept; C' is the logic formula that represents the state/action being regulated. For simplicity, the authors omit the norm expiration and activation conditions, and the sanctions and rewards in the representation of a norm instance. In this work, the norm components are propositions.

4.14 Reiter's Default Logic approach [Giannikis and Daskalopulu, 2011]

The approach presented in [Giannikis and Daskalopulu, 2009; Giannikis and Daskalopulu, 2011] (introduced in Section 3.11) describes techniques to resolve *direct* and *indirect* normative conflicts that arise for agents engaging in electronic contracting at *runtime*.

The normative conflict resolution is based on applying priorities and generating preferred extensions. These priorities may be specified dynamically either by considering different assumptions or by specifying domain-dependent criteria. According to the proposal, the priority may be established via the three classical principles found in literature: *lex posterior*, *lex superior*, *lex specialis*. To represent the priorities dynamically, the priorities are specified as default rules.

4.15 Commitment conflicts [Günay and Yolum, 2013b]

The work described in [Günay and Yolum, 2013b] (presented in Section 3.14) states that in an organization conflicts may occur among roles or within a specific role. The obligations and prohibitions associated with the roles are represented through commitments. The authors present strategies to deal with *direct* conflicts among commitments at *runtime*.

A role specification is inconsistent if there is a conflict among its commitments. Such conflicts may be resolved by modifying the antecedents of the commitments or reconsidering the commitments that must be part of the role. When agents assume more than one role at the same time, conflicts may arise among these roles. In this case, an override relation between the commitments must be specified. However, even though the conflicts are avoided at *design time*, when an agent does not have the resources needed to perform a certain behaviour established by the commitments, conflicts may arise at *runtime*.

A commitment set is feasible if it is possible to fulfill all its commitments. The feasibility can be achieved by delegation, i.e., agents playing certain roles may have the power to delegate some of their commitments to other ones. The delegation will resolve a conflict only if the deputy or deputies have enough resources and are able to fulfill the commitments. The authors also suggest the use of new commitments when a violation occurs. When an agent violates a commitment, a new commitment (with sanctions) may be introduced to repair the consequences of the violation and bring the system back into a proper state.

4.16 Neural-symbolic framework [Boella et al., 2012]

In [Boella et al., 2012] the authors present a neural-symbolic framework for reasoning and learning about norms. It can resolve *direct* conflicts at *runtime*. This approach also adopts the notion of the Input/Output (I/O) Logic [Makinson and van der Torre, 2000], a symbolic formalism to represent and reason about norms and produce outputs from the inputs. The neural-symbolic paradigm [Garcez et al., 2012] provides means to embed symbolic formalism into neural networks.

Norms are represented using I/O Logic rules. Therefore, a norm is a pair (A, D (B)), where A is the antecedent, i.e., the condition that activates the norm; B is the consequent; and D is the deontic operator that indicates either obligation or permission. A prohibition is represented as an obligation of a negative literal. A and B are sets of literals that can be in the conjunctive or disjunctive form and represent actions/states. For instance, the norm (getFine, O (payFine)) means: if you are given a fine, you ought to pay it. This work assumes that something is permitted if not explicitly forbidden.

The authors present the concepts of: (i) dilemmas, which are situations where there are two contradictory obligations, i.e, obligations that cannot be fulfilled together; (ii) contrary to duties, which establish that new obligations may be introduced when a norm violation occurs; and (iii) exceptions: particular situations in which a norm should be followed instead of another.

The strategy to resolve conflicts consists in establishing priorities between norms that give a partial ordering between them. This strategy may be adopted in the case of exceptions, for example. To encode the priority among the norms, the authors use negation as failure ($\sim$). Given two norms $\mathrm{R}_1 = (\mathrm{A}_1 \wedge \mathrm{A}_3, \mathrm{O}\ (\beta_1))$ and $\mathrm{R}_2 = (\mathrm{A}_2 \wedge \mathrm{A}_3, \mathrm{O}\ (\beta_2))$ and a priority relation $\mathrm{R}_1 \succ \mathrm{R}_2$ between them, the priority relation is encoded by modifying the antecedent of the norm with lower priority. Thus, the negation as failure of the literals in the antecedent of the norm with higher priority, which does not appear in the antecedent of the lower priority norm, is included in the antecedent of the norm with the lower priority.

This approach transforms the norms and the priority relations into a neural network. The neural network built is trained with instances that contain normal situations and situations in which contrary-to-duty is applied. After that, the neural network is able to learn the priority based orderings that regulate the contrary to duties.

4.17 ANA

The work in [dos Santos Neto et al., 2012] (introduced in Section 3.12) presents a mechanism to resolve *direct* normative conflicts between prohibitions and obligations at *runtime*. To resolve a normative conflict, the algorithm performs two calculations. First, the motivation for fulfilling the first norm plus the motivation for violating the second is calculated. After that, the motivation for violating the first norm plus the motivation to fulfilling the second norm is calculated. When the first calculation results in a greater value than the second, the agent selects the first norm to fulfill. If the opposite occurs, the second norm is selected to be fulfilled. If the two calculations have the same motivation, the agent can choose either norm to fulfill.

To evaluate the fulfilment of a norm, the agent sums the deontic influence on the state regulated by the norm and the motivation for receiving the rewards. The motivation to violate a norm is the motivation for receiving the punishments. The motivation for receiving the rewards or punishments is the sum of the motivation of the agent's goals that specify the states regulated by the rewards or punishments. If the norm is an obligation, and the motivation of the agent to achieve such a goal is positive, then the deontic influence is equal to the motivation of this goal. However, if the motivation to achieve such a goal is negative, then the norm influences the agent negatively. If the norm is a prohibition and the motivation to achieve the goal is negative, the norm influences the agent positively. Otherwise, if the agent's motivation to achieve such goal is positive, the norm influences the agent negatively.

4.18 The NBDI Architecture

The NBDI architecture [dos Santos Neto et al., 2013] (introduced in Section 3.13) has functions for reasoning about norms and has a component responsible for solving *direct* normative conflicts at *runtime*. When a conflict is detected, the two conflicting norms are evaluated based on their influence over the agent's desires/intentions. There is a method that calculates the contribution of a norm: the algorithm verifies if the state regulated by the norm is a state that the agent has the desire or intention to achieve. In affirmative cases, if the norm is an obligation, then the contribution receives a positive value (based on the priority of the desire). On the other hand, if the norm is a prohibition, then the contribution receives a negative value, since it disturbs the achievement of the desired or intended state. In any other case, the contribution is zero. After that, the algorithm analyzes the rewards and punishments associated with the conflicting norms. A reward can add a positive value to the norm contribution and the punishment can add a negative value to the norm contribution.

Then, the contribution coming from the fulfilment of the first norm plus the contribution coming from the violation of the second norm (and vice-versa) are calculated. The conflict is solved by selecting the case that results in the highest total contribution. The norm related to the highest total contribution is prioritized over the other.

5 Comparative Analysis

In this section, we present two comparative tables about the proposals described in the previous sections. All the papers found in the literature are able to deal with *direct* conflicts. Additionally, all the mechanisms to detect conflicts analyze norms in pairs.

Table 7 summarizes the approaches to detect and resolve normative conflicts. In order to reduce the length, we have chosen to list only the most significant papers of each approach/section. Table 7 is organized as follows: each row of the table indicates a paper, identified by a reference in column 1; the *Deontic concepts* column indicates the deontic concepts considered by the approach, namely, O for obligation, P for permission and F for prohibition; the *Kind of conflict* column refers to the kinds of conflicts that the work is able to deal with, i.e., conflicts between which deontic modalities can be detected by the proposal. This column also describes conflicts involving legal power, dispensations, and the attitudes considered by the BOID architecture [Broersen et al., 2001a; Broersen et al., 2001b] (Belief – B; Desire – D; Intention – I). The character p stands for the behaviour being regulated; the *Detection guarantees* column indicates whether the detection mechanism has guarantees such as correctness and completeness, or not; the *Resolution guarantees* column indicates whether the resolution mechanism has guarantees such as correctness and completeness, or not; and the *Available Mechanism* columns state whether the paper presents computational mechanisms to detect/resolve normative conflicts (an implementation or an algorithm). Note that in Table 7 some of the approaches have implementations but no described algorithm.

Table 7 Approaches that deal with normative conflicts

			Conflict Detection				Conflict Resolution			
	Deontic Concepts	Kind of conflict	Detect	Detection guarantees	Available Mechanism		Resolve	Resolution guarantees	Available Mechanism	
					Algorithm	Implementation			Algorithm	Implementation
[Aphale et al., 2013]	O, P, F	O x F, P x F	✓		✓	✓	✓		✓	✓

[Boella et al., 2012]	O, P	O(p) x O(¬p)
		P(p) x O(¬p)
			✓				✓		✓	

[Broersen et al., 2001a]	O	B x B, O x O, I x I, D x D B x O, B x I, B x D, O x I, O x D, I x D, B x O x I, B x O x D, B x I x D, O x I x D B x O x I x D	✓			✓	✓			✓
[Cholvy and Cuppens, 1995]	O, P, F	Or$_i$p x Or$_j$¬p, Or$_i$p x Fr$_j$p, Or$_i$p x Pr$_j$¬p	✓				✓			
[Cholvy and Cuppens, 1998]	O, P, F	O x F, P x F	✓				✓			
[Criado et al., 2010c]	O, P, F	O x F, P x F	✓			✓	✓			✓
[dos Santos Neto et al., 2012]	O, F	O x F	✓			✓	✓			✓
[dos Santos Neto et al., 2013]	O, F	O x F	✓		✓	✓	✓		✓	✓
[Fenech et al., 2008]	O, P, F	O x F, P x F, O x O, O x P	✓	✓						
[Gaertner et al., 2007]	O, P, F	O x F, P x F	✓	✓	✓		✓	✓	✓	
[García-Camino et al., 2007]	O, P, F	O x F, P x F	✓				✓		✓	
[Garcia et al., 2013]	O, P, F	O x F, P x F	✓			✓				
[Giannikis and Daskalopulu, 2011]	O, P, F	O x F, P x F, O x O, power x F, ∼power x O	✓				✓			
[Günay and Yolum, 2013b]	O, F	O x F	✓				✓			
[Kagal and Finin, 2007]	O, P, F	P x F, O x F, O x dispensation	✓			✓	✓			✓
[Kollingbaum et al., 2007]	O, P, F	O x F, P x F	✓			✓	✓			✓
[Li, 2014]	O, P, F	O x F, P x F	✓		✓			✓		
[Oren et al., 2008]	O, P, F	O x F, P x F, O x O	✓		✓		✓		✓	
[Vasconcelos et al., 2009]	O, P, F	O x F, P x F	✓	✓	✓		✓	✓	✓	
[Vasconcelos and Norman, 2009]	O, P, F	O x F, P x F	✓	✓	✓		✓		✓	
[Zahn, 2015] [Zahn and da Silva, 2014]	O, P, F	O x F, P x F, O x O, O x P	✓		✓	✓				

Table 8 presents which relationships are analyzed in order to detect normative conflicts in the different approaches. It is organized as follows: each row of the table refers to a paper, identified by column 1; the *Side-effects* column refers to the ability of the approach to analyze actions' side-effects; the *Composition* column states whether the paper considers relationships of composition among actions; the *Orthogonality* column indicates whether the approach can detect conflicts among mutually exclusive actions; the *Precondition* column indicates whether the work considers relationships of dependency among actions, i.e., if an action is a precondition of another one; the *Refinement* column refers to the relationship of specialization among actions; and the *Play* column states if the detection method verifies which roles an entity may play.

Since some kinds of relationships are presented only in [da Silva et al., 2015; da Silva and Zahn, 2014; Zahn, 2015; Zahn and da Silva, 2014] and [Vasconcelos et al., 2009] they have not been included in Table 8.

Table 8 A comparative analysis about the relationships detected

	Action relationships					Entity relationships
	Side-effects	Composition	Orthogonality	Precondition	Refinement	Play
[Aphale et al., 2013]	✓			✓		
[Cholvy and Cuppens, 1995]						✓
[Cholvy and Cuppens, 1998]						✓
[da Silva et al., 2015]		✓	✓	✓	✓	✓
[da Silva and Zahn, 2014]		✓			✓	✓
[Fenech et al., 2009]			✓			
[Fenech et al., 2008]			✓			
[García-Camino et al., 2007]		✓				
[Giannikis and Daskalopulu, 2009]			✓			
[Giannikis and Daskalopulu, 2011]			✓			
[Kollingbaum and Norman, 2004]	✓					
[Kollingbaum and Norman, 2006]	✓					
[Kollingbaum et al., 2006]	✓					
[Kollingbaum et al., 2007]	✓					
[Oren et al., 2008]			✓			
[Şensoy et al., 2012]	✓			✓		
[Vasconcelos et al., 2009]	✓	✓				
[Zahn, 2015]		✓	✓	✓	✓	✓
[Zahn and da Silva, 2014]		✓	✓	✓	✓	✓

Table 9 exhibits the computational cost of the detection/ resolution algorithms described in each approach. Some approaches do not mention anything about the computational costs of their algorithms and do not present the algorithms explicitly. The N/A column represents those approaches. Although Sections 3.2–3.4, Section 3.9, and Section 4.1 present an analysis of the computational complexity we could not confirm their analysis since those approaches do not present sub-procedures used by the main algorithms. The areas in Section 3.8 and Section 4.11 are not listed in Table 9 because although their algorithms are not self-contained; that is, they rely on sub-algorithms/procedures which are not explained, or they do not present their computational complexity. We have carried out the analysis of the computational complexity of the remaining approaches.

Table 9 Computational cost of the different approaches

	Linear $O(n)$	Exponential $O(c^n)$	Polynomial $O(n^c)$	Decidable but intractable	NEXPTIME-complete	PSPACE-complete	N/A
3.1 Conflict Checker			✓				
3.2 Unification and constraints [Vasconcelos et al., 2009]	✓						
3.3 Preemptive approach [Vasconcelos and Norman, 2009]	✓						
3.4 OWL-POLAR				✓	✓		
3.5 Deontic Logic approach [Cholvy and Cuppens, 1995]							✓
3.6 FUSION Logic [Cholvy and Cuppens, 1998]							✓
3.7 CL Contracts [Fenech et al., 2008]							✓
3.9 The Normative Structure [Gaertner et al., 2007]	✓						
3.10 The NoA Architecture							✓
3.11 Reiter's Default Logic approach [Giannikis and Daskalopulu, 2011]							✓
3.12 ANA							✓
3.13 The NBDI Architecture			✓				
3.14 Commitment conflicts [Günay and Yolum, 2013b]							✓
3.15 Normative Conflict Graph [Oren et al., 2008]			✓				
3.16 ROMAS CASE tool							✓
4.1 OWL-POLAR						✓	
4.2 Deontic Logic approach [Cholvy and Cuppens, 1995]							✓
4.3 FUSION Logic [Cholvy and Cuppens, 1998]							✓
4.4 Normative Conflict Graph [Oren et al., 2008]	✓						
4.5 Compound activities [García-Camino et al., 2007]			✓				
4.6 Institutional facts [Li, 2014]							✓
4.7 Conversation policies [Kagal and Finin, 2007]							✓
4.8 The BOID Architecture							✓
4.9 The NoA Architecture							✓
4.10 Unification and constraints [Vasconcelos et al., 2009]		✓					
4.12 The Normative Structure [Gaertner et al., 2007]	✓						
4.13 n-BDI							✓
4.14 Reiter's Default Logic approach [Giannikis and Daskalopulu, 2011]							✓
4.15 Commitment conflicts [Günay and Yolum, 2013b]							✓
4.16 Neural-symbolic framework [Boella et al., 2012]							✓
4.17 ANA							✓
4.18 The NBDI Architecture	✓						

6 Conclusions

Multi-agent systems are composed of software agents, which are autonomous and possibly heterogeneous entities. Due to the autonomy and heterogeneity of the software agents, predicting the overall behaviour of a MAS is a challenging issue. Therefore norms have been used to guide and model the behaviour of software agents, without restricting their autonomy. These norms are commonly associated with deontic concepts, such as obligations, permissions, and prohibitions. However, in a system governed by multiple norms, normative conflicts may arise, i.e., sometimes the fulfilment of a norm automatically violates another one. For this reason, the agents or the MAS must have mechanisms to detect and resolve conflicts among norms.

In this paper, we have surveyed different approaches that deal with normative conflicts, described them, and present a comparative analysis of various aspects of these approaches. The main contribution of our work is the organization of knowledge about normative conflict detection and resolution. This allows us to contrast and compare disparate approaches, highlighting their representation and computational issues. Our research can be useful to different audiences or readers, such as (but not limited to):

- *MASs practitioners* – These have pragmatic concerns related to the design, analysis and implementation of practical MASs, including knowledge engineering, requirement analysis, and verification, and are interested in ready-to-use tools and implemented mechanisms. For such audiences, we particularly recommend, for instance, the approaches surveyed in Sections 3.1 (Conflict Checker), 3.4 and 4.1 (OWL-POLAR), 3.10 (The NoA Architecture), 3.16 (ROMAS CASE tool) since they have associated (implemented) mechanisms and descriptions of algorithms.
- *Those interested in logic-theoretical, legal or philosophical aspects* (e.g., logicians, philosophers, policy-makers, lawyers, and so on). For such audiences, we recommend approaches related to knowledge representation and information modelling as well as deontic reasoning, such as the approaches described in Sections 3.5 and 4.2 (Deontic Logic approach), 3.6 and 4.3 (FUSION Logic), 3.11 and 4.14 (Reiter's Default Logic approach), for instance.

The computational cost required by the methods of detection and conflict resolution is an important factor when taking an approach to deal with conflicts within a MAS. For this reason, we analyzed the complexity of the different approaches but could not analyze the approaches that did not present an algorithm explicitly or did not detail their sub-procedures. The majority of the detection/resolution approaches analyzed have linear or polynomial complexities.

There are several approaches dedicated to the detection of conflicts while others are dedicated to both detection and resolution of conflicts. We analyzed these approaches in order to identify possible combinations between them. First of all, we focused on finding the proposals that are incompatible with all other approaches.

- The BOID architecture (Section 4.8) is incompatible with all other approaches since BOID detects and solves conflicts among beliefs, obligations, intentions and desires while the others focus on conflicts between prohibitions, permissions and obligations.
- The resolution methods described in ANA [dos Santos Neto et al., 2012] (Section 4.17) and in the NBDI Architecture [Broersen et al., 2001a] (Section 4.8) find the motivation of an agent to comply with a norm and the motivation to violate the norm in order to decide which norm to prioritize. Since the representation of motivations is a specific

attribute of those methods, we conclude that they cannot be combined with other approaches.
- Institutional facts [Li, 2014] (Section 3.8 and 4.6) are used in the analysis of conflicts between norms of different institutions, and the associated resolution strategy prioritizes the norm belonging to the most important institution. Since none of the other approaches deal with conflicts among institutions, we consider that the resolution strategy described in Sections 3.8 and 4.6 cannot be combined with other approaches.
- We can also infer that some approaches cannot be combined with others due to their unique norm representation. This is the case of Normative Structures (Section 3.9 and Section 4.12), in which norms regulate illocutions; the Neural-symbolic framework [Boella et al., 2012] (Section 4.16) described with I/O Logic rules and which cannot represent prohibitions; the proposal of [Günay and Yolum, 2013b] (Section 3.14 and Section 4.15) dealing with conflicts among commitments (which are not commonly represented); and the approach adopting Reiter's Default Logic (Section 3.11 and Section 4.14), which deals with conflicts among mutual legal relations involving two agents.

An analysis of the possible combinations taking into account the strategy adopted by the resolution approach and the norm definition of the detection approaches is as follows:

- OWL-POLAR [Şensoy et al., 2012] (Section 4.1), Compound activities [García-Camino et al., 2007] (Section 4.5), Unification and constraints [Vasconcelos et al., 2009] (Section 4.10) present resolution methods that are based on prioritization adopting the *lex specialis* strategy. For this reason, we consider that they can be combined with the detection approaches that capture relationships of specialization among entities or among actions, such as, Conflict Checker (Section 3.1) and OWL-POLAR (Section 3.4), or with detection approaches that can identify a specialization relationship based on the scope of influence of the norm, such as, the NoA Architecture (Section 3.10), Unification and constraints [Vasconcelos et al., 2009] (Section 3.2) and the Pre-emptive approach [Vasconcelos and Norman, 2009] (Section 3.3).
- OWL-POLAR [Şensoy et al., 2012] (Section 4.1), Compound activities [García-Camino et al., 2007] (Section 4.5), Conversation policies [Kagal and Finin, 2007] (Section 4.7) and the NoA Architecture [Kollingbaum, 2005] (Section 4.9) present resolution methods based on prioritization according to the *lex posterior* strategy. We therefore suggest that they can be combined with the detection approaches establishing the time when the norm was created or the time when the norm becomes active, such as, Conflict Checker (Section 3.1), Unification and constraints [Vasconcelos et al., 2009] (Section 3.2), Pre-emptive approach [Vasconcelos and Norman, 2009] (Section 3.3), OWL-POLAR (Section 3.4).
- OWL-POLAR [Şensoy et al., 2012] (Section 4.1), Normative Conflict Graph [Oren et al., 2008] (Section 4.4), Conversation policies [Kagal and Finin, 2007] (Section 4.7), and Unification and constraints [Vasconcelos et al., 2009] (Section 4.10) adopt resolution methods that are based on the *lex superior* strategy. They therefore can be combined with detection approaches that establish the entity that created/imposed the norm, such as, Unification and constraints [Vasconcelos et al., 2009] (Section 3.2) and Normative Conflict Graph [Oren et al., 2008] (Section 3.15).
- The Normative Conflict Graph [Oren et al., 2008] (Section 4.4) and Conversation policies [Kagal and Finin, 2007] (Section 4.7) can also resolve conflicts by establishing an order of prioritization according to the modality of the norm, and thus can be combined with all detection approaches independently of the norm representation.

– The Normative Conflict Graph [Oren et al., 2008] (Section 4.4) can also solve conflicts by dropping norms randomly or according to an argumentation heuristic. Thus this approach can be combined with all detection approaches since this strategy does not depend on the norm representation.
– According to the Deontic Logic approach [Cholvy and Cuppens, 1995] (Section 4.2) a normative conflict is solved by establishing an order of prioritization between the roles mentioned in the conflicting norms. We thus agree that this approach can be combined with all detection approaches which include a role in the norm specification, such as the Conflict Checker [da Silva and Zahn, 2014] (Section 3.1), Unification and constraints [Vasconcelos et al., 2009], Pre-emptive approach [Vasconcelos and Norman, 2009], OWL-POLAR [Şensoy et al., 2012] (Section 3.4), the Deontic Logic approach [Cholvy and Cuppens, 1995] (Section 3.5), the NoA Architecture [Kollingbaum, 2005] (Section 3.10), and the ROMAS CASE tool [Garcia et al., 2013] (Section 3.16).
– The resolution methods described in the NoA Architecture [Kollingbaum, 2005] (Section 3.10), Unification and constraints [Vasconcelos et al., 2009] (Section 4.10), Pre-emptive approach [Vasconcelos and Norman, 2009] (Section 3.3), Compound activities [García-Camino et al., 2007] (Section 4.5), and Normative Structures [Gaertner et al., 2007] (Section 4.12) solve conflicts by extending the scope of the norm. This can be done by adding constraints to the conflicting norms. We thus infer that these approaches can be combined with the ones that explicitly represent constraints in norms such as Unification and constraints [Vasconcelos et al., 2009] (Section 3.2), Pre-emptive approach [Vasconcelos and Norman, 2009] (Section 3.3), OWL-Polar [Şensoy et al., 2012] (Section 3.4), and the NoA Architecture [Kollingbaum, 2005](Section 3.10).
– The resolution strategy described in n-BDI [Criado et al., 2010b] (Section 4.13) considers the agents' goals and beliefs, and the modalities of the conflicting norms. For this reason, we infer that this approach can be combined with all detection approaches based on the classic BDI architecture, such as, the NBDI Architecture [dos Santos Neto et al., 2013] (Section 3.13), ANA [dos Santos Neto et al., 2012] (Section 3.12) and the NoA Architecture [Kollingbaum, 2005] (Section 3.10).

There are many challenges and limitations in approaches to normative MASs. While some approaches can only deal with norms regulating atomic actions, others support the regulation of parameterized actions, a set of actions as well as states of affair (resulting from actions). All approaches can detect *direct* conflicts. On the other hand, none of the approaches presented in this paper is able to detect conflicts among multiple norms, i.e., all the approaches detect conflicts involving only two norms (BOID [Broersen et al., 2001a; Broersen et al., 2001b] identifies conflicts among attitudes, though). However, sometimes normative conflicts can only be detected when we analyze more than two norms simultaneously.

Additionally, to find all potential normative conflicts that may occur within a MAS is a task that depends on several issues, such as the analysis of the characteristics of the application domain, the investigation of relationships between the actions, agents, states and the context in which the norms are applied. Furthermore, the consequences and side-effects of performing an action must also be considered, and this is not always a trivial task. The majority of approaches that can detect and resolve *indirect* conflicts between norms are applied only within a specific architecture. We also notice that the mechanisms that identify *indirect* conflicts at *design time* require that the relationships between the entities of the system are pre-determined and established by the designer. All mechanisms require the explicit representation of the relationships by the designer.

We thus conclude that there is no single detection/resolution method that is best to detect/resolve conflicts in normative MASs. The inevitability of dealing with specific domains and the need for more practical solutions justify and motivate the proposal of several approaches. Therefore, in order to decide which strategy to adopt to deal with normative conflicts, the software engineer must take into account the purpose and characteristics of the normative MAS and pay attention to several factors, such as:

1. The norm expressiveness needed;
2. The deontic modalities considered;
3. The availability of mechanisms (ideally implemented);
4. The relationships that can be captured by the detection method;
5. Whether the strategy to detect and resolve normative conflicts has guarantees such as correctness, completeness and termination;
6. When the approach can be used, that is, if at *design time* or *runtime*;
7. The computational complexity (time and/or memory) of the strategy.

Finally, we detect an important gap in the current literature on the study of normative conflict, namely, the absence of work on *ethical* and *moral* aspects of norm conflict detection and resolution. In spite of the significant overlap between norms and moral/ethical issues, as studied in, for instance, [McNamara, 2006], [Bicchieri, 2006], and [von Wright, 1951], to name a few, such concerns feature only peripherally, if at all, in the surveyed literature. Given the current interest in *ethical autonomy* [Kirkpatrick, 2015; Vardi, 2015], especially related to technologies which will impact day-to-day activities of many people, such as autonomous vehicles, it is imperative that more attention should be devoted by the community to the topic.

References

[Aphale et al., 2013] Aphale, M., Norman, T. J., and Şensoy, M. (2013). Goal-directed policy conflict detection and prioritisation. In Aldewereld, H. and Sichman, J. S., editors, *Coordination, Organisations, Institutions and Norms in Agent Systems VIII*, volume 7756 of *Lecture Notes in Computer Science*, pages 87–104. Springer.

[Bicchieri, 2006] Bicchieri, C. (2006). *The Grammar of Society: the nature and dynamics of social norms*. Cambridge Univ. Press.

[Boella et al., 2012] Boella, G., Tosatto, S. C., Garcez, A. D., Genovese, V., Perotti, A., and van der Torre, L. (2012). Learning and reasoning about norms using neural-symbolic systems. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems-Volume 2*, pages 1023–1030. International Foundation for Autonomous Agents and Multiagent Systems.

[Boella et al., 2006] Boella, G., van der Torre, L., and Verhagen, H. (2006). Introduction to normative multiagent systems. *Computational & Mathematical Organization Theory*, 12(2-3):71–79.

[Broersen et al., 2001a] Broersen, J., Dastani, M., Hulstijn, J., Huang, Z., and van der Torre, L. (2001a). The BOID Architecture: Conflicts between Beliefs, Obligations, Intentions and Desires. In *Proceedings of the Fifth International Conference on Autonomous Agents*, AGENTS '01, pages 9–16, New York, NY, USA. ACM.

[Broersen et al., 2001b] Broersen, J., Dastani, M., and van der Torre, L. (2001b). Resolving conflicts between Beliefs, Obligations, Intentions, and Desires. In *Proceedings of the 6th European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty*, ECSQARU '01, pages 568–579, London, UK, UK. Springer-Verlag.

[Cholvy, 2006] Cholvy, L. (2006). Querying contradictory databases by taking into account their reliability and their number. In *Flexible Databases Supporting Imprecision and Uncertainty*, pages 149–168. Springer.

[Cholvy and Cuppens, 1995] Cholvy, L. and Cuppens, F. (1995). Solving normative conflicts by merging roles. In *Proceedings of the 5th International Conference on Artificial Intelligence and Law*, ICAIL '95, pages 201–209, New York, NY, USA. ACM.

[Cholvy and Cuppens, 1998] Cholvy, L. and Cuppens, F. (1998). Reasoning about norms provided by conflicting regulations. *Norms, logics and information systems: new studies in deontic logic and computer science*, pages 247–264.

[Cholvy and Garion, 2004] Cholvy, L. and Garion, C. (2004). Answering queries addressed to several databases according to a majority merging approach. *Journal of Intelligent Information Systems*, 22(2):175–201.

[Cliffe et al., 2007] Cliffe, O., De Vos, M., and Padget, J. (2007). Answer set programming for representing and reasoning about virtual institutions. In Inoue, K., Satoh, K., and Toni, F., editors, *Computational Logic in Multi-Agent Systems*, volume 4371 of *Lecture Notes in Computer Science*, pages 60–79. Springer Berlin Heidelberg.

[Criado et al., 2010a] Criado, N., Argente, E., and Botti, V. (2010a). A BDI architecture for normative decision making. In *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems: volume 1-Volume 1*, pages 1383–1384. International Foundation for Autonomous Agents and Multiagent Systems.

[Criado et al., 2010b] Criado, N., Argente, E., and Botti, V. J. (2010b). Normative deliberation in graded BDI agents. In *Multiagent System Technologies, 8th German Conference, MATES 2010, Leipzig, Germany, September 27-29, 2010. Proceedings*, pages 52–63.

[Criado et al., 2010c] Criado, N., Argente, E., Noriega, P., and Botti, V. J. (2010c). Towards a normative BDI architecture for norm compliance. In *Proceedings of The Multi-Agent Logics, Languages, and Organisations Federated Workshops (MALLOW 2010), Lyon, France, August 30 - September 2, 2010*, pages 65–81.

[Şensoy et al., 2010] Şensoy, M., Norman, T. J., Vasconcelos, W. W., and Sycara, K. (2010). OWL-POLAR: Semantic policies for agent reasoning. In Patel-Schneider, P., Pan, Y., Hitzler, P., Mika, P., Zhang, L., Pan, J., Horrocks, I., and Glimm, B., editors, *The Semantic Web – ISWC 2010*, volume 6496 of *Lecture Notes in Computer Science*, pages 679–695. Springer Berlin Heidelberg.

[Şensoy et al., 2012] Şensoy, M., Norman, T. J., Vasconcelos, W. W., and Sycara, K. (2012). OWL-POLAR: A framework for semantic policy representation and reasoning. *Web Semantics: Science, Services and Agents on the World Wide Web*, 12-13:148–160.

[da Silva et al., 2015] da Silva, V. T., Braga, C., and Zahn, J. O. (2015). Indirect normative conflict - conflict that depends on the application domain. In *ICEIS 2015 - Proceedings of the 17th International Conference on Enterprise Information Systems, Volume 1, Barcelona, Spain, 27-30 April, 2015*, pages 452–461.

[da Silva and Zahn, 2014] da Silva, V. T. and Zahn, J. O. (2014). Normative conflicts that depend on the domain. In Balke, T., Dignum, F., van Riemsdijk, M. B., and Chopra, A. K., editors, *Coordination, Organizations, Institutions, and Norms in Agent Systems IX*, volume 8386 of *Lecture Notes in Computer Science*, pages 311–326. Springer International Publishing.

[Diller, 1990] Diller, A. (1990). *Z: An introduction to formal methods*, volume 2. Wiley England.

[dos Santos Neto et al., 2010] dos Santos Neto, B. F., da Silva, V. T., and de Lucena, C. J. P. (2010). Using Jason to develop normative agents. In da Rocha Costa, A., Vicari, R., and Tonidandel, F., editors, *Advances in Artificial Intelligence – SBIA 2010*, volume 6404 of *Lecture Notes in Computer Science*, pages 143–152. Springer Berlin Heidelberg.

[dos Santos Neto et al., 2012] dos Santos Neto, B. F., da Silva, V. T., and de Lucena, C. J. P. (2012). An architectural model for autonomous normative agents. In Barros, L., Finger, M., Pozo, A., Gimenénez-Lugo, G., and Castilho, M., editors, *Advances in Artificial Intelligence - SBIA 2012*, Lecture Notes in Computer Science, pages 152–161. Springer Berlin Heidelberg.

[dos Santos Neto et al., 2013] dos Santos Neto, B. F., da Silva, V. T., and de Lucena, C. J. P. (2013). Developing goal-oriented normative agents: The NBDI architecture. In Filipe, J. and Fred, A., editors, *Agents and Artificial Intelligence*, volume 271 of *Communications in Computer and Information Science*, pages 176–191. Springer Berlin Heidelberg.

[Dung, 1995] Dung, P. M. (1995). On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artificial intelligence*, 77(2):321–357.

[Elhag et al., 2000] Elhag, A. A., Breuker, J. A., and Brouwer, P. (2000). On the formal analysis of normative conflicts. *Information & Communications Technology Law*, 9(3):207–217.

[Fenech et al., 2009] Fenech, S., Pace, G., and Schneider, G. (2009). Automatic conflict detection on contracts. In Leucker, M. and Morgan, C., editors, *Theoretical Aspects of Computing - ICTAC 2009*, volume 5684 of *Lecture Notes in Computer Science*, pages 200–214. Springer Berlin Heidelberg.

[Fenech et al., 2008] Fenech, S., Pace, G. J., and Schneider, G. (2008). Detection of conflicts in electronic contracts. *NWPT 2008*, page 34.

[Fitting, 1990] Fitting, M. (1990). First-order logic. In *First-Order Logic and Automated Theorem Proving*, Texts and Monographs in Computer Science, pages 97–125. Springer US.

[Gaertner et al., 2007] Gaertner, D., Garcia-Camino, A., Noriega, P., Rodriguez-Aguilar, J. A., and Vasconcelos, W. W. (2007). Distributed norm management in regulated multiagent systems. In *Proceedings of the*

6th International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS '07, pages 90:1–90:8, New York, NY, USA. ACM.

[Garcez et al., 2012] Garcez, A. D., Broda, K., and Gabbay, D. M. (2012). *Neural-symbolic learning systems: foundations and applications*. Springer Science & Business Media.

[Garcia et al., 2013] Garcia, E., Giret, A., and Botti, V. (2013). A model-driven CASE tool for developing and verifying regulated open MAS. *Science of Computer Programming*, 78(6):695–704.

[García-Camino et al., 2005] García-Camino, A., Noriega, P., and Rodríguez-Aguilar, J. A. (2005). Implementing norms in electronic institutions. In *Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems*, pages 667–673. ACM.

[García-Camino et al., 2007] García-Camino, A., Noriega, P., and Rodríguez-Aguilar, J. A. (2007). An algorithm for conflict resolution in regulated compound activities. In O'Hare, G., Ricci, A., O'Grady, M., and Dikenelli, O., editors, *Engineering Societies in the Agents World VII*, volume 4457 of *Lecture Notes in Computer Science*, pages 193–208. Springer Berlin Heidelberg.

[Gelfond and Lifschitz, 1991] Gelfond, M. and Lifschitz, V. (1991). Classical negation in logic programs and disjunctive databases. *New Generation Computing*, 9(3-4):365–385.

[Giannikis and Daskalopulu, 2009] Giannikis, G. K. and Daskalopulu, A. (2009). Normative conflicts-patterns, detection and resolution. In *WEBIST*, pages 527–532.

[Giannikis and Daskalopulu, 2011] Giannikis, G. K. and Daskalopulu, A. (2011). Normative conflicts in electronic contracts. *Electronic Commerce Research and Applications*, 10(2):247–267.

[Grossi et al., 2010] Grossi, D., Gabbay, D., and van der Torre, L. (2010). The norm implementation problem in normative multi-agent systems. In Dastani, M., Hindriks, K. V., and Meyer, J.-J. C., editors, *Specification and Verification of Multi-agent Systems*, pages 195–224. Springer US.

[Günay and Yolum, 2013a] Günay, A. and Yolum, P. (2013a). Constraint satisfaction as a tool for modeling and checking feasibility of multiagent commitments. *Applied Intelligence*, 39(3):489–509.

[Günay and Yolum, 2013b] Günay, A. and Yolum, P. (2013b). Engineering conflict-free multiagent systems. In *First International Workshop on Engineering Multiagent Systems (EMAS)*.

[Hill, 1987] Hill, H. (1987). A functional taxonomy of normative conflict. *Law and Philosophy*, 6(2):227–247.

[Holzmann, 2004] Holzmann, G. J. (2004). *The SPIN model checker: Primer and reference manual*, volume 1003. Addison-Wesley Reading.

[Kagal and Finin, 2005] Kagal, L. and Finin, T. (2005). Modeling communicative behavior using permissions and obligations. In van Eijk, R., Huget, M.-P., and Dignum, F., editors, *Agent Communication*, volume 3396 of *Lecture Notes in Computer Science*, pages 120–133. Springer Berlin Heidelberg.

[Kagal and Finin, 2007] Kagal, L. and Finin, T. (2007). Modeling conversation policies using permissions and obligations. *Autonomous Agents and Multi-Agent Systems*, 14(2):187–206.

[Kirkpatrick, 2015] Kirkpatrick, K. (2015). The moral challenges of driverless cars. *Commun. ACM*, 58(8):19–20.

[Kollingbaum, 2005] Kollingbaum, M. J. (2005). *Norm-governed practical reasoning agents*. PhD thesis, University of Aberdeen.

[Kollingbaum and Norman, 2004] Kollingbaum, M. J. and Norman, T. J. (2004). Strategies for resolving norm conflict in practical reasoning. In *ECAI Workshop Coordination in Emergent Agent Societies*, volume 2004.

[Kollingbaum and Norman, 2006] Kollingbaum, M. J. and Norman, T. J. (2006). Informed deliberation during norm-governed practical reasoning. In Boissier, O., Padget, J., Dignum, V., Lindemann, G., Matson, E., Ossowski, S., Sichman, J., and Vázquez-Salceda, J., editors, *Coordination, Organizations, Institutions, and Norms in Multi-Agent Systems*, volume 3913 of *Lecture Notes in Computer Science*, pages 183–197. Springer Berlin Heidelberg.

[Kollingbaum et al., 2006] Kollingbaum, M. J., Norman, T. J., Preece, A., and Sleeman, D. (2006). Norm refinement: Informing the re-negotiation of contracts. In *ECAI 2006 Workshop on Coordination, Organization, Institutions and Norms in Agent Systems, COIN@ ECAI*, volume 2006, pages 46–51.

[Kollingbaum et al., 2007] Kollingbaum, M. J., Norman, T. J., Preece, A., and Sleeman, D. (2007). Norm conflicts and inconsistencies in virtual organisations. In Noriega, P., Vázquez-Salceda, J., Boella, G., Boissier, O., Dignum, V., Fornara, N., and Matson, E., editors, *Coordination, Organizations, Institutions, and Norms in Agent Systems II*, volume 4386 of *Lecture Notes in Computer Science*, pages 245–258. Springer Berlin Heidelberg.

[Kollingbaum et al., 2008a] Kollingbaum, M. J., Vasconcelos, W. W., García-Camino, A., and Norman, T. J. (2008a). Conflict resolution in norm-regulated environments via unification and constraints. In Baldoni, M., Son, T., van Riemsdijk, M., and Winikoff, M., editors, *Declarative Agent Languages and Technologies V*, volume 4897 of *Lecture Notes in Computer Science*, pages 158–174. Springer Berlin Heidelberg.

[Kollingbaum et al., 2008b] Kollingbaum, M. J., Vasconcelos, W. W., García-Camino, A., and Norman, T. J. (2008b). Managing conflict resolution in norm-regulated environments. In Artikis, A., O'Hare, G., Stathis, K., and Vouros, G., editors, *Engineering Societies in the Agents World VIII*, volume 4995 of *Lecture Notes in Computer Science*, pages 55–71. Springer Berlin Heidelberg.

[Kyas et al., 2008] Kyas, M., Prisacariu, C., and Schneider, G. (2008). Run-time monitoring of electronic contracts. In Cha, S., Choi, J.-Y., Kim, M., Lee, I., and Viswanathan, M., editors, *Automated Technology for Verification and Analysis*, volume 5311 of *Lecture Notes in Computer Science*, pages 397–407. Springer Berlin Heidelberg.

[Li, 2013] Li, T. (2013). Normative conflict detection and resolution in cooperating institutions. In *Proceedings of the Twenty-Third international joint conference on Artificial Intelligence*, pages 3231–3232. AAAI Press.

[Li, 2014] Li, T. (2014). *Normative conflict detection and resolution in cooperating institutions*. PhD thesis, University of Bath.

[Li et al., 2014] Li, T., Jiang, J., Aldewereld, H., De Vos, M., Dignum, V., and Padget, J. (2014). Contextualized institutions in virtual organizations. In Balke, T., Dignum, F., van Riemsdijk, M. B., and Chopra, A. K., editors, *Coordination, Organizations, Institutions, and Norms in Agent Systems IX*, volume 8386 of *Lecture Notes in Computer Science*, pages 136–154. Springer International Publishing.

[Makinson and van der Torre, 2000] Makinson, D. and van der Torre, L. (2000). Input/output logics. *Journal of Philosophical Logic*, 29(4):383–408.

[McNamara, 2006] McNamara, P. (2006). Deontic logic. In *Logic and the Modalities in the Twentieth Century*, volume 7. North-Holland.

[Meneguzzi and Luck, 2009] Meneguzzi, F. and Luck, M. (2009). Norm-based behaviour modification in BDI agents. In *Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems-Volume 1*, pages 177–184. International Foundation for Autonomous Agents and Multiagent Systems.

[Meyer and Wieringa, 1993] Meyer, J.-J. C. and Wieringa, R. (1993). *Applications of deontic logic in computer science: A concise overview*. John Wiley & Sons.

[Meyer and Wieringa, 1994] Meyer, J.-J. C. and Wieringa, R. J. (1994). *Deontic logic in computer science: normative system specification*. John Wiley and Sons Ltd.

[Oren et al., 2008] Oren, N., Luck, M., Miles, S., and Norman, T. J. (2008). An argumentation inspired heuristic for resolving normative conflict. In *5th International Workshop on Coordination, Organisations, Institutions and Norms in Agent Systems (COIN@AAMAS 2008)*.

[Parsia and Sirin, 2004] Parsia, B. and Sirin, E. (2004). Pellet: An OWL DL reasoner. In *Third International Semantic Web Conference-Poster*, volume 18.

[Prisacariu and Schneider, 2007] Prisacariu, C. and Schneider, G. (2007). A formal language for electronic contracts. In Bonsangue, M. M. and Johnsen, E., editors, *Formal Methods for Open Object-Based Distributed Systems*, volume 4468 of *Lecture Notes in Computer Science*, pages 174–189. Springer Berlin Heidelberg.

[Reiter, 1980] Reiter, R. (1980). A logic for default reasoning. *Artificial intelligence*, 13(1):81–132.

[Russell and Norvig, 2009] Russell, S. and Norvig, P. (2009). *Artificial Intelligence: A Modern Approach (3rd Edition)*. Prentice Hall, 3rd edition.

[Searle, 1969] Searle, J. R. (1969). *Speech acts: An essay in the philosophy of language*, volume 626. Cambridge university press.

[Sierra et al., 2004] Sierra, C., Rodriguez-Aguilar, J. A., Noriega, P., Esteva, M., and Arcos, J. L. (2004). Engineering multi-agent systems as electronic institutions. *European Journal for the Informatics Professional*, 4(4):33–39.

[Vardi, 2015] Vardi, M. Y. (2015). On lethal autonomous weapons. *Commun. ACM*, 58(12):5–5.

[Vasconcelos et al., 2012] Vasconcelos, W. W., García-Camino, A., Gaertner, D., Rodríguez-Aguilar, J. A., and Noriega, P. (2012). Distributed norm management for multi-agent systems. *Expert Systems with Applications*, 39(5):5990–5999.

[Vasconcelos et al., 2007a] Vasconcelos, W. W., Kollingbaum, M. J., García-Camino, A., and Norman, T. J. (2007a). Achieving conflict freedom in norm-based societies. In *Workshop on Coordination, Organizations, Institutions, and Norms in Agent Systems*. Citeseer.

[Vasconcelos et al., 2007b] Vasconcelos, W. W., Kollingbaum, M. J., and Norman, T. J. (2007b). Resolving conflict and inconsistency in norm-regulated virtual organizations. In *Proceedings of the 6th international joint conference on Autonomous agents and multiagent systems*, page 91. ACM.

[Vasconcelos et al., 2009] Vasconcelos, W. W., Kollingbaum, M. J., and Norman, T. J. (2009). Normative conflict resolution in multi-agent systems. *Autonomous Agents and Multi-Agent Systems*, 19(2):124–152.

[Vasconcelos and Norman, 2009] Vasconcelos, W. W. and Norman, T. J. (2009). Contract formation through preemptive normative conflict resolution. In *Proceedings of the 2009 Conference on Artificial Intelligence Research and Development: Proceedings of the 12th International Conference of the Catalan Association for Artificial Intelligence*, pages 179–188, Amsterdam, The Netherlands, The Netherlands. IOS Press.

[von Wright, 1951] von Wright, G. H. (1951). Deontic logic. *Mind*, 60(237).

[Weiss, 1999] Weiss, G. (1999). *Multiagent systems: a modern approach to distributed artificial intelligence*. MIT press.

[Wooldridge, 2009] Wooldridge, M. J. (2009). *An Introduction to MultiAgent Systems (2nd Edition)*. Wiley, 2nd edition.

[Zahn, 2015] Zahn, J. O. (2015). Um Mecanismo de Verificação de Conflitos Normativos Indiretos. Master's thesis, Instituto de Computação - Universidade Federal Fluminense (IC/UFF), Niteroi, Brasil.

[Zahn and da Silva, 2014] Zahn, J. O. and da Silva, V. T. (2014). On the checking of indirect normative conflicts. In *Workshop Escola de Sistemas de Agentes, seus Ambientes e aplicações, 2014, Porto Alegre. Anais do Workshop-Escola de Sistemas de Agentes, seus Ambientes e aplicações*, pages 13–24.