Using Normative Markov Decision Processes for Evaluating Electronic Contracts

Moser Silva Fagundes [a,*], Sascha Ossowski [a], Michael Luck [b] and Simon Miles [b]

[a] *Centre for Intelligent Information Technologies*
University Rey Juan Carlos, Madrid, Spain
E-mail: {moser.fagundes,sascha.ossowski}@urjc.es
[b] *King's College London*
Department of Informatics
E-mail: {michael.luck,simon.miles}@kcl.ac.uk

Before signing electronic contracts, a rational agent should estimate the expected utilities of these contracts and calculate the violation risks related to them. In order to perform such pre-signing procedures, this agent has to be capable of computing a policy taking into account the norms and sanctions in the contracts. In relation to this, the contribution of this work is threefold. First, we present the Normative Markov Decision Process, an extension of the Markov Decision Process for explicitly representing norms. In order to illustrate the usage of our framework, we model an example in a simulated aerospace aftermarket. Second, we specify an algorithm for identifying the states of the process which characterize the violation of norms. Finally, we show how to compute policies with our framework and how to calculate the risk of violating the norms in the contracts by adopting a particular policy.

Keywords: Norms, Electronic Contracts, MDP, NMDP

1. Introduction

In regulated multiagent systems [20,31,15] the agents may be subject to mechanisms for adjusting their behaviour at some level in order to orchestrate a global behaviour. The usefulness of such regulations becomes more prominent in open systems, where heterogeneous agents are able to join and leave the system at runtime. In these open systems, there are no guarantees that the agents will act in a particular manner, and in this case, the establishment of some type of control over them makes possible the coordination of tasks and the avoidance of particular undesired states of the world.

Electronic contracts have been shown to be suitable for regulating the behaviour of agents in several domains, such as for service procurement in the insurance industry, service level agreement management in software engineering, and aircraft engine aftercare [24]. While we do not detail the specific applications in this paper, [24] provides substantial further details of the use cases for these. In particular, electronic contracts express the responsibilities of each of the involved parties through the specification of norms; it is this core normative aspect of contracts that is the focus of this paper. These contracts thus represent agreements of the parties, making explicit what each party can expect from the others, but providing flexibility in how they accomplish their own obligations.

Once a contract has been specified as a set of norms, we can divide its lifetime in two phases:

- **Pre-signing:** the agents *evaluate* the contract by computing their expected earnings and probabilities of norm violations (risks) by the addressees of the contracts;
- **Post-signing:** if the contract is signed, the agents engage in its *execution*; the signed contract establishes a commitment whose violation implies the execution of sanctions; in order to ensure the detection of such deviations, *monitoring* activities are performed [35].

In order to behave rationally in a system regulated by contracts, a self-interested agent must be capable of calculating the expected utilities of signing them, and estimating the likelihood, or risk, of norm violations for a particular course of action. In this paper, we focus on the *pre-signing* phase, aiming at the evaluation of contracts by a rational agent in a stochastic environment where norm violations may happen intentionally or as a consequence of the intrinsic uncertainty of the system.

*Corresponding Author: Moser Silva Fagundes
Room 2013-B (Annexe to the Rectorate)
Calle Tulipán S/N
Móstoles (28933), Madrid, Spain.

AI Communications
ISSN 0921-7126, IOS Press.

The contribution of this paper is threefold. First, we put forward the *Normative Markov Decision Process* (NMDP) framework, an extension of the *Markov Decision Process* (MDP) framework [3] for explicitly representing norms. In order to illustrate the application of NMDPs, we develop a case study in a simulated aerospace aftermarket domain. In this case study, we implement an engine manufacturer agent which evaluates sets of contracts that can be signed with airline operators and suppliers of engine parts. Second, we specify an algorithm for identifying the states of the process which characterize the violation of the norms in the contracts. Finally, we show how to compute policies for NMDPs and how to calculate the risk of violating a contract by adopting a particular policy. In this respect, the focus of this paper is on the norms within a contract rather than any other concern.

The paper is organized as follows. Section 2 critically analyses related work and extracts requirements for our approach. Section 3 introduces the NMDP framework. Section 4 presents our case study in the aerospace aftermarket domain. Section 5 describes how to associate states with norm violations. Section 6 proposes a way of calculating the probability of violating contracts. Finally, Section 7 draws the conclusion and future research directions.

2. Related work

In this section, we expound our motivation for developing the model shown in Section 3. As indicated above, our focus is specifically on the norms within contracts that express the obligations of the different parties, and we thus undertook a survey of normative agent architectures, where the autonomous agent research meets the ideas from deontic logic and traditional normative systems studied in Philosophy and Laws. Normative reasoning approaches that are not linked to a particular agent architecture lie outside the scope of this section. For a broader review of relevant research on normative multiagent systems, see Criado et al. [16].

As our survey confirms, most research on normative reasoning by autonomous agents has been done from a practical reasoning perspective, through goal-oriented architectures. Such work proposes cognitive models for weighting competing alternatives (some of them non-compliant with the norms) on the basis of preference orders. Alternatively, some approaches apply normative reasoning with utility-based models, focusing on the maximization of rewards, which is used as a quantitative measure of the profits and losses brought about by the adoption of a given set of norms.

In order to compare the norm-aware agent models, they are analysed along five dimensions. Table 1 provides a summary of the outcome of this analysis, where the properties accounted for in the respective agent architectures are checked.

- **Norm-autonomy**: determines whether an agent is capable of exercising control over its own actions that are regulated by the norms; thus, a norm-autonomous agent is assumed to be free to accept or refuse norms, as well as violating or complying with them. On the other hand, an agent with no autonomy with respect to norms does not control its own actions which are ruled by the norms; this is the case of agent architectures where norms are automatically enforced by hard-wiring them into the agent's specification;
- **Explicitness of norms**: this dimension concerns the property of having norms represented as some explicit normative structure, which allows the agents to be informed about the norms regulating the environment. It also eases the process of mirroring changes in the norms, and can be used for communicating norms to other agents;
- **Explicitness of sanctions**: this dimension corresponds to the possibility of explicitly representing sanctions, which are reactions used to enforce norms when they are violated;
- **Quantitative decision model**: refers to agent architectures that make decisions exclusively on the basis of quantitative information, namely expected utilities, obtained from executing actions and/or achieving states of the world; on the other hand, BDI and goal-oriented agent architectures usually use some preference orderings related to agents' mental attitudes in order to determine the intentions or goals to be pursued. For a detailed comparison between classical decision theory with qualitative decision theory, knowledge-based systems and BDI models, see [17].
- **Non-determinism**: concerns an agent's capability to account for uncertainty about the outcome of its actions; this dimension is related to the realization of stochastic strategies or action sequences to achieve some state of affairs (e.g. means-end reasoning of BDI agents) or maximize expected utilities (e.g. MDPs).

Table 1
Summary of the properties of the norm-aware agent models.

Work reference	Norm-autonomy	Explicitness of norms	Explicitness of sanctions	Quantitative decision model	Non-determinism
Castelfranchi et al. [14]	✓	✓			
Boman [6]				✓	✓
Dignum et al. [19]	✓	✓			✓
Broersen et al. [9]	✓	✓			
Kollingbaum and Norman [26]	✓	✓			
Boella and van der Torre [4]	✓	✓	✓		
Ågotnes et al. [1]	✓	✓		✓	
Andrighetto et al. [2]	✓	✓	✓		
López et al. [31]	✓	✓	✓		
Meneguzzi and Luck [34]	✓	✓			
Cardoso and Oliveira [11]	✓	✓	✓	✓	
Joseph et al [25]	✓	✓	✓		
Oh et al. [38]		✓		✓	✓

In the rest of this section, we briefly outline some of the most relevant norm-aware agent architectures and promote a discussion which relates these models to the problem of reasoning rationally about contracts.

2.1. Castelfranchi et al. (1999)

Deliberative normative agents are agents that have explicit knowledge about the enacted norms in a multiagent environment and can make a choice whether to obey the norms or not in specific cases. Castelfranchi et al. [14] propose a *generic* architecture for deliberative normative agents, which is able to know that a norm exists in a society, able to adopt that norm, able to deliberatively follow that norm, and able to deliberatively violate that norm in a intelligent way. This architecture, designed as a refinement of the generic agent model proposed by Brazier et al. [8], includes components for reasoning in environments governed by norms.

The generic model is one of the first efforts to incorporate norms into an agent architecture. Until then, experiments with normative agents aimed at running social simulations so as to compare selfish and altruistic behaviours which were hard-coded into the agents' specification – these agents could not modify their behaviour over the time based on their experience. According to the authors, the precise knowledge by which goals are generated depends on the application addressed; this generic normative agent model only provides elements that can be used; it does not commit to a specific approach to goal generation. Therefore, an implementation capable of evaluating contracts with explicit sanctions in stochastic environments could be made, but it is up to the designer to choose the technique for managing the goals of the agent.

2.2. Boman (1999)

Boman [6] describes a method for enforcing norms onto *supersoft* agents programmed to represent and evaluate vague and imprecise information. These agents are assumed to behave in accordance with advices obtained from their individual *decision module*, with which they can communicate. Such decision module contains algorithms for efficiently evaluating supersoft *decision data* concerning probability, utility, credibility, and reliability. Three ways of enforcing norm-compliant behaviour on the agents are proposed. The first one consists of manipulating the utilities in the lowest level to skew assessments of the *decision data* to have an overly positive or negative attitude towards some consequences. The second way proceeds via the elimination of actions with disastrous consequences. The last way consists of disqualifying certain actions by referring to their negative impact on the global utility in the multiagent system. This last option is a form of social norm adoption and it requires knowledge regarding how to find the global utility values.

In fact, this work does not propose an agent model, but a method for enforcing norm-compliance onto a particular type of *utility-driven* agent. It assumes the existence of a decision module, whose advices are always followed by the agent. In other words, the agents are told what they must and must not do. In this approach norms cannot be violated and sanctions cannot be represent. This is a severe limitation if we assume that electronic contracts can be violated and sanctions can be imposed, especially in stochastic domains where the actions' outcomes can be unpredictable.

2.3. Dignum et al. (2000)

Dignum et al. [19] present an approach to social reasoning that integrates prior work on norms and obligations [42] with the BDI agent model developed by Rao and Georgeff [40]. Such an integration aims at introducing norms and obligations to support the socially motivated deliberation process of the agent. This type of agent has knowledge about the enacted norms and makes choices whether or not to obey norms, and how to weigh the impact of punishments for obligation violation in particular cases. The norms are not hard-wired into the agents: circumstances might change, making norms obsolete; and agents might interact with other agents that follow different norms, so an explicit representation of norms and obligations can support a more flexible and appropriate reasoning.

The main control algorithm of the architecture is similar to the one presented in [40], except that the selected events used in the *option-generator* are augmented with *potential deontic events* related to the applicability of norms and obligations. Constraints between the obligations are handled in the *option-generator*, which outputs a set of plans that can be executed simultaneously. In other words, each option contains a set of plans that are compatible among them. The *deliberation* process focuses on the selection of plans on the basis of *preferences*. The preference ordering of norms is based on a preference of social benefit of a situation, while the preference ordering of obligations are based on the punishments for violating them.

This approach focuses on the development of socially responsible agents which adopt norms in an attempt to support collaborative behaviour. The authors point out reasons why an agent might not comply with an applicable norm automatically: the norm is in conflict with other norms; the norm does not achieve its original intention; and, the norm is applicable but the agent has not adopted it. Only the first reason is dealt with in the paper. Choices between conflicting norms are made with predefined preference orderings, rather than by derived utilities.

2.4. Broersen et al. (2002)

In the BOID architecture, the goals of the agent are generated from the interaction between beliefs, obligations, intentions and desires. In the paper [9], the authors focus on the goal generation, which uses these four primitive mental attitudes and is biased by the type of agent – *realistic*, *stable*, *selfish* and *social*.

The compliance with norms in the BOID architecture depends on the establishment of commitments to the respective normative goals. The agent's candidate goals are selected based on a static priority function on the rules that govern the agent's behaviour, which also determine which inference steps must be made. As a consequence, some rules may be overridden by others, enabling the resolution of conflict between mental attitudes by different agent types. Thus, BOID agents always consider norms in the same manner; that is, they cannot decide to follow or violate a given norm according to their circumstances. Despite the fact that BOID agents explicitly represent obligations through a particular mental attitude, the observations obtained from the environment never change these obligations. The authors briefly discuss the impact of obligations in the planning and scheduling. Uncertainty with respect to the action outcomes and the notion of utility are not accounted, which makes this model inappropriate for estimating costs of signing contracts and risks of breaking norms in stochastic environments.

2.5. Kollingbaum and Norman (2003)

The Normative Agent Architecture [26], referred to as NoA, aims to support the development of norm-motivated practical reasoning agents. NoA is based on classic BDI concepts with extensions that allow an agent to reason about norms. It is implemented as a reactive planning architecture composed of two main elements: the NoA *language* for the specification of plans and norms and the NoA *interpreter*, which is capable of interpreting and executing plan and norm specifications formulated in the NoA language. The NoA language contains constructs for the specification of beliefs, goals, plans and norms. The interpreter, through the informed deliberation [29], allows the agent to remain *norm-autonomous* in that options for forbidden actions (or plans) which are not excluded, but are instead labelled as forbidden and remain options if an agent chooses to act in violation to resolve conflicts between norms. The consistency problem of norms in NoA is detailed in [27,28].

This architecture is influenced by AgentSpeak(L) [39], with extensions that allow an agent to reason about norms. It takes influences from classical planners with respect to the declaration of plans – the behaviour of the agent is determined by pre-specific plans composed of deterministic actions. In this model, punishments for norm violations are not taken into account – the agents aim at maximizing the consistency level of the adopted norms, and norm violations take place only in circumstances where compliance is not possible.

2.6. Boella and van der Torre (2006)

Boella and van der Torre [4] develop a formal game theoretic model for agents negotiating contracts. This work introduces a qualitative game theory based on recursive modeling, where the agents recursively model the normative system to predict whether their behavior counts as a norm violation and will be sanctioned. In this model of normative multiagent system, the agents' decisions are made on the basis of priority relations (qualitative), which resolve conflicts among motivations (desires and goals), and the outcome of the agents' actions are deterministic.

2.7. Ågotnes et al. (2007)

Ågotnes et al. [1] develop a normative multiagent system in which the agents are assumed to have multiple goals of increasing priority. In this model, transitions are represented through Kripke structures and norms are implemented as constraints over these structures. A normative system is then simply a subset of the Kripke structure, which contains the arcs that are forbidden by the normative system. The goals of the agents are specified as a hierarchy of formulae of Computational Tree Logic, which defines a model of ordinal utility, making possible the interpretation of the Kripke-based normative systems as games. Thus, the agents decide whether to defect or not from the normative system based on prioritized lists of goals and game theoretic solution concepts.

This work accounts for strategic behaviour by agents when deciding whether to comply with the normative system or not. To do so, the authors use an ordinal utility which allows the interpretation of their Kripke-based normative system as games, in which agents determine whether to follow the norms. In this work, norms are limited to prohibitions, which are explicitly represented as forbidden transitions in the Kripke structure. The representation of sanctions and sequential decision making are outside the scope of their work. When dealing with contracts, calculating the costs of executing the contract and handling sanctions is fundamental so as to take a rational decision.

2.8. Andrighetto et al. (2007)

Andrighetto et al. [2] analyse inter agents and intra agent processes needed to deal with the emergence of norms. To do so, the authors put forward EMIL-A, an agent architecture for recognising new norms and generating normative beliefs, deciding whether to adopt the norms, generating normative goals, determining whether to comply with them generating normative intentions, and finally, generating plans to achieve the normative intentions. In order to support the normative reasoning, there is also a normative long term memory containing a set of existing norms and normative information, and a repertoire of normative action plans consisting of norm-compliant actions. EMIL-A agents are capable of determining the pertinence of norms and their degree of activation, that is, the norm salience. This norm salience is used as a criterion for accepting or rejecting norms.

With this agent architecture, the authors explain the phases that norms undergo so as to evolve from the environment into the internal state of norm-autonomous agents. The decision whether to obey a given accepted norm is determined by the expected utility that agents should obtain if they fulfil or violate the norm, taking into account the sanctions and incentives associated to the normative frame. However, the work presented in [2] neither includes an explicit representation of sanctions, nor does it detail the normative action planner.

2.9. López et al. (2007)

López et al. [31] present a formal normative framework for agent-based systems that includes a canonical model of norms, a model of normative multi-agent systems and a model of normative autonomous agents. In this paper, the authors put together the framework components, whose publication has been done in different forums.

To comply with a norm, the agent's deliberation process evaluates the set of goals that might be hindered by satisfying the normative goals, and set of goals that might benefit from the associated rewards. On the other hand, to reject a norm, an agent evaluates the damaging effects of punishments. Similarly to the BOID architecture, this autonomous normative agent architecture adopts the notion of stereotypes (*social*, *rebellious*, *pressured*, *opportunistic*), which are mapped to goal selection strategies ruled by priority functions.

2.10. Meneguzzi and Luck (2009)

Assuming that agents are to operate in open environments, they need to adapt to changes in the norms that regulate such multiagent environments. In response, the paper [34] provides a technique to extend BDI lan-

guages by enabling them to enact behaviour modification at runtime in response to newly accepted norms.

To represent norms it proposes a schema, which contains the *norm* itself, its *activation condition* and *expiration condition*. The norm has two possible deontic modalities: *prohibition* and *obligation*. Both can refer to declarative world states or actions. The interpreter includes meta-level actions that allow an agent to scan its own plan library for plans that would violate a set of norms that an agent has previously accepted to comply. For prohibitions, violating plans are temporarily removed from the library while the prohibition is in effect. On the contrary, for obligations, new plans are created using a planning mechanism [32] so that an agent has plans that can accomplish such norms.

This work focuses on the problem of adapting the agent's behaviour to the *accepted* norms. The technique shown in this work could be employed for the development of agents capable of violating norms, however it would demand the implementation of a process for deciding what norms are accepted or rejected.

The technique proposed by this proposal focuses on the problem of adapting the agent's behaviour to the *accepted* norms. Such a technique can be employed for the development of agents capable of adapting to contracts, however it requires the implementation of a decision model for choosing which clauses in the contract (norms) are accepted or rejected. The strategy used by this decision process restricts the autonomy level of the agent with respect to the norms. The authors demonstrate the viability of their approach through an implementation in AgentSpeak(L) [39], a language that uses first-order logic to represent beliefs, with which is not possible to represent probabilistic information.

The authors also develop a global monitoring architecture [35] for determining the source of violations in signed contracts – *post-signing* phase. The monitor agents gather information from entrusted observers and generate explanations for violations within a supply chain in a simulated aerospace aftermarket. The explanations are employed to detect norm violations at runtime and to improve the contracts by relaxing some prohibitions.

2.11. Cardoso and Oliveira (2009)

Cardoso and Oliveira [11] implement norm-aware utility-oriented agents to study adaptive mechanisms that enable a normative framework to change deterrence sanctions in contracts according to an agent population. Although the agents are essentially utility maximizers, they are characterized by different levels of *risk tolerance* and *social awareness*, which makes possible the generation of heterogeneous populations. The risk tolerance affects the decisions regarding opportunities to sign new contracts, and corresponds to the agent's willingness to contract in the presence of violation penalties. An agent decides to contract on the basis of the highest fine that is associated with the commitments for the assigned role. In order to contract, the following relation must be true:

$$hf(role) \leq b * RT/(1 - RT)$$

where $hf(role)$ is the highest fine for the given $role$, b is a slope parameter related to the agent's budget and RT is the risk tolerance parameter. The social awareness, on the other hand, is related to decisions concerning ongoing contracts. This last parameter impels the agent to fulfil its obligations even when it does not have a strict advantage in doing so. An agent decides to fulfil an obligation o whenever the following relation is true:

$$v(o) - f(o) \leq b * SA/(1 - SA)$$

where $v(o)$ is the violation outcome, $f(o)$ is the fulfilment outcome, b is a slope parameter related to the agent's budget and SA is the agent's social awareness parameter.

However, the agents developed in this work do not construct plans. Based on the social awareness and risk tolerance parameters, the agents calculate the expected utilities and decide whether to violate a norm or not. The performance of individual agents is not studied since the authors focus on the effectiveness of adaptive regulative mechanisms at the macro-level.

2.12. Joseph et al. (2010)

Joseph et al. [25] formalize the principles of deductive coherence proposed by Thagard [41], and then, specify a coherence-driven agent architecture which extends the BDI theory with the theory of coherence. Such an agent architecture takes decisions and actions, possibly non-normative, based on the coherence maximization. Based on the coherence framework, the authors propose a BDI architecture for coherence-driven agents. This agent architecture is an adaptation of the multi-context graded BDI model [13], on which the theories of the contexts will produce coherence graphs. In the norm context, a graded norm is interpreted in terms of its priority, which corresponds to a measure of

its importance within a system of norms. To represent and reason with norms, the authors use the Probability-valued Deontic Logic [18], which do not represent of sanctions. In order to relate norm violations to sanctions, the coherence-driven agent architecture uses the implication operator: a proposition representing a violating state entails a proposition representing the effects of the respective sanction.

The representation of non-deterministic information is accounted with grades, and the sanction outcomes are represented as beliefs. The norm adoption process is made by maximizing the coherence between mental states in the place of maximizing expected utilities. An agent implemented with this architecture would sign the contracts evaluated as the most coherent ones with respect to the agent's current state of mind, which are not necessarily the most profitable ones. A planning context is described in the graded BDI model [12], however, this agent architecture does not specify this context.

2.13. Oh et al. (2011)

Oh et al. [38] describe an agent for normative reasoning assistance that can proactively prevent human users from violating norms in a time-constrained environment. This assistant agent is composed of a *plan recognizer*, a *norm reasoner* and a *planner*. The plan recognizer, introduced in [37], identifies the user's needs in advance so the agent works in parallel with the user to ensure that the assistance is ready by the time the user needs it. Based on the assumption that human users generally reason about consequences and take decisions to maximize their long-term rewards, the agent represents the planning problem as a Markov Decision Process (MDP) and uses an optimal policy to predict the future activities of the user.

Inspired by the normative structure proposed in [21], a norm is defined in terms of its *deontic modality*, its *context condition* specifying when a norm is relevant to a given state, and its *normative condition* specifying the constraints imposed to the agent when the norm is considered relevant.

Given a predicted user plan in a plan-tree, the norm reasoner visits each node in the tree and evaluates the associated user state for any norm violations. Thus, the assistant can alert the user of active violations and proactively steer the user away from those violations that are likely to happen in the future. In order to accomplish this, for each state that violates a norm, the agent looks for the nearest state that is compliant with all norms. When a compliant state is found, this state becomes a new goal state for the agent, generating a planning problem to the planner module such that the agent needs to find a series of actions to move from initial state to this goal state.

In the work described in [38], prohibitions and obligations are addressed to the human users, not to the agent. Thus, the assistant agent cannot be considered norm-autonomous, only the assisted human users can. The agent is deployed in a time-constrained collaborative environment, where no sanctions are imposed to norm violations.

3. Normative MDPs and contracts

Motivated by the discussion promoted in the previous section, we put forward our proposal for modelling norm-autonomous agents, acting in non-deterministic and dynamic environments. The individually rational decision-making of our agents is based on the expected rewards of their actions in such an environment, where norm transgressions can be explicitly sanctioned.

For this purpose, our agents make use of the well-known *Markov Decision Process* (MDP) framework [3], a formal mathematical framework widely utilized for modelling decision making, planning and control in stochastic domains. It provides a model of an agent interacting synchronously with its environment, where the actions' outcome may be subject to stochastic dynamics.

A MDP can be described as a tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{C}, \mathcal{T}, \mathcal{R} \rangle$ where: $\mathcal{S}$ denotes a finite set of states of the world; $\mathcal{A}$ denotes a finite set of actions; $\mathcal{C} : \mathcal{S} \rightarrow \mathcal{A}$ is a capability function that denotes the set of admissible actions for each state ($\mathcal{C}(s_i)$ corresponds to the set of admissible actions in s_i); $\mathcal{T} : \mathcal{S} \times \mathcal{A} \rightarrow \Pi(\mathcal{S})$ is a state-transition function, giving for each state and action, a probability distribution over states ($\mathcal{T}(s_i,a,s_j)$ for the probability of executing a at s_i, and ending at s_j); and finally, $\mathcal{R} : \mathcal{A} \rightarrow \Re$ corresponds to a reward function that gives the expected immediate reward gained by the agent for executing a given action ($\mathcal{R}(a)$ corresponds to the immediate reward for executing a).

Previous research, as shown in the overview made by Boutilier et al. [7], has demonstrated that *intentional* or *factored* representations for state spaces from AI planning can be used to design efficient MDP solution techniques which assume a certain *structure* in the state space. Differently from an *extensional* representation in which states are enumerated directly, the *factored*

representations make the state space structure explicit through *factors* or *features*. Let a feature $f_i, 1 \leq i \leq x$, take on a finite number of values and let $\mathcal{F}_{f_i}$ stand for the finite set of possible values. Then, a state is any possible assignment of values to these features, and the state space $\mathcal{S}$ is the cross product of the value spaces for the individual features as follows:

$$\mathcal{S} = \times_{i=1}^{x} \mathcal{F}_{f_i}$$

Still, in normative systems some of the interactions between an agent and its environment may be regulated by norms. A way of expressing the normative behaviour in standard MDPs (independently of whether they are based on iconic or factored state representations) would be to hard-code them as constraints on the agent's transition model and capability function. However, such an approach does not account for an explicit representation of norms, which constrains the agent's ability of mirroring changes in the contracts. Besides, self-interested rational agents should have the freedom to ignore a norm in order to maximize their expected utilities, which is impossible if the norm-compliance is hard-coded into the MDP. Bearing in mind these limitations, we put forward the *Normative Markov Decision Process* (NMDP) framework, an extension of the MDP framework for explicitly expressing norms utilizing elements in the state space and action space. With such an extension we offer ease of modification of the process, keeping the actions which may not be norm-compliant.

The NMDP framework extends the MDP framework so as to include a parameter for explicitly representing norms. A NMDP can be described as a tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{C}, \mathcal{T}, \mathcal{R}, \mathcal{N} \rangle$ where $\mathcal{N}$ is the set of known norms; and the remaining parameters are inherited from the MDP framework.

Norms can be classified with respect to their purpose as defined in [5]. In this classification, the authors distinguish between *substantive* and *procedural* norms. The first provides the specifications of how the agents should behave, and the second type of norm describes the reactions on violations of the substantive norms. We describe a substantive norm as a set of *states* that are *prohibited* or *obliged* for an *addressee* agent. Regarding sanctions, our approach does not represent them as obligations triggered by violations and addressed to enforcer agents. Instead, we represent the *outcomes* (effects) of the sanctions on the addressees of the substantive norms. These outcomes are expressed as modifications to be made in the agent's capability function and transition function.

On the basis of these considerations, we use the following form to represent a norm:

(*modality*, *addressee*,
beneficiary, *content*,
sanction)

where the following constraints hold:

- *modality* $\in \{$*prohibition, obligation*$\}$.
- *addressee* $\in \Gamma$; the *addressee* is an agent to which the norm applies, and Γ is the set of all agents participating in the system.
- *beneficiary* $\in \Gamma$; the *beneficiary* is an agent that benefits from the norm compliance.
- *content* (normative content) is represented as a logical sentence ϱ specified with *features* of the state space $\mathcal{S}$ and logical connectives. The following grammar defines its syntax:

$$\varrho ::= \alpha \mid \varrho \vee \varrho \mid \varrho \wedge \varrho \mid (\varrho)$$
$$\alpha ::= (f = v) \mid \neg(f = v)$$

  where f is a feature of the state space $\mathcal{S}$, $v \in \mathcal{F}_f$ (assigned *value* for f), $\neg$ is the negation operator, $\vee$ and $\wedge$ are the connectives disjunction and conjunction, respectively, and the parentheses are for setting priorities. In the absence of parentheses, operators are left-associative. The way this logical sentence is interpreted depends on the deontic modality of the norm. Its interpretation is detailed in Section 5.

- *sanction* has the form:

  (*outcome*$_{\mathcal{T}}$, *outcome*$_{\mathcal{C}}$)

  where:

  - *outcome*$_{\mathcal{T}}$ specifies a set of modifications to the state-transition function[1]; an element of this set has the form (*state*$_i$, *action*, *state*$_j$, $[0 \ldots 1]$), which indicates the probability of executing an *action* at *state*$_i$ and ending at *state*$_j$;
  - *outcome*$_{\mathcal{C}}$ specifies a set of modifications to the capability function; an element of this set has the form (*state*, *action*, $\{0,1\}$), where 1 means that the *action* is admissible in the *state*, 0 otherwise.

[1] In the context of this work, the transition and capability functions are implemented as lookup tables; then, modifications to these functions consist of updating the values in the table denoting the state-transition probabilities and action admissibility, respectively.

Contracts are enforceable agreements between two or more parties with clauses expressing mutual obligations and prohibitions. We represent contracts as sets of norms, which are encoded in the NMDP specification.

For instance, assume a set of contracts $\{c_1, \ldots, c_n\}$ where each contract c_i is specified as a set of norms. Thus, if we want to reason about these contracts, we have to specify the set of norms $\mathcal{N}$ of the NMDP as the union of these contracts:

$$\mathcal{N} = \bigcup_{i=1}^{n} c_i$$

4. Case study

In this section we introduce the aerospace aftermarket domain, and then, we specify our engine manufacturer agent which inhabits a simulated aftermarket.

4.1. Aerospace aftermarket domain

The aerospace aftermarket use case described in this section was first introduced by Jakob et al. [24], and further developed by Meneguzzi et al. [33]. According to the authors, the aerospace aftermarket is increasingly populated by customers buying a service rather than a product. In this domain, the aircraft engine manufacturers provide long term commitments, which make these manufacturers responsible for providing serviceable engines to airline operators at specific locations (a given engine manufacturer may service an airline operator at multiple sites), not allowing any aircraft to be idle for greater than an agreed duration.

Minimum service level commitments are stipulated in *aftercare contracts*. If these commitments are violated (for example, when an airline aircraft is on the ground, awaiting functioning engines for a period of time greater than that agreed with the engine manufacturer), then engine manufacturers receive predetermined financial penalties. In this business model, servicing and maintenance becomes a key driver of profitability for the engine manufacturer since aftercare contracts are worth millions of euros.

On the basis of these aftercare contracts, the engine manufacturers establish *parts supply contracts* with engine part suppliers. In order to repair an engine, a manufacturer requests the required engine parts from contracted suppliers. Once all engine parts have been obtained, the manufacturer resumes repairing the engine, and readies it in the designated aircraft, notifying the airline operator that the repair has been accomplished.

There are three relevant types of autonomous agents in the aerospace aftermarket domain (see Figure 1):

- **Airline operators** are the customers for aftercare contracts; each operator has its own fleet of aircraft which needs to be kept in service;
- **Engine manufacturers** are providers of aftercare contracts; they attempt to perform the engine repair as specified in the contract or incur penalties; moreover, manufacturers are customers for parts supply contracts;
- **Part suppliers** are providers of parts supply contracts; they deliver engine parts to the manufacturers.

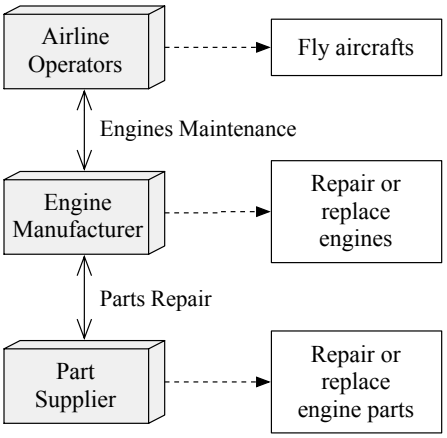

Fig. 1. Actors and services in the aerospace aftermarket domain.

The provision of services is regulated by contractual agreements between the involved parties. Two particular types of contract are studied in this work:

- **Aftercare contract** which specifies the terms and conditions under which an engine manufacturer undertakes to supply and maintain engines for an airline operator; an aftercare contract can specify, for example, the serviceable engine rate, financial penalties applicable if the agreed service levels are not met, etc;
- **Parts supply contract** which regulates how an engine manufacturer asks a supplier of engine parts to produce and deliver new parts or refurbished old parts of a given type over a given period; this contract can specify, for example, the locations where part supplies should be delivered, the cost of parts, delivery times, etc.

4.2. Engine manufacturer agent

In this subsection we employ the agent model introduced in Section 3 so as to model an engine manufacturer agent as described in the Subsection 4.1.

Our engine manufacturer agent (em) is part of an environment populated by other four agents, one airline operator (ao) and three part manufacturers (sx, sy and sz). There are four possible contracts in our example: one aftercare contract ac to be signed with ao, and three parts supply contracts, pscx, pscy and pscz which can be signed with sx, sy and sz, respectively. For the sake of simplicity we assume that only part p is required to perform the repair of an engine, and the three suppliers are capable of providing this part. Thus, the manufacturer em signs one aftercare contract and one parts supply contract, which results in three possible sets, each of them representing a set of contracts:

$$\mathcal{W}_1=\{\text{ac}, \text{pscx}\}$$
$$\mathcal{W}_2=\{\text{ac}, \text{pscy}\}$$
$$\mathcal{W}_3=\{\text{ac}, \text{pscz}\}$$

For each set $\mathcal{W}_i$ we specify a nmdp_i whose set of norms $\mathcal{N}_i$ contains the contractual norms in $\mathcal{W}_i$. The computation of expected utilities for these NMDPs will indicate the most profitable set of contracts.

$$\text{nmdp}_1 = \langle \mathcal{S}_1, \mathcal{A}, \mathcal{C}_1, \mathcal{T}_1, \mathcal{R}_1, \mathcal{N}_1 \rangle$$
$$\text{nmdp}_2 = \langle \mathcal{S}_2, \mathcal{A}, \mathcal{C}_2, \mathcal{T}_2, \mathcal{R}_2, \mathcal{N}_2 \rangle$$
$$\text{nmdp}_3 = \langle \mathcal{S}_3, \mathcal{A}, \mathcal{C}_3, \mathcal{T}_3, \mathcal{R}_3, \mathcal{N}_3 \rangle$$

The state space of these NMDPs is described using sets of multi-valued features, as follows:

- **Engine part** – f_1. Has em ordered the part p required to perform the repair of the engine e? If false, this feature assumes the value $\overline{\text{O}}$. If the engine part p has been ordered, it assumes the value O. If p has been received, the feature assumes the value X, Y or Z, which corresponds to the respective supplier sx, sy or sz.
- **Order deadline** – f_2. Is the engine part supplied by the order deadline specified in the supply contract? Initially, this feature assumes the value A, which means that the supplier still have time to deliver the part p; if this deadline has passed and the part has not been received, this feature assumes the value $\overline{\text{A}}$, otherwise it remains A.
- **Engine condition** – f_3. Is the engine e repaired by engine manufacturer em? Is the engine e handed over to the airline operator ao? If the engine is not repaired, then this feature assumes the value $\overline{\text{R}}$; if the engine is repaired but not delivered, it assumes the value R; if the engine is repaired and delivered, it assumes the value D.
- **Repair deadline** – f_4. Has em repaired the engine e by the deadline in the aftercare contract ac with ao? Initially, this feature assumes the value B, which means that em have time to repair and deliver e; if this deadline has passed and the engine has not been repaired and delivered, it assumes the value $\overline{\text{B}}$.

For instance, the initial state $\overline{\text{O}}\text{A}\overline{\text{R}}\text{B}$ means the part p has not been ordered, the engine e is neither repaired nor delivered, and the engine manufacturer em has satisfied both deadlines. In this work, deadlines are represented simply as features of the state space. A more sophisticated treatment of time in normative systems, such as for instance the use of branching time logic [10] is certainly interesting, but not necessary for the purpose of this article.

The action space is composed of six actions: order an engine part from a supplier, receive an ordered part, repair an engine, deliver an engine, pay a penalty and charge a penalty.

$$\begin{aligned}\mathcal{A} = \{\ & \text{order}(\text{Part}, \text{Supplier}),\\ & \text{receive}(\text{Part}, \text{Supplier}),\\ & \text{repair}(\text{Engine}),\\ & \text{deliver}(\text{Engine}, \text{Operator}),\\ & \text{pay}(\text{Amount}, \text{Operator}),\\ & \text{charge}(\text{Amount}, \text{Supplier})\ \}\end{aligned}$$

All actions in the process are assumed to be deterministic, except for receive and repair which in some states are not. We specify part of the transition model, showing only the non-deterministic transitions:

$$\begin{aligned}\mathcal{T}_1 = \{\ & (\text{O}\overline{\text{A}}\overline{\text{R}}\text{B}, \text{receive}(\text{p}, \text{sx}), \text{X}\overline{\text{A}}\overline{\text{R}}\text{B}) \rightarrow 0.85,\\ & (\text{O}\overline{\text{A}}\overline{\text{R}}\text{B}, \text{receive}(\text{p}, \text{sx}), \text{X}\overline{\text{A}\text{R}}\text{B}) \rightarrow 0.10,\\ & (\text{O}\overline{\text{A}}\overline{\text{R}}\text{B}, \text{receive}(\text{p}, \text{sx}), \text{X}\overline{\text{A}\text{R}\text{B}}) \rightarrow 0.05,\\ & (\text{X}\overline{\text{A}}\overline{\text{R}}\text{B}, \text{repair}(\text{e}), \text{XA}\overline{\text{R}}\text{B}) \rightarrow 0.95,\\ & (\text{X}\overline{\text{A}}\overline{\text{R}}\text{B}, \text{repair}(\text{e}), \text{XA}\overline{\text{R}\text{B}}) \rightarrow 0.05,\\ & (\text{X}\overline{\text{A}\text{R}}\text{B}, \text{repair}(\text{e}), \text{X}\overline{\text{A}}\text{RB}) \rightarrow 0.95,\\ & (\text{X}\overline{\text{A}\text{R}}\text{B}, \text{repair}(\text{e}), \text{X}\overline{\text{A}}\text{R}\overline{\text{B}}) \rightarrow 0.05,\\ & \ldots\ \}\end{aligned}$$

$$\begin{aligned}\mathcal{T}_2 = \{\ & (\text{O}\overline{\text{A}}\overline{\text{R}}\text{B}, \text{receive}(\text{p}, \text{sy}), \text{Y}\overline{\text{A}}\overline{\text{R}}\text{B}) \rightarrow 0.90,\\ & (\text{O}\overline{\text{A}}\overline{\text{R}}\text{B}, \text{receive}(\text{p}, \text{sy}), \text{Y}\overline{\text{A}\text{R}}\text{B}) \rightarrow 0.09,\\ & (\text{O}\overline{\text{A}}\overline{\text{R}}\text{B}, \text{receive}(\text{p}, \text{sy}), \text{Y}\overline{\text{A}\text{R}\text{B}}) \rightarrow 0.01,\\ & (\text{Y}\overline{\text{A}}\overline{\text{R}}\text{B}, \text{repair}(\text{e}), \text{YA}\overline{\text{R}}\text{B}) \rightarrow 0.95,\\ & (\text{Y}\overline{\text{A}}\overline{\text{R}}\text{B}, \text{repair}(\text{e}), \text{YA}\overline{\text{R}\text{B}}) \rightarrow 0.05,\\ & (\text{Y}\overline{\text{A}\text{R}}\text{B}, \text{repair}(\text{e}), \text{Y}\overline{\text{A}}\text{RB}) \rightarrow 0.95,\\ & (\text{Y}\overline{\text{A}\text{R}}\text{B}, \text{repair}(\text{e}), \text{Y}\overline{\text{A}}\text{R}\overline{\text{B}}) \rightarrow 0.05,\\ & \ldots\ \}\end{aligned}$$

$\mathcal{T}_3$ = { (OA$\overline{\text{R}}$B, receive(p, sz), ZA$\overline{\text{R}}$B)$\rightarrow 0.98$,
(OA$\overline{\text{R}}$B, receive(p, sz), Z$\overline{\text{A}}\overline{\text{R}}$B)$\rightarrow 0.01$,
(OA$\overline{\text{R}}$B, receive(p, sz), Z$\overline{\text{ARB}}$)$\rightarrow 0.01$,
(ZA$\overline{\text{R}}$B, repair(e), ZARB)$\rightarrow 0.95$,
(ZA$\overline{\text{R}}$B, repair(e), ZAR$\overline{\text{B}}$)$\rightarrow 0.05$,
(Z$\overline{\text{A}}\overline{\text{R}}$B, repair(e), Z$\overline{\text{A}}$RB)$\rightarrow 0.95$,
(Z$\overline{\text{A}}\overline{\text{R}}$B, repair(e), Z$\overline{\text{A}}R\overline{\text{B}}$)$\rightarrow 0.05$,
$\dots$ }

According to the transition model, receive changes the engine part feature from O to X, Y or Z, depending on the supplier; if the engine part is not received from the supplier by the deadline, then the order deadline feature changes from A to $\overline{\text{A}}$. The action repair changes the repair feature from $\overline{\text{R}}$ to R; if the repair is not performed by the deadline, the repair deadline feature changes from B to $\overline{\text{B}}$.

Dealing with different counterparts may provide different rewards, so we specify a reward function $\mathcal{R}_i$ for each nmdp_i. We have assigned rewards to actions instead of states or transitions. This decision is supported by the fact that the penalties for norm violations are associated with action parameters, more specifically, with the actions charge and pay.

$\mathcal{R}_1$ = {order(p, sx) $\rightarrow -1.0$,
receive(p, sx) $\rightarrow 0.0$,
charge(1.0, sx) $\rightarrow 1.0$,
deliver(e, ao) $\rightarrow 3.0$,
repair(e) $\rightarrow -1.0$,
pay(-2.0, ao) $\rightarrow -2.0$ }

$\mathcal{R}_2$ = {order(p, sy) $\rightarrow -0.8$,
receive(p, sy) $\rightarrow 0.0$,
charge(1.0, sy) $\rightarrow 1.0$,
deliver(e, ao) $\rightarrow 3.0$,
repair(e) $\rightarrow -1.0$,
pay(-2.0, ao) $\rightarrow -2.0$ }

$\mathcal{R}_2$ = {order(p, sz) $\rightarrow -1.5$,
receive(p, sz) $\rightarrow 0.0$,
charge(1.5, sz) $\rightarrow 1.5$,
deliver(e, ao) $\rightarrow 3.0$,
repair(e) $\rightarrow -1.0$,
pay(-2.0, ao) $\rightarrow -2.0$ }

The first contract of our example is an aftercare contract between the engine manufacturer em and the airline operator ao. This contract, named ac, includes the following norms:

- n_1: an obligation on the engine manufacturer em to repair and deliver the engine by the repair deadline B; if this norm is violated then the engine manufacturer will have to pay a penalty to the airline operator ao in order to accomplish the delivery (paying the penalty resets the repair deadline); the reward for paying the penalty is -2.0; any transition that arrives at states with the repair deadline feature equal to $\overline{\text{B}}$ characterizes a violation;
- n_2: a prohibition on the manufacturer em to order parts from sy; if the manufacturer purchases a part from sy then the airline operator ao refuses the engine; this norm is violated if the manufacturer achieves any state with the feature engine part equal to Y.

In the supply contracts, the engine manufacturer em is not an addressee of the norms since the obligations are on the suppliers. However, the manufacturer, as the beneficiary, expects to receive the engine parts by the deadline. The following three norms correspond to the parts supply contracts with sx, sy and sz, respectively:

- n_3: an obligation on the supplier sx to produce and deliver the requested part p by the order deadline A; if this norm is violated, the engine manufacturer can charge a penalty on the part supplier sx for delaying the delivery (charging resets the order deadline); the reward for charging the penalty is 1.0; any transition that arrives at states with the order deadline feature equal to $\overline{\text{A}}$ characterizes a violation of this norm;
- n_4: as in n_3 except that sy replaces sx;
- n_5: as in n_3 except that sz replaces sx and the reward for charging the penalty is 1.5.

The specification of the set of norms is as follows:

$\mathcal{N}_1$ = { n_1, n_2, n_3 }
$\mathcal{N}_2$ = { n_1, n_2, n_4 }
$\mathcal{N}_3$ = { n_1, n_2, n_5 }

n_1 = (obligation, em, ao,
(f_3=D) $\wedge$ (f_4=B),
({ (XAR$\overline{\text{B}}$, pay(-2.0, ao), XARB, 1.0),
(X$\overline{\text{A}}$R$\overline{\text{B}}$, pay(-2.0, ao), X$\overline{\text{A}}$RB, 1.0) ,
(YAR$\overline{\text{B}}$, pay(-2.0, ao), YARB, 1.0),
(Y$\overline{\text{A}}$R$\overline{\text{B}}$, pay(-2.0, ao), Y$\overline{\text{A}}$RB, 1.0) ,
(ZAR$\overline{\text{B}}$, pay(-2.0, ao), ZARB, 1.0),
(Z$\overline{\text{A}}$R$\overline{\text{B}}$, pay(-2.0, ao), Z$\overline{\text{A}}$RB, 1.0) },

$\{$ $(\text{XAR}\overline{\text{B}}, \text{deliver}(\text{e}, \text{ao}), 0)$,
$(\text{X}\overline{\text{A}}\text{R}\overline{\text{B}}, \text{deliver}(\text{e}, \text{ao}), 0)$,
$(\text{YAR}\overline{\text{B}}, \text{deliver}(\text{e}, \text{ao}), 0)$,
$(\text{Y}\overline{\text{A}}\text{R}\overline{\text{B}}, \text{deliver}(\text{e}, \text{ao}), 0)$,
$(\text{ZAR}\overline{\text{B}}, \text{deliver}(\text{e}, \text{ao}), 0)$,
$(\text{Z}\overline{\text{A}}\text{R}\overline{\text{B}}, \text{deliver}(\text{e}, \text{ao}), 0)$,
$(\text{XAR}\overline{\text{B}}, \text{pay}(-2.0, \text{ao}), 1)$,
$(\text{X}\overline{\text{A}}\text{R}\overline{\text{B}}, \text{pay}(-2.0, \text{ao}), 1)$,
$(\text{YAR}\overline{\text{B}}, \text{pay}(-2.0, \text{ao}), 1)$,
$(\text{Y}\overline{\text{A}}\text{R}\overline{\text{B}}, \text{pay}(-2.0, \text{ao}), 1)$,
$(\text{ZAR}\overline{\text{B}}, \text{pay}(-2.0, \text{ao}), 1)$,
$(\text{Z}\overline{\text{A}}\text{R}\overline{\text{B}}, \text{pay}(-2.0, \text{ao}), 1)$ $\}$))

n_2 = (prohibition, em, ao,
(f_1=Y),
($\{$ $\}$,
$\{$ $(\text{YARB}, \text{deliver}(\text{e}, \text{ao}), 0)$,
$(\text{YAR}\overline{\text{B}}, \text{deliver}(\text{e}, \text{ao}), 0)$,
$(\text{Y}\overline{\text{A}}\text{RB}, \text{deliver}(\text{e}, \text{ao}), 0)$,
$(\text{Y}\overline{\text{A}}\text{R}\overline{\text{B}}, \text{deliver}(\text{e}, \text{ao}), 0)$ $\}$))

n_3 = (obligation, sx, em,
(f_1=X) $\wedge$ (f_2=A),
($\{$ $(\text{X}\overline{\text{A}}\text{R}\overline{\text{B}}, \text{charge}(1.0, \text{sx}), \text{XA}\overline{\text{R}}\overline{\text{B}}, 1.0)$,
$(\text{X}\overline{\text{A}}\text{R}\overline{\text{B}}, \text{charge}(1.0, \text{sx}), \text{XA}\overline{\text{R}}\text{B}, 1.0)$ $\}$,
$\{$ $(\text{X}\overline{\text{A}}\text{R}\overline{\text{B}}, \text{charge}(1.0, \text{sx}), 1)$,
$(\text{X}\overline{\text{A}}\text{RB}, \text{charge}(1.0, \text{sx}), 1)$ $\}$))

n_4 = (obligation, sy, em,
(f_1=Y) $\wedge$ (f_2=A),
($\{$ $(\text{Y}\overline{\text{A}}\text{R}\overline{\text{B}}, \text{charge}(1.0, \text{sy}), \text{YA}\overline{\text{R}}\overline{\text{B}}, 1.0)$,
$(\text{Y}\overline{\text{A}}\text{R}\overline{\text{B}}, \text{charge}(1.0, \text{sy}), \text{YA}\overline{\text{R}}\text{B}, 1.0)$ $\}$,
$\{$ $(\text{Y}\overline{\text{A}}\text{R}\overline{\text{B}}, \text{charge}(1.0, \text{sy}), 1)$,
$(\text{Y}\overline{\text{A}}\text{RB}, \text{charge}(1.0, \text{sy}), 1)$ $\}$))

n_5 = (obligation, sz, em,
(f_1=Z) $\wedge$ (f_2=A),
($\{$ $(\text{Z}\overline{\text{A}}\text{R}\overline{\text{B}}, \text{charge}(1.5, \text{sz}), \text{ZA}\overline{\text{R}}\overline{\text{B}}, 1.0)$,
$(\text{Z}\overline{\text{A}}\text{R}\overline{\text{B}}, \text{charge}(1.5, \text{sz}), \text{ZA}\overline{\text{R}}\text{B}, 1.0)$ $\}$,
$\{$ $(\text{Z}\overline{\text{A}}\text{R}\overline{\text{B}}, \text{charge}(1.5, \text{sz}), 1)$,
$(\text{Z}\overline{\text{A}}\text{RB}, \text{charge}(1.5, \text{sz}), 1)$ $\}$))

5. Identifying contract violations

The present section explains how to identify which states violate which norms and describes how the respective sanctions are associated to these norm violating states. This procedure is specified through an algorithm and illustrated with examples of the engine manufacturer agent em introduced in the previous section.

First of all, we construct sets of states that match the *normative content* of the norms. This process is carried out by selecting those states whose features match the features specified within the normative content. Then, from these sets we obtain information regarding which states are obliged or forbidden.

Let $\mathcal{E}_{ji}$ be the set of states matching the content of $\text{n}_j \in \mathcal{N}_i$. For instance, the following sets contain states from $\mathcal{S}_1$, $\mathcal{S}_2$ and $\mathcal{S}_3$, respectively, which match the normative content of n_1:

$$\mathcal{E}_{11} = \{\text{XADB}, \text{X}\overline{\text{A}}\text{DB}\}$$
$$\mathcal{E}_{12} = \{\text{YADB}, \text{Y}\overline{\text{A}}\text{DB}\}$$
$$\mathcal{E}_{13} = \{\text{ZADB}, \text{Z}\overline{\text{A}}\text{DB}\}$$

Once we have computed the set of states matching the normative content, we check the state space $\mathcal{S}_i$ in order to find the violating states. They are identified as follows: If the deontic modality of a given norm n_j is an *obligation*, then the addressee agent must be capable of achieving at least one state in $\mathcal{E}_{ji}$ so as to comply with the norm. If none of these states can be achieved given the agent's capabilities and current state, then the obligation n_j is considered violated. On the other hand, if this norm is a *prohibition*, then no state in $\mathcal{E}_{ji}$ should be visited. If the agent's current state is an element of $\mathcal{E}_{ji}$, then the agent is transgressing the prohibition n_j.

Algorithm 1 specifies how to identify which states in $\mathcal{S}_i$ violate which norms in $\mathcal{N}_i$. The first input of this algorithm is mdp_i, which is a replica of nmdp_i without the set of norms $\mathcal{N}_i$. This MDP will be modified along the algorithm execution in order to express the effect of the sanctions. The other two inputs are the set of states $\mathcal{S}_i$ and the set of norms $\mathcal{N}_i$ of nmdp_i.

For each state $\text{s} \in \mathcal{S}_i$, the algorithm determines if the norm n_j is violated in this state. Let ψ_{ji} be a set, initially empty, for storing states from $\mathcal{S}_i$ that violate the norm n_j, and $\mathcal{E}_{ji}$ the set of states from $\mathcal{S}_i$ matching the normative content of n_j. If the norm is an *obligation*, the algorithm verifies if there is *at least one* state $\text{s'} \in \mathcal{E}_{ji}$ which is reachable from s. In other words, if there is a path starting at s and ending at s' then we say the agent is still capable of fulfilling the obligation in the state s. For example, the obligation $\text{n}_1 \in \mathcal{N}_1$ is not obeyed in the set of states ψ_{11}, from which it is not possible to achieve any state in $\mathcal{E}_{11}$:

$$\psi_{11} = \{\ \text{XA}\overline{\text{RB}}, \text{X}\overline{\text{ARB}},$$
$$\text{XAR}\overline{\text{B}}, \text{X}\overline{\text{A}}\text{R}\overline{\text{B}},$$
$$\text{XAD}\overline{\text{B}}, \text{X}\overline{\text{A}}\text{D}\overline{\text{B}}\ \}$$

Algorithm 1

Input: mdp_i : initial MDP,
$\mathcal{S}_i$: set of states, $\mathcal{N}_i$: set of norms,

1. **for all** $(\mathrm{n}_j \in \mathcal{N}_i)$ **do**
2. $\quad\psi_{ji} \leftarrow \{\,\}$
3. $\quad\mathcal{E}_{ji} \leftarrow match(\mathrm{n}_j, \mathcal{S}_i)$
4. $\quad$**for all** $(\mathrm{s} \in \mathcal{S}_i)$ **do**
5. $\quad\quad$**if** (n_j is obligation) **then**
6. $\quad\quad\quad$violation $\leftarrow$ true
7. $\quad\quad\quad$**for all** $(\mathrm{s}' \in \mathcal{E}_{ji})$ **do**
8. $\quad\quad\quad\quad$**if** (s' is reachable from s) **then**
9. $\quad\quad\quad\quad\quad$violation $\leftarrow$ false
10. $\quad\quad\quad\quad\quad$*break*
11. $\quad\quad\quad\quad$**end if**
12. $\quad\quad\quad$**end for**
13. $\quad\quad\quad$**if** (violation $=$ true) **then**
14. $\quad\quad\quad\quad\psi_{ji} \leftarrow \psi_{ji} \cup \{\mathrm{s}\}$
15. $\quad\quad\quad\quad\Lambda_j \leftarrow sanction(\mathrm{n}_j)$
16. $\quad\quad\quad\quad represent(\Lambda_j, \mathrm{s}, \mathrm{mdp}_i)$
17. $\quad\quad\quad$**end if**
18. $\quad\quad$**else**
19. $\quad\quad\quad$**if** (n_j is prohibition) **then**
20. $\quad\quad\quad\quad$**if** $(\mathrm{s} \in \mathcal{E}_{ji})$ **then**
21. $\quad\quad\quad\quad\quad\psi_{ji} \leftarrow \psi_{ji} \cup \{\mathrm{s}\}$
22. $\quad\quad\quad\quad\quad\Lambda_j \leftarrow sanction(\mathrm{n}_j)$
23. $\quad\quad\quad\quad\quad represent(\Lambda_j, \mathrm{s}, \mathrm{mdp}_i)$
24. $\quad\quad\quad\quad$**end if**
25. $\quad\quad\quad$**end if**
26. $\quad\quad$**end if**
27. $\quad$**end for**
28. **end for**

If the norm is a *prohibition*, then the algorithm verifies if s is an element of the set $\mathcal{E}_{ji}$. If $\mathrm{s} \in \mathcal{E}_{ji}$ then s violates n_j. For example, the subset of states from $\mathcal{S}_2$ that violate the prohibition $\mathrm{n}_2 \in \mathcal{N}_2$ is:

$$\begin{aligned}\psi_{22} = \{\ &\textsc{yarb}, \textsc{yar}\overline{\textsc{b}},\\ &\textsc{y}\overline{\textsc{a}}\textsc{rb}, \textsc{y}\overline{\textsc{a}}\textsc{r}\overline{\textsc{b}},\\ &\textsc{yadb}, \textsc{yad}\overline{\textsc{b}},\\ &\textsc{y}\overline{\textsc{a}}\textsc{db}, \textsc{y}\overline{\textsc{a}}\textsc{d}\overline{\textsc{b}}\ \}\end{aligned}$$

Another approach to identify states that violate prohibitions consists of checking if all paths from a given state lead only to prohibited states. Compared to this, our approach has the advantage of avoiding the search for paths, which would have a negative impact on the computational complexity of the reasoning process.

For each norm violation found, the algorithm represents the respective sanction with the violating state by calling the function $represent$. This function modifies $\mathcal{T}_i$ and $\mathcal{C}_i$ of mdp_i in order to represent the outcomes of sanction Λ_j within the state s. At the end of the execution of Algorithm 1, mdp_i will include the representation of sanctions and ψ_{ji} will contain all states from $\mathcal{S}_i$ that violate n_j.

Figure 2 illustrates mdp_1 which results from executing Algorithm 1 with nmdp_1. From *outcome*$_{\mathcal{C}}$ of the norms in the contract, we obtain the actions that must be enabled or disabled, while from *outcome*$_{\mathcal{T}}$ we obtain the transition probabilities that must be updated. In Figure 2, the enabled actions are written in bold font, while the disabled actions are written in grey font. The remaining actions are the original ones from nmdp_1, as specified in Section 4.

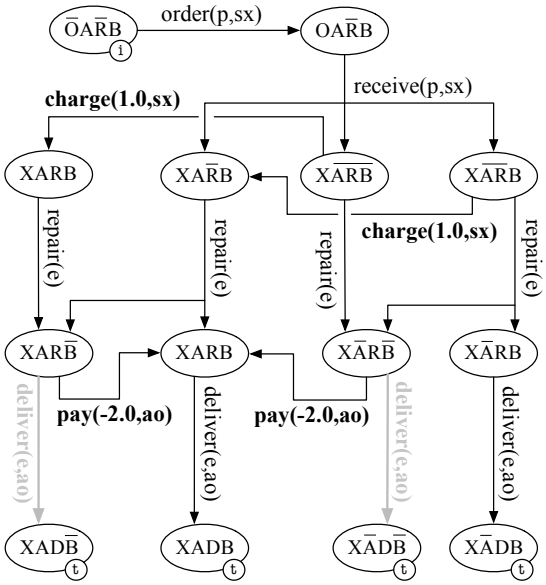

Fig. 2. mdp_1 resulting from running Algorithm 1 with nmdp_1.

6. Contract signing and risks

In the present work, the decision concerning which contracts to sign is supported by the expected utilities (EUs) calculated for each MDP (self-interested strategy). To calculate the policies and their EUs, we have employed the well-known Policy Iteration algorithm [23]. Thus, for each mdp_i we construct a policy π_i. In Figure 3, we illustrate the Markov Chain (MC) resulting from the combination of mdp_1 and its optimal policy π_1^*. The grey boxes indicate which norms from $\mathcal{N}_1$ are violated in which states.

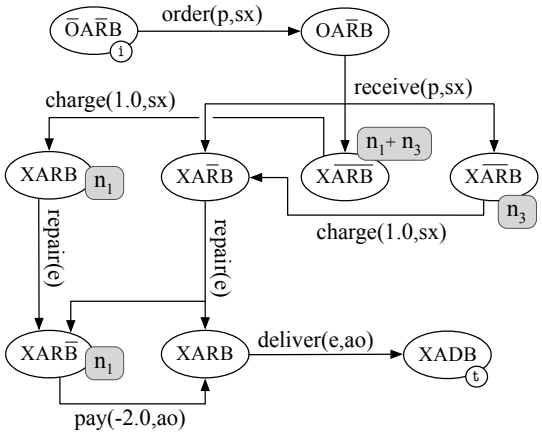

Fig. 3. Optimal policy for mdp_1.

In order to decide which contracts to sign, we compare the EU for the initial state of each mdp_i, and then we choose the one that maximizes the expected earnings. In our aerospace aftermarket example, the engine manufacturer decides to sign the contracts in $\mathcal{W}_1$ given that mdp_1 provides the highest EU at $\overline{\mathrm{O}}\mathrm{A}\overline{\mathrm{R}}\mathrm{B}$:

$$\mathrm{EU}(\overline{\mathrm{O}}\mathrm{A}\overline{\mathrm{R}}\mathrm{B}, \mathrm{mdp}_1) = 0.955$$
$$\mathrm{EU}(\overline{\mathrm{O}}\mathrm{A}\overline{\mathrm{R}}\mathrm{B}, \mathrm{mdp}_2) = -1.699$$
$$\mathrm{EU}(\overline{\mathrm{O}}\mathrm{A}\overline{\mathrm{R}}\mathrm{B}, \mathrm{mdp}_3) = 0.411$$

In this case study, $\mathcal{W}_2$ is not optimal primarily because of the costs of prohibition n_2, and $\mathcal{W}_3$ is not optimal primarily because the part supplier sz has a high probability of not delivering on time.

In Section 5 we have described how to identify the states in which norms are broken. Knowing these violating states, we can estimate the contract risks (probability of violating the norms in the contract) under a given policy. To do so, we combine mdp_i and the computed policy π_i, which results in a MC, as shown in Figure 3. By calculating the probability of reaching the states that violate a given norm, we know the violation probability of this norm.

Algorithm 2 shows how to estimate the risks within a given contract by calculating the probability of violating each norm in this contract. This algorithm has three inputs: mc_i, the MC resulting from the combination of mdp_i and its respective policy π_i; s_0, the initial state of the process; and ψ_{ji}, the set of states from $\mathcal{S}_i$ that violate the norm n_j. Algorithm 2 begins by setting all states in ψ_{ji} as terminal (absorbing). Then, it calculates the absorption probability in these states with the function ap (this function is implemented using the fundamental matrix of the MC; for details on the computation of absorption probabilities, see [36]). Such values are summed and stored in the variable ω_{ji} which corresponds to the chance of violating the norm n_j starting from the initial state s_0 under the policy π_i.

Algorithm 2

Input: mc_i markov chain,
s_0 initial state,
ψ_{ji} set of violating states

1. **for all** $(\mathrm{s} \in \psi_{ji})$ **do**
2. $\quad terminal(\mathrm{s}, \mathrm{mc}_i)$
3. **end for**
4. $\omega_{ji} \leftarrow 0.0$
5. **for all** $(\mathrm{s} \in \psi_{ji})$ **do**
6. $\quad \omega_{ji} \leftarrow \omega_{ji} + ap(\mathrm{s}, \mathrm{s}_0, \mathrm{mc}_i)$
7. **end for**

In Figure 4, we show the probabilities of violating n_1 and n_3 by signing the contracts in $\mathcal{W}_1$ and running the policy π_1^* illustrated in Figure 3. The absorbing states that characterize the violation of these norms are shaded grey and labelled with the probability of reaching them. The probability of violating n_1 is 9.75% and n_3 is 15%. In this case, n_2 is not violated since the manufacturer em does not purchase the required engine part from the supplier sy.

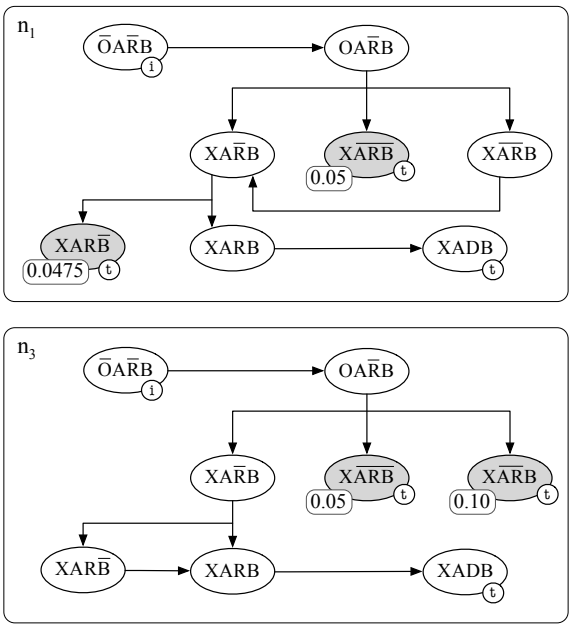

Fig. 4. Risks of violating n_1 and n_3 by signing the contracts in $\mathcal{W}_1$.

7. Conclusion and future work

This paper proposes the NMDP framework, which extends the well-known MDP framework so as to express normative structures. Since the norms are specified as a separate parameter in the NMDP (not *hard-coded* in the process), the agent is capable of deciding whether to comply with them, and capable of adapting its behaviour when they change. Moreover, we specify two general algorithms, one for identifying the states of the NMDP in which norms are broken, and the other for calculating the violation probability of each norm in the NMDP.

To demonstrate the validity of our approach, we develop a case study of normative reasoning over contracts in a simulated aerospace aftermarket domain. In this case study, we specify an engine manufacturer agent which evaluates sets of contracts by calculating their expected utilities and violation probabilities.

As future work we intend to study the relation between the minimization of contract violations, utility maximization and the agents' reputation. Before signing contracts, the signatories take into account their past experiences, which provide data to construct the reputation of the agents. In the NMDP framework, the decay of reputation reputation can be measured in terms of estimated loss of utility due to lack of future opportunities. Our second research direction aims at generating detailed explanations of norm violations within the NMDP framework, identifying their direct and indirect causes as well as the involved agents. Finally, our last research direction consists of studying normative structures and factored representations [22] to improve the efficiency of the policy construction algorithms.

Acknowledgment

This research is supported by the Spanish Ministry of Science and Innovation through grants TIN2009-13839-C03-02 (co-funded by Plan E) and CSD2007-0022 (CONSOLIDER-INGENIO 2010)

References

[1] T. Ågotnes, W. van der Hoek and M. Wooldridge, Normative system games, *Proceedings of the 6th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS'07)*, E. H. Durfee, M. Yokoo, M. N. Huhns, O. Shehory, eds, IFAAMAS, Honolulu, Hawaii, USA, 2007, pp. 1-8.

[2] G. Andrighetto, R. Conte, M. Campenni and M. Paolucci, On the immergence of norms: a normative agent architecture, in: *Proceedings of AAAI Fall Symposium, Social and Organizational Aspects of Intelligence*, Washington DC., USA, The AAAI Press, Technical Report FS-07-04, 2007.

[3] R. Bellman, A Markovian Decision Process, *Journal of Mathematics and Mechanics*, **6** (1957), 679-684.

[4] G. Boella and L. van der Torre, A game theoretic approach to contracts in multiagent systems, *IEEE Transactions on Systems, Man, and Cybernetics, Part C* **36**(1) (2006), 68-79.

[5] G. Boella and L. van der Torre, Substantive and procedural norms in normative multiagent systems, *Journal of Applied Logic* **6**(2) (2008), 152-171.

[6] M. Boman, Norms in Artificial Decision Making, *Artificial Intelligence and Law*, **7**(1) (1999), 17-35.

[7] C. Boutilier, T. Dean and S. Hanks, Decision-theoretic planning: Structural assumptions and computational leverage, *Journal of Artificial Intelligence Research (JAIR)* **11** (1999), 1-94.

[8] F. M. T. Brazier, C. M. Jonker and J. Treur, Compositional Design and Reuse of a Generic Agent Model, *Applied Artificial Intelligence*, **14**(5) (2000), 491-538.

[9] J. Broersen, M. Dastani, J. Hulstijn and L. van der Torre, Goal Generation in the BOID Architecture, *Cognitive Science Quarterly Journal*, **2**(3-4) (2002), 428-447.

[10] J. Broersen, F. Dignum, V. Dignum and J. J. Meyer, Designing a Deontic Logic of Deadlines, in: *Proceedings of the 7th International Conference on Deontic Logic in Computer Science (DEON'04)*, Madeira, Portugal, A. Lomuscio, D. Nute, eds, LNAI, Vol. 3065, Springer, 2004, pp. 43-56.

[11] H. L. Cardoso and E. C. Oliveira, Adaptive Deterrence Sanctions in a Normative Framework, in: *Proceedings of the 2009 IEEE/WIC/ACM International Conference on Intelligent Agent Technology (IAT'09)*, IEEE, Milan, Italy, 2009, pp. 36-43.

[12] A. Casali, L. Godo and C. Sierra, Graded BDI Models for Agent Architectures, in: *5th International Workshop on Computational Logic in Multi-Agent Systems (CLIMA)*, Lisbon, Portugal, J. A. Leite and P. Torroni, eds, LNCS, Vol. 3487, Springer, 2004, pp. 126-143.

[13] A. Casali, L. Godo and C. Sierra, A graded BDI agent model to represent and reason about preferences, *Artificial Intelligence*, **175**(7-8) (2011), 1468-1478.

[14] C. Castelfranchi, F. Dignum, C. Jonker and J. Treur, Deliberative Normative Agents: Principles and Architecture, in: *6th International Workshop on Intelligent Agents VI, Agent Theories, Architectures and Languages (ATAL'99)*, Orlando, Florida, USA, N. R. Jennings, Y. Lespérance, eds, LNCS, Vol. 1757, Springer, 1999, pp. 364-378.

[15] R. Centeno, H. Billhardt, R. Hermoso and S. Ossowski, Organising MAS: A Formal Model Based on Organisational Mechanisms, in: *Proceedings of the 2009 ACM Symposium on Applied Computing (SAC'09)*, Honolulu, Hawaii, USA, March 9-12, 2009. pp. 740-746. ACM (2009)

[16] N. Criado, E. Argente and V. J. Botti, Open issues for normative multi-agent systems, *AI Communications*, **24**(3) (2011), 233-264.

[17] M. Dastani, J. Hulstijn, L. van der Torre, How to decide what to do?, *European Journal of Operational Research*, **160**(3) (2005), 762-784.

[18] P. Dellunde, L. Godo, Introducing Grades in Deontic Logics, in: *9th International Conference on Deontic Logic in Computer Science (DEON'08)*, Luxembourg, R. van der Meyden, L. van der Torre, eds, LNAI, Vol. 5076, Springer, 2008, pp. 248-262.

[19] F. Dignum, D. N. Morley, L. Sonenberg and L. Cavedon, Towards socially sophisticated BDI agents, in: Proceedings of the 4th International Conference on MultiAgent Systems (ICMAS'00), E. Durfee, eds, IEEE Computer Society Press, Boston, Massachusetts, USA, 2000, pp. 111-118.

[20] A. García-Camino, P. Noriega, J. A. Rodríguez-Aguilar, Implementing norms in electronic institutions, *Proceedings of the 4th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS'05)*, F. Dignum, V. Dignum, S. Koenig, S. Kraus, M. P. Singh, M. Wooldridge, eds, ACM, Utrecht, The Netherlands, 2005, pp. 667-673.

[21] A. García-Camino, J. A. Rodríguez-Aguilar, Carles Sierra and Wamberto Weber Vasconcelos, Constraint rule-based programming of norms for electronic institutions. *Autonomous Agents and Multi-Agent Systems*, **18**(1) (2009), 186-217.

[22] C. Guestrin, D. Koller, R. Parr, S. Venkataraman, Efficient Solution Algorithms for Factored MDPs, *Journal Artificial Intelligence Research (JAIR)*, **19** (2003), 399-468.

[23] R. A. Howard, *Dynamic Programming and Markov Processes*, The M.I.T. Press, Cambridge, USA, 1960.

[24] M. Jakob, M. Pechoucek, S. Miles and M. Luck, Case Studies for Contract-based Systems, in: *Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS'08): Industry and Applications Track*, M. Berger, B. Burg, S. Nishiyama, eds, IFAAMAS, Estoril, Portugal, 2008, pp. 55-62.

[25] S. Joseph, C. Sierra, M. Schorlemmer and P. Dellunde, Deductive Coherence and Norm Adoption, *Logic Journal of the IGPL*, **18**(1) (2010), 118-156.

[26] M. J. Kollingbaum and T. J. Norman, NoA - A Normative Agent Architecture, in: *Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI'03)*, Morgan Kaufmann, Acapulco, Mexico, 2003, pp. 1465-1466.

[27] M. J. Kollingbaum and T. J. Norman, Norm adoption and consistency in the NoA agent architecture, in: *1st International Workshop on Programming Multi-Agent Systems (PROMAS'03)*, Melbourne, Australia, Mehdi Dastani, Jurgen Dix and Amal El Fallah-Seghrouchni, eds, LNCS, Vol. 3067, Springer-Verlag, 2003, pp. 169-186.

[28] M. J. Kollingbaum and T. J. Norman, Norm Adoption in the NoA Agent Architecture, in: *Proceedings of the 2nd International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS'03)*, ACM, Melbourne, Australia, 2003, pp. 1038-1039.

[29] M. J. Kollingbaum and T. J. Norman, Informed deliberation during norm-governed practical reasoning, in: *AAMAS 2005 International Workshops*, Utrecht, The Netherlands, O. Boissier, J. A. Padget, V. Dignum, G. Lindemann, E. T. Matson, S. Ossowski, J. S. Sichman, J. Vázquez-Salceda, eds, LNCS, Vol. 3913, Springer-Verlag, 2006, pp. 183-197.

[30] F. López, M. Luck and M. dInverno, Constraining autonomy through norms, in: *Proceedings of the 1st International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS'02)*, ACM, Bologna, Italy, 2002, pp. 674-681.

[31] F. López, M. Luck and M. dInverno, A Normative Framework for Agent- Based Systems, in: *Normative Multi-agent Systems, Dagstuhl Seminar Proceedings*, Vol. 07122, G. Boella, L. van der Torre, H. Verhagen, eds, Internationales Begegnungs- und Forschungszentrum für Informatik (IBFI), Schloss Dagstuhl, Germany, 2007.

[32] F. Meneguzzi and M. Luck, Composing high-level plans for declarative agent programming, in: *5th International Workshop on Declarative Agent Languages and Technologies (DALT'05)*, Honolulu, USA, M. Baldoni, T. C. Son, M. B. van Riemsdijk and M. Winikoff, eds, LNCS, Vol. 4897, Springer-Verlag, 2007, pp. 69-85.

[33] F. Meneguzzi, S. Miles, C. Holt, M. Luck, N. Oren, S. Modgil, N. Faci, M. Kollingbaum, Electronic contracting in aircraft aftercare: A case study, in: *Proceedings of the 7th International Conference on Autonomous Agents and Multiagent Systems (AAMAS'08)*, L. Padgham, D. Parkes, J. Mller, S. Parsons, eds, IFAAMAS, Estoril, Portugal, 2008, pp. 63-70.

[34] F. Meneguzzi and M. Luck, Norm-based behaviour modification in BDI agents, in: *Proceedings of the 8th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS'09)*, C. Sierra, C. Castelfranchi, K. S. Decker, J. S. Sichman, eds, IFAAMAS, Budapest, Hungary, 2009, pp. 177-184.

[35] S. Modgil, N. Faci, F. R. Meneguzzi, N. Oren, S. Miles and M. Luck, A framework for monitoring agent-based normative systems, in: *Proceedings of the 8th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS'09)*, C. Sierra, C. Castelfranchi, K. S. Decker, J. S. Sichman, eds, IFAAMAS, Budapest, Hungary, 2009, pp. 153-160.

[36] J. Norris, *Markov Chains*, Cambridge University Press, USA, 1999.

[37] J. Oh, F. Meneguzzi and K. P. Sycara, Probabilistic plan recognition for intelligent information agents - towards proactive software assistant agents, in: *Proceedings of the 3rd International Conference on Agents and Artificial Intelligence (ICAART'11)*, SciTePress, Rome, Italy, 2011, pp. 281287.

[38] J. Oh, F. Meneguzzi, K. P. Sycara and T J. Norman, An agent architecture for prognostic reasoning assistance, in: *Proceedings of the 22nd International Joint Conference on Artificial Intelligence (IJCAI'11)*, IJCAI/AAAI, Barcelona, Catalonia, Spain, 2011, pp. 2513-2518.

[39] A. S. Rao, AgentSpeak(L): BDI Agents Speak Out in a Logical Computable Language, in: *7th European Workshop on Modelling Autonomous Agents in a Multi-Agent World (MAAMAW'96)*, Eindhoven, The Netherlands, W. V. de Velde, J. W. Perram, eds, LNCS, Vol. 1038, Springer, 1996, pp. 42-55.

[40] A. S. Rao and M. P. Georgeff, BDI Agents: From Theory to Practice, in: *Proceedings of the 1st International Conference on Multiagent Systems (ICMAS'95)*, V. R. Lesser, L. Gasser, eds, M.I.T. Press, 1995, pp. 312-319.

[41] P. Thagard, *Coherence in Thought and Action (Life and Mind: Philosophical Issues in Biology and Psychology)*, The MIT Press, 2002.

[42] L. van der Torre and Y. H. Tan, Contrary-to-Duty Reasoning with Preference-based Dyadic Obligations, *Annals of Mathematics and Artificial Intelligence*, **27**(1-4) (1999), 49-78.